Chapter 3
Learning Correlations

Abstract After a short introduction of the general concept of decision rule to relate input and target features, this chapter describes some generic and most popular methods for learning correlations over two or more features. Four of them pertain to quantitative targets (linear regression, canonical correlation, neural network, and regression tree), and seven to categorical ones (linear discrimination, support vector machine, naïve Bayes classifier, classification tree, contingency table, distance between partition and ranking relations, and the correspondence analysis). Of these, classification trees are treated in a most detailed way including a number of theoretical results that are not well known. These establish firm relations between popular scoring functions and bivariate measures—Quetelet indexes in contingency tables and, rather unexpectedly, normalization options for dummy variables representing target categories. Some related concepts such as Bayesian decision rules, bag-of-word model in text analysis, VC-dimension and kernel for non-linear classification are introduced too. The Chapter outlines several important characteristics of summarization and correlation between two features, and displays some of the properties of those. They are:

- linear regression and correlation coefficient for two quantitative variables (Sect. 3.2);
- tabular regression and correlation ratio for the mixed scale case (Sect. 3.8.3); and
- contingency table, Quetelet index, statistical independence, and Pearson's chi-squared for two nominal variables; the latter is treated as a summary correlation measure, in contrast to the conventional view of it as just a criterion of statistical independence (Sect. 3.6.1); moreover, a few less known least-squares based concepts are outlined, including canonical correlation and correspondence analysis.

3.1 General: Decision Rules, Fitting Criteria, and Learning Protocols

To specify a problem of learning correlation in a data table, one has to distinguish between two parts in the feature set: *predictor*, or *input*, features and *target*, or *output*, features. Typically, the number of target features is small, and in generic tasks, there is just one target feature. Target features are usually difficult to measure

B. Mirkin, *Core Data Analysis: Summarization, Correlation, and Visualization*, Undergraduate Topics in Computer Science,
https://doi.org/10.1007/978-3-030-00271-8_3

or impossible to know beforehand. This is why one would want to derive a decision rule relating predictors and targets so that prediction of targets can be made after measuring predictors only. Examples of learning problems include:

(a) chemical compounds: input features are of the molecular structure, whereas target features are activities such as toxicity or healing effects;
(b) types of grain in agriculture: input features are those of the seeds, soil and weather, and target features are of productivity and protein contents,
(c) industrial enterprises: input features refer to technology, investment and labor policies, whereas target features are of sales and profits;
(d) postcode districts in marketing research: input features refer to demographic, social and economic characteristics of the district residents, target features—to their purchasing behavior;
(e) bank loan customers: input features characterize demographic and income, whereas output features are of (potentially) bad debt;
(f) gene expression data: input features relate to levels of expression of DNA materials in the earlier stages of an illness, and output features to those at later stages.

A *decision rule* predicts values of target features from values of input features. A rule is referred to as a classifier if the target is categorical and as a regression if the target is quantitative. A generic categorical target problem is defined by specifying just a subset of entities labeled as belonging to the class of interest—the correlation problem in this case would be of building such a decision rule that would recognize, for each of the entities, whether it belongs to the labeled class or not. A generic regression problem—the bivariate linear regression—is considered in Sect. 3.1; its extension to the multivariate case is described in Sect. 3.3.

A decision rule is learnt over a dataset in which values of the targets are available. These data are frequently referred to as the training data. The idea underlying the process of learning is to look at the difference between predicted and observed target feature values on the training data set and to minimize them over a class of admissible rules. The structure of such a process is presented on the upper part of Fig. 3.1.

The notion that it ought to be a class of admissible rules pre-specified emerges because the training data is finite and, therefore, can be fit exactly by using a sufficient number of parameters. However, this would be valid on the training set only, because the fit would capture all the errors and noise inevitable in data collecting processes. Take a look, for example, at the 2D regression problem on Fig. 3.2 depicting seven points on (x,u)-plane corresponding to observed combinations of input feature x and target feature u.

The seven points on Fig. 3.2 can be exactly fitted by a polynomial of 6th order $u = p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6$. Indeed, they would lead to 7 equations $u_i = p(x_i)$ $(i = 1,\ldots,7)$, so that, in a typical case, the 7 coefficients a_k of the polynomial can be exactly determined. Having N points observed would require an $(N\text{-}1)$-th degree polynomial to exactly fit them.

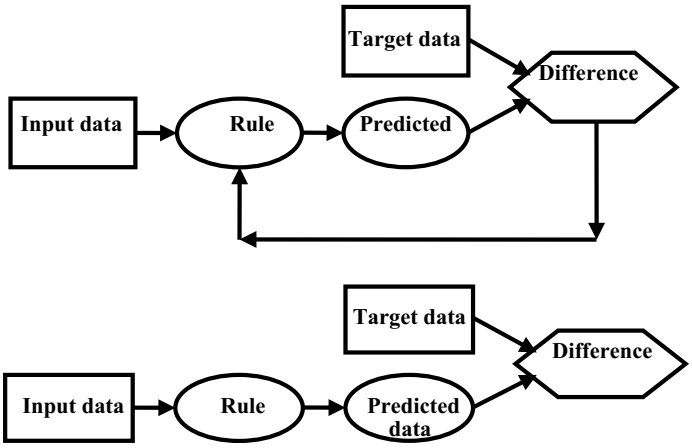

Fig. 3.1 Structure of a training/testing problem: In training, on the top, the decision rule is fitted to minimize the difference between the predicted and observed target data. In testing, the bottom part, the rule is used to predict so that no feedback to the rule is utilized

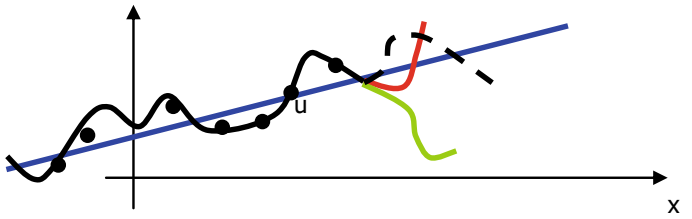

Fig. 3.2 Possible graphs of interrelation between x and u according to observed data points (black circles)

However, the polynomial, on which graph all the observations lie, has no predictive power both within and beyond the range. The curve may go either course (like those shown) depending on small changes in the data. The power of a theory —and a regression line is a theory in this case—rests on its generalization power, which, in this case, can be cast down as the relation between the number of observations and the number of parameters: the greater the better. When this ratio is relatively small, statisticians would refer to this as an *over-fitted* rule. The overfitting normally produce very poor predictions on newly added observations. The blue straight line fits none of the points, but it expresses a simple and very robust tendency and should be preferred because it summarizes the data much deeper: the seven observations are summarized here in just two parameters, slope and intercept, whereas the polynomial line provides no summary: it involves as many parameters as the data entities. This is why, in learning decision rules problems, a class of admissible rules should be selected first. Unfortunately, as of this moment, there is

no model based advice, within the data analysis discipline, on how this can be done, except very general ones like "look at the shapes of scatter plots". If there is no domain knowledge to choose a class of decision rules to fit, it is hard to tell what class of decision rules to use.

A most popular advice relates to the so-called *Occam's razor*, which means that the complexity of the data should be balanced by the simplicity of the decision rule. A British monk philosopher William Ockham (*c*. 1285–1349) made a claim: "Entities should not be multiplied unnecessarily." This is usually interpreted as saying that all other things being equal, the simplest explanation tends to be the best one. Operationally, this is further translated as the Principle of Maximum Parsimony, which is referred to when there is nothing better available. In the format of the so-called "Minimum description length" principle, this approach can be meaningfully applied to problems of estimation of parameters of statistic distributions (see Grünwald 2007). Somewhat wider, and perhaps more appropriate, explication of the Occam's razor is proposed by Vapnik (2006). In a slightly modified form, to avoid mixing different terminologies, it can be put as follows: "Find an admissible decision rule with the smallest number of free parameters to explain the observed facts" (Vapnik 2006, p. 448). However, even in this format, the principle gives no guidance about how to choose an adequate functional form. For example, which of two functions, the power function $f(x) = ax^b$ or logarithmic one, $g(x) = b\log(x) + a$, both having just two parameters a and b, should be preferred as a summarization tool for graphs on Fig. 3.3?

Another set of advices, not incompatible with those above, relates to the so-called falsifiability principle by Popper (1902–1994), which can be expressed as follows: "Explain the facts by using such an admissible decision rule which is easiest to falsify" (Vapnik 2006, p. 451). In principle, to falsify a theory one needs to give an example contradicting to it. Falsifiability of a decision rule can be

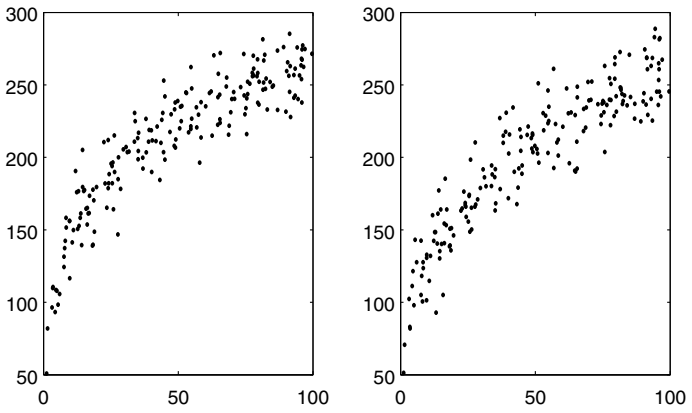

Fig. 3.3 A graph of one out of two functions, $f(x) = 65x^{0.3}$ and $g(x) = 50\log(x) + 30$, both with an added normal noise $N(0,15)$, is presented on each of the plots. Can the reader give an educated guess of which is which? (Answer: *f(x)* is on the right and *g(x)* on the left)

formulated in terms of the so-called *VC*-dimension, a measure of complexity of classes of decision rules: the smaller VC- dimension, the greater the falsifiability.

Let us explain the concept of VC-dimension for the case of a categorical target, so that a decision rule to be would be a classifier. However many categorical target features are specified, different combinations of target categories can be assigned different labels, so that a classifier is bound to predict a label. A set of classifiers Φ is said to shatter the training sample if for any possible assignment of the labels, a classifier exactly reproducing the labels can be found in Φ. Given a set of admissible classifiers Φ, the VC-dimension of a classifying problem is the maximum number of entities that can be shattered by classifiers from Φ. For example, 2D points have VC complexity 3 in the class of linear decision rules. Indeed, any three points, not lying on a line, can be shattered by a line; yet not all four-point sets can be shattered by lines, as shown on Fig. 3.4, the left and right parts, respectively.

The VC complexity is an important characteristic of a correlation problem especially within the probabilistic machine learning paradigm. Under the conventional conditions of the independent random sampling of the data, a reliable classifier "with probability $a\%$ will be $b\%$ accurate, where b depends not only on a, but also on the sample size and VC-dimension" (Vapnik 2006).

The problem of learning correlation in a data table can be stated, in general terms, as follows. Given N pairs (x_i, u_i), $i = 1, \ldots, N$, in which x_i are predictor/input p-dimensional vectors $x_i = (x_{i1},\ldots,x_{ip})$ and $u_i = (u_{i1},\ldots,u_{iq})$ are target/output q-dimensional vectors (usually $q = 1$), find a decision rule

$$\hat{u} = F(x) \tag{3.1}$$

such that the difference between computed $\hat{u}$ and observed u is minimal over a pre-specified class Φ of admissible rules F.

To specify a correlation learning problem one should specify assumptions regarding a number of constituents including:

(i) Type of target
Two types of target features are considered usually: quantitative and categorical. In the former case, Eq. (3.1) is usually referred to as regression; in the latter case, decision rule, and the learning problem is referred to as that of "classification" or "pattern recognition".

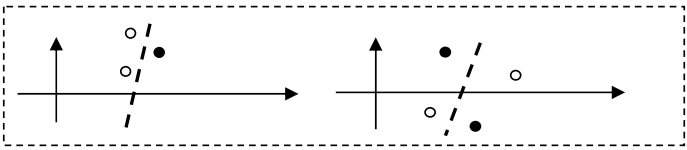

Fig. 3.4 Any two-part split of three points (not on one line) can be made by a line, but the presented case of four points on the right cannot be split by a line

(ii) Type of rule
A rule involves a postulated mathematical structure whose parameters are to be learnt from the data. The mathematical structures considered further on are:

- *linear* combination of features
- *neural network* mapping a set of input features into a set of target features
- *decision tree* built over a set of features
- *partition* of the entity set into a number of non-overlapping clusters

(iii) Criterion
Criterion of the quality of fitting depends on the framework in which the learning task is formulated. Most popular criteria are: maximum likelihood (in a probabilistic model of data generation), least-squares (data recovery approach) and relative errors. According to the least-squares criterion, the difference between u and $\hat{u}$ is measured with the average squared error,

$$E = \langle u - \hat{u}, u - \hat{u} \rangle / N = \langle u - F(x), u - F(x) \rangle / N \tag{3.2}$$

which is to be minimized over all admissible F.

(iv) Training protocol
The rule F is to be learnt from a training dataset. The way the data become available can be referred to as the learning protocol. Three popular training protocols are: batch, random and on-line. The batch mode is the case when all the training set is available and used at once, the other two refer to cases when data entities come one by one so that the learning goes incrementally. In the random protocol, the data are available at once, yet the learning process is organized incrementally by picking up entities randomly one-by-one, possibly many times each. In contrast, in an on-line protocol each entity comes from an external source and can be seen only once.

3.2 Two-D Linear Regression and Special Cases

3.2.1 Case of Two Features

Let us first focus on a most illustrative case when only two quantitative features are considered. Three most popular concepts are: scatter plot, correlation coefficient, and regression.

We consider them in turn by using two features from the Market towns dataset, Population Resident and Number of Primary Schools. The data are taken from Table 1.4 (see below an extract for four towns out of 45):

	PopRes (x)	PSchools (y)	(x,y)-point
Tavistock	10,222	5	(10,222.5)
Bodmin	12,553	5	(12,553.5)
Saltash	14,139	4	(14,139.4)
Brixham	15,865	7	(15,865.7)

Scatter plot is a presentation of entities as 2D points in the plane of two pre-specified features. On the left-hand side of Fig. 3.5, a scatter-plot of Market town features PopRes (Axis x) and PSchools (Axis y) is presented.

One can think that these two features are approximately related by a linear equation $y = ax + b$ where a and b are some constant coefficients, referred to as the slope and intercept, respectively, because the number of schools should be related to the number of children which is related to the number of residents. This equation is referred to as the linear regression of y over x. Obviously, most relations are not necessarily that simple because they also depend on other factors such as school sizes, population's age, etc. It would be a miracle if one equation fitted well all 45 towns. The possible inconsistencies in the equation can be modeled as additive errors, or residuals. The slope a and intercept b are taken in such a way that the inconsistencies of the equation on the 45 towns are minimized.

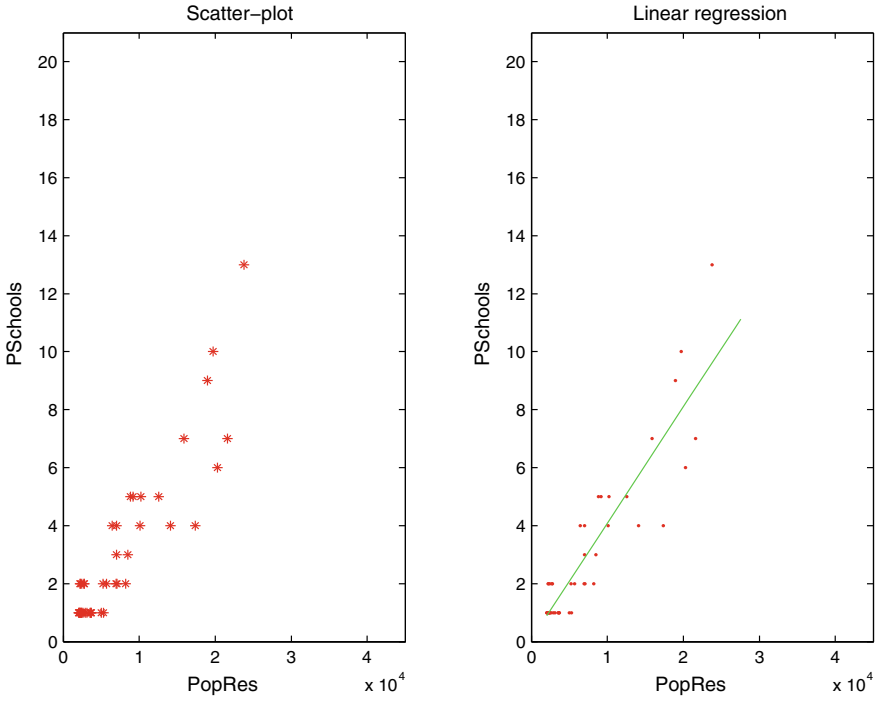

Fig. 3.5 Scatter plot of PopRes versus PSchools in Market town data. The right hand graph includes a regression line of PSchools over PopRes

When a linear regression equation is fitted, its validity should be checked. A valid equation can be used for both (i) prediction and (ii) description.

The term "regression" relates to an episode in the long struggle by Francis Galton for the recognition of his obsession with "inherited talent". He thought, in about 1888, like this: "Currently, I cannot measure the talent—why cannot I take a feature, that I can measure, say, the height? The son's height should be related to the father's height, adjusted by that of the mother, of course." This thought was possibly supported by the idea, rather relevant at that time, that the human's height may have an evolutionary dimension so that, in the UK, the taller people might have had better chances of survival. The relation appeared to exist. However, unexpectedly, it turned out to be rather counter-intuitive. The taller father's son was, on average, not as tall as his father, whereas, in contrast, the shorter fathers' sons were relatively taller than their fathers. Galton considered that as a phenomenon of regress to the mediocrity. And it took some time, to figure out that the regress did not contradict the law of natural selection established by his cousin, Charles Darwin.[1]

The Galton-Pearson theory of linear regression involves a useful and very popular parameter, the correlation coefficient, that shows the extent of linearity in the relation between two features. Its square, referred to as the determinacy coefficient, can be used for a quick check of the validity of the regression: it shows the proportion of the variance of y that is taken into account by the regression. The correlation coefficient between the two features, PopRes and PSchools, is 0.909. The correlation coefficient, in general, ranges between -1 and 1, and a value close to 1 or -1 indicates a high extent of the linear relation between the features. In physics or chemistry, a high value of the correlation coefficient is rather usual; in social sciences, rather not—that is, the current features are highly related indeed.

Most other features in Market town data—such as the numbers of Post offices or Doctors—are also highly related to Pop feature, but not the number of Farmers markets. This latter feature appears to be binary here: a town either has a farmers market or not. The low value of the correlation coefficient, just below 0.15, shows that the size of the town does not much matter in this part of the world: a farmers market is as likely in a small town as it is in a larger town.

A low or even zero value of the correlation coefficient does not necessarily mean "no relation at all", but rather just "no *linear* relation". A zero correlation coefficient may hide a different type of functional relation, as shown on Fig. 3.6, which presents three different cases of the zero correlation. Only one of these, that on the left, case is genuine—there is no relation between x and y according to the picture indeed. Each of the other two cases relates to a rather high association between x and y. Specifically, the figure in the middle refers to a quadratic dependence and the figure on the right, to a split between two subsamples of highly linear but inverse relations.

[1]Both, Francis Galton and Charles Darwin, were grandsons of a celebrated medical doctor and philosopher, Erasmus Darwin.

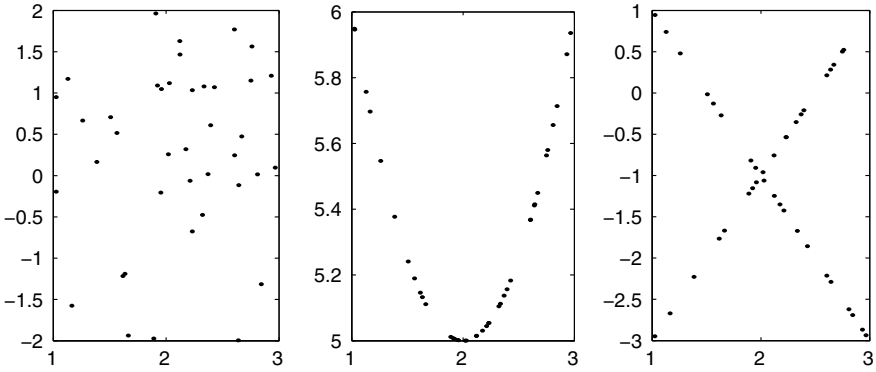

Fig. 3.6 Three scatter-plots corresponding to zero or almost zero correlation coefficient ρ; the case on the left: no relation between x and y; the case in the middle: a non-random quadratic relation $y = (x - 2)^2 + 5$; the case on the right: two symmetric linear relations, $y = 2x - 5$ and $y = -2x + 3$, each holding at a half of the entities

Then the regression equation, estimated according to formulas (3.6)–(3.8) in Sect. 3.2.3, is this:

$$\text{PSchool} = 0.401 * \text{PopRes} + 0.072 \tag{3.3}$$

where Population resident (PopRes) is expressed in thousands to make the slope the thousand times greater than it would be if population is expressed in the absolute numbers. The slope expresses how much target changes when the input changes by 1. Because the target's values are integers, the value of slope can be rephrased as follows: the growth of population in a town by 2.5 thousand would lead, on average, to building one more primary school.

3.2.2 Validity of the Regression

A regression function built over a data set should be validated. Three types of validity checks can be considered:

(a) The proportion of the variance of target variable taken into account by the regression, the determinacy coefficient: the greater the determinacy the better the fit.
(b) The confidence intervals of regression parameters—their ranges can give an idea of how stable the regression is.
(c) The direct testing of the accuracy of prediction both on data used for building the regression and data not used for that.

Worked Example 3.1. Determinacy Coefficient
Consider feature PSchools as target versus PopRes as input, in Market Data (Fig. 3.5). The correlation coefficient between them is 0.909. The determinacy coefficient, in the case of linear regression, is its square, that is, $0.909^2 = 0.826$, which shows that the linear dependence on PopRes decreases the variance of PSchools by 83.6%, a rather high value.

If the determinacy coefficient is not that high, still the hypothesis of linear relation may hold—depending on the distribution of residuals, that is, differences between the observed values of PSchool and those computed from PopRes according to Eq. (3.3). This distribution should be Gaussian or approximately Gaussian, so that the principle of maximum likelihood and formulas derived from that are appropriate. The distribution for the case under consideration is presented on Fig. 3.7. It is similar to a Gaussian distribution indeed, at the 5 bin histogram. The histogram with 10 bins is less so because it is somewhat dented—probably the sample is too small for this level of granularity: on average, only 4–5 entities fall in each of the bins.

A more straightforward validity test can be performed without any statistic theory at all—by purely computational means using the so-called bootstrapping which is a procedure for obtaining a multitude of random estimates of the parameters of interest by using random samples from the dataset as illustrated in the Worked Example 3.2.

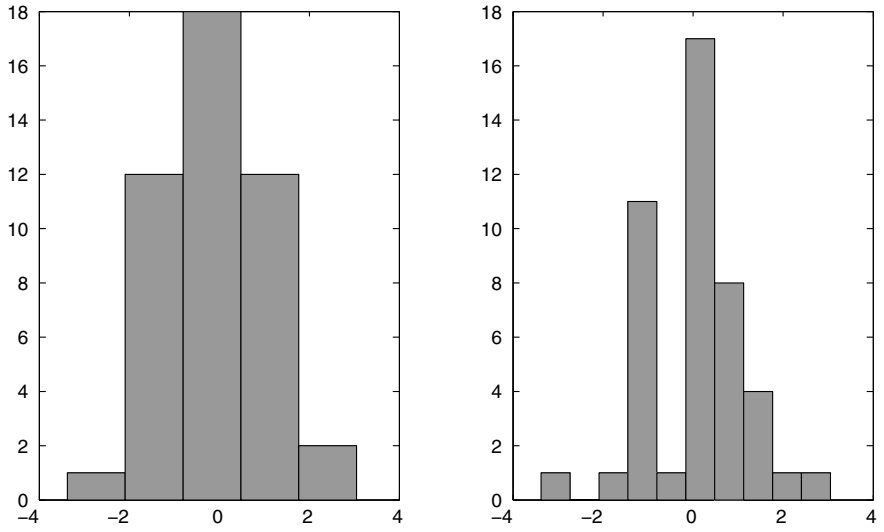

Fig. 3.7 Histograms of the residuals, the differences between values of PSchool as observed and those computed from Pop by using Eq. (3.1), with 5 bins (on the left) and 10 bins (on the right). The dents in the finer histogram can be attributed to the fact that the sample of 45 instances is too small to use 10 bins

Worked Example 3.2. Bootstrap Validity Testing
Consider the linear regression of PSchools over PopRes in Eq. (3.3) in the previous section. How stable are its slope and intercept regarding change of the sample? This can be tested by using an approach referred to as the bootstrap. One bootstrap trial involves three stages:

1. Randomly choose, with replacement, as many entities as there are in the sample—45 in this case. Here is the sequence of indices of the entities randomly drawn with replacement while writing this text: r = {26,17,36,11,29,39,32, 25,27,26, 29,4,4,33,10,1,5,45,17,16, 13,5, 42,43,28, 26,35,2,37,44,6,39,33, 21,15, 11,33,1,44,30,26,25,5,37,24}. Some indices made it into the sample more than once, most notably 26—four times, whereas many others did not make it into the sample at all—altogether, 16 objects such as 3,7,8 are absent from the sample. The proportion of the absent indices is 16/45 = 0.356, which is rather close to the theoretic estimate 1/e = 0.3679 derived in Sect. 2.2.3.3, p. 102.
2. Take "resampled" versions of PopRes and PSchools as their values on the elements drawn on step 1.
3. Find values of the slope and intercept for the resampled PopRes and PSchools and store them.

The MatLab computation steps are similar to those in Sect. 2.2.3.3. After 400 trials the stored slopes and intercepts form distributions presented as 20 bin histograms on Fig. 3.8 a and b, respectively. After 4000 trials, the respective histograms are c and d. One can easily see the smoothing effect of the increased number of trials on the histogram shapes—at 4000 trials they do look Gaussian.

The bootstrapping trials give a diversity needed for estimating the average values of the slope and intercept. Moreover, one can draw confidence boundaries for the values.

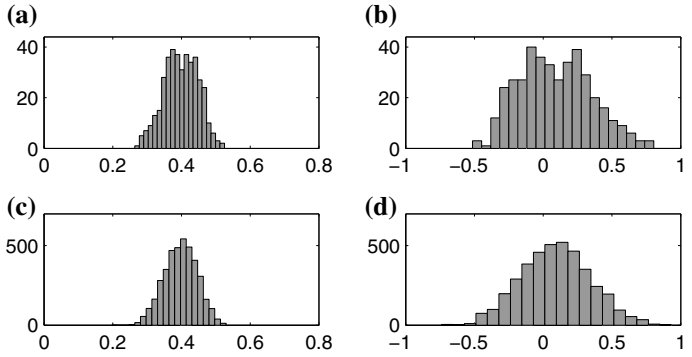

Fig. 3.8 Histograms of the distributions of the slope, on the left, and intercept, on the right, found at 400 (on top) and 4000 (bottom) bootstrapping trials on PopRes, expressed in thousands, and PSchool features in Market town data

Table 3.1 Parameters of the linear regression of PopRes over PSchool found on the original set, as well as on the bootstrap 400 and 4000 trials

Regression parameters	Set	400 trials			4000 trials		
		Mean	2.5%	97.5%	Mean	2.5%	97.5%
Slope	0.401	0.399	0.296	0.486	0.398	0.303	0.488
Intercept	0.072	0.089	−0.343	0.623	0.092	−0.400	0.594

The latter involves the average values, as well as the lower and upper 2.5% quantiles

How can one obtain, say, 95% confidence boundaries? According to the non-pivotal method, lower and upper 2.5% quantiles are cut out from the distribution in a symmetric way: 95% of the observations fall between the quantiles. For the case of 400 trials, 2.5% equals 10, so that the lower quantile corresponds to 11th and the upper quantile to 390th elements in the sorted set of values. For the case of 4000 trials, 2.5% equals 100: the quantiles correspond to 101st and 3900th elements of the sorted sets. They are shown in Table 3.1 at both of the cases, 400 and 4000 trials. One can see that these provide consistent and rather tight boundaries for the slope: it is between 0.303 and 0.488 in 95% of all trials, according to the 4000-trial data, and more or less the same at the 400-trial data. The values of intercept are distributed with a greater dispersion and provide for a worsened accuracy. Symmetric 95% confidence intervals for the intercept are [−0.343,0.623] at 400 trials and [−0.400,0.594] at 4000 trials.

How a pivotal bootstrapping rule can be applied here? This would provide more stable evaluations than empirical distributions. The standard deviations of the slope and intercept are 0.0493 and 0.2606, respectively, at 400 bootstrapping trials; they are somewhat smaller, 0.0477 and 0.2529, at the 4000 trials. Can one derive from this a symmetric 95% confidence interval for the slope or intercept? Tip: in a Gaussian distribution, 95% of all values fall within interval mean ±1.96 * std. This is the so-called pivotal bootstrapping method.

Q.3.1. Can you give an estimate of the level of variance of the differences between PSchool observed and computed values?

A final validity test of the regression equation is probably the toughest one—by the prediction error (see Worked Example 3.3).

Worked Example 3.3. Prediction Error of the Regression Equation
Compare the observed values of PSchool with those computed through PopRes according to Eq. (3.3). Table 3.2 presents a few examples taken from both ends of the sorted PopRes feature.

On average, the predictions are close, but, in some cases, are less so. One can easily estimate the relative error, which is [(1 − 0.89)/1] * 100 = 11% on the first element, [(2 − 0.97)/2] * 100 = 51.5% on the second element, etc. The average relative error of Eq. (3.3) on the set of all 45 towns is equal to 30.7%. Can it be made smaller? On the first glance, no, it cannot, because Eq. (3.1) minimizes the error.

Table 3.2 Observed numbers of primary schools versus those predicted from the population resident data on some market towns

PS obs.	PS comp.	Pop	PS obse.	PS comp.	Pop
1	0.89	2040	2	3.35	5676
2	0.97	2230	2	3.9	7044
2	1.06	2452	4	3.12	10,092
2	1.19	2786	7	6.44	15,865
1	1.54	3660	4	7.05	17,390

But, the error minimized by Eq. (3.1) is the average quadratic error, not the relative error under consideration. The two errors do differ, and Eq. (3.1) is not necessarily optimal with regard to the relative error.

The classical optimization theory has virtually nothing to propose for the minimization of the relative error—this criterion is neither linear, nor quadratic, nor convex. Instead, the evolutionary optimization approach can be applied to the task. This approach uses a population of solutions randomly evolving, iteration after iteration, in the search for better solutions as explained in Project 3.2. Applying the algorithm from that project to minimize the criterion of relative error, one can find a different solution, in fact, a set of solutions each leading to the average relative error of 26.4%, a reduction of 3.3 points, one seventh of the relative error of the Eq. (3.3).

The new solution is PSchool = 0.28 * PopRes + 0.33 expressing a smaller rate of increase in school numbers at the growth of population.

3.2.3 *Fitting the Equation of Linear Regression*

Let us derive parameters of linear regression. Given target feature y and predictor x at N entities (x_1,y_1), (x_2,y_2),..., (x_N,y_N), we are interested at finding a linear equation relating them so that

$$y = ax + b \tag{3.4}$$

The exact fit can occur only if all pairs (x_i,y_i) belong to the same straight line on (x,y)-plane, which is rather unlikely on real-world data. Therefore, Eq. (3.4) will have an error at each pair (x_i,y_i) so that the equation should be rewritten as

$$y_i = ax_i + b + e_i \quad (i = 1, 2, \ldots, N) \tag{3.4'}$$

where e_i are referred to as errors or residuals. The problem is of determining the two parameters, a and b, in such a way that the residuals are least-squares minimized, that is, the average square error

$$L(a,b) = \Sigma_i e_i^2/N = \Sigma_i (y_i - ax_i - b)^2/N, \tag{3.5}$$

reaches its minimum over all possible a and b, given x_i and y_i (i = 1, 2, ...,N). This minimization problem is easy to solve with the elementary calculus tools.

Indeed $L(a, b)$ is a "bottom down" parabolic function of a and b, so that its minimum corresponds to the point at which both partial derivatives of $L(a, b)$ are zero (the first-order optimality condition):

$$\partial L/\partial a = 0 \quad \text{and} \quad \partial L/\partial b = 0.$$

Leaving the task of actually finding the derivatives to the reader as an exercise, let us focus on the unique solution to the first-order optimality equations defined by the following formulas (3.6), for a, and (3.8), for b:

$$a = \rho\sigma(y)/\sigma(x) \tag{3.6}$$

where

$$\rho = \left[\Sigma_i (x_i - m_x)(y_i - m_y)\right]/[N\sigma(x)\sigma(y)] \tag{3.7}$$

is the so-called correlation coefficient and m_x, m_y are means of x_i, y_i, respectively;

$$b = m_y - am_x \tag{3.8}$$

By putting these optimal a and b into (3.5), one can express the minimum criterion value as

$$L_m(a,b) = \sigma^2(y)(1 - \rho^2) \tag{3.9}$$

The Eq. (3.4) is referred to as the linear regression of y over x, index ρ in (3.6) and (3.7) as the correlation coefficient, its square ρ^2 in (3.9) as the determinacy coefficient, and the minimum criterion value L_m in (3.9) is referred to as the unexplained, or residual, variance.

3.2.4 Correlation Coefficient and Its Properties

The meaning of the coefficients of correlation and determinacy, in the data recovery framework of data analysis, is provided by Eqs. (3.6)–(3.9). Here are some formulations.

Property 1 Determinacy coefficient ρ^2 shows the rate of decrease of the variance of y after its linear relation to x has been taken into account by the regression (follows from (3.9)).

Property 2 Correlation coefficient ρ ranges between -1 and 1, because ρ^2 is between 0 and 1, as follows from the fact that value L_m in (3.9) cannot be negative because the items in its expression (3.5) are all squares. The closer ρ to either 1 or -1, the smaller are the residuals in the regression equation. For example, $\rho = 0.9$ implies that y's unexplained variance L_m is $1 - \rho^2 = 19\%$ of the original value.

Property 3 The slope a is proportional to ρ according to (3.6); a is positive or negative depending on the sign of ρ. If $\rho = 0$, the slope is 0: in this case, y and x are referred to as not correlated.

Property 4 The correlation coefficient ρ does not change under shifting and rescaling of x and/or y, which can be seen from Eq. (3.7). Its formula (3.7) becomes especially simple if the so-called z-scoring has been applied to standardize both x and y.

To perform z-scoring over a feature, its mean m is subtracted from all the values and the results are divided by the standard deviation σ:

$$x_i' = (x_i - m_x)/\sigma(x) \quad \text{and} \quad y_i' = (y_i - m_y)/\sigma(y), \quad i = 1, 2, \ldots, N$$

Using the z-score standardization, formula (3.7) can be rewritten as

$$\rho = \Sigma_i x_i' y_i'/N = \langle x', y' \rangle / N \tag{3.7'}$$

where $\langle x', y' \rangle$ denotes the inner product of vectors $x' = (x_i')$ and $y' = (y_i')$.

The next property refers to one of the fundamental discoveries by K. Pearson, interpretation of the correlation coefficient in terms of the bivariate Gaussian distribution. A generic formula for the density function of this distribution, in the case in which the features have been pre-processed by using z-score standardization described above, is

$$f(u, \Sigma) = C \exp\{-u^T \Sigma^{-1} u/2\} \tag{3.10}$$

where $u = (x, y)$ is a two-dimensional vector of two variables, x and y, under consideration and Σ is the so-called correlation matrix

$$\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

In formula (3.10), ρ is a parameter with a very clear geometric meaning. Consider a set of points $u = (x,y)$ on (x,y)–plane making function $f(u, \Sigma)$ in (3.10) equal to a pre-specified constant. Such a set makes the values of $u^T \Sigma^{-1} u$ constant too. That means that a constant density set of points $u = (x, y)$ must satisfy equation $x^2 - 2\rho xy + y^2 = \text{const}$. This equation defines a well-known quadratic curve, the ellipsis. At $\rho = 0$ the equation becomes that of a circle, $x^2 + y^2 = const$, and the greater the difference between ρ and 0, the more skewed is the ellipsis, so that at $\rho = \pm 1$ the ellipsis becomes a bisector line $y = \pm x + b$ because the left part of the

equation makes a full square, in this case, $x^2 \pm 2xy + y^2 = const$, that is, $(y \pm x)^2 = const$. The size of the ellipsis is proportional to the constant: the greater the constant the greater the size.

Property 5 The correlation coefficient (3.7) is a sample based estimate of the parameter ρ in the Gaussian density function (3.10) under the conventional assumption that the sample points (y_i, x_i) are drawn from a Gaussian population randomly and independently.

This striking fact is behind a longstanding controversy. Some say that the usage of the correlation coefficient is justified only when the sample is taken from a Gaussian distribution, because the coefficient has a clear-cut meaning only in this model. This logic seems somewhat overly restrictive. True, the usage of the coefficient for estimating the density function is justified only when the function is Gaussian. However, when trying to linearly represent one variable through the other, the coefficient has a very different meaning in the approximation context, which has nothing to do with the Gaussian distribution, as expressed above with Eqs. (3.6)–(3.9).

3.2.5 *Linearization of Non-linear Regression*

Non-linear dependencies also can be fit by using the same criterion of minimizing the square error. Consider a popular case of exponential regression, that is, representing correlation between target y and predictor x as $y = ae^{bx}$ where a and b are unknown constants and e the base of natural logarithm. Given some a and b, the average square error is calculated as

$$E = \left([y_1 - a\exp(bx_1)]^2 + \ldots + [y_N - a\exp(bx_N)]^2\right)/N = \Sigma_i[y_i - a\exp(bx_i)]^2/N \tag{3.11}$$

There is no method that would straightforwardly lead to a globally optimal solution of the problem of minimization of E in (3.11) because it is too complex function of the unknown values. This is why conventionally the exponential regression is fit by what should be referred to as its linearization: transforming the original problem to that of linear regression.

Indeed, let us take the logarithm of both parts of the equation that we want to fit, $y = ae^{bx}$. The resulting equation is $\ln(y) = \ln(a) + bx$. This equation has the format of linear equation, $z = \alpha x + \beta$, where $z = \ln(y)$, $\alpha = b$ and $\beta = \ln(a)$. This leads to the following idea. Let us take the target be $z = \ln(y)$ with its values $z_i = \ln(y_i)$. By fitting the linear regression equation with data x_i and z_i, one finds optimal α and β, so that the original exponential parameters are found as $a = \exp(\beta)$ and $b = \alpha$. These values do not necessarily minimize (3.11), but the hope is that they are close to the optimum anyway. Unfortunately, this may be very wrong sometimes as the material in Project 3.2. clearly demonstrates.

Q.3.2. Find the derivatives of L over a and b and solve the first-order optimality conditions.
Q.3.3. Derive the optimal value of L in (3.9) for the optimal a and b.
Q.3.4. Prove or find a proof in the literature that any linear equation $y = ax + b$ corresponds to a straight line on Cartesian xy plane, for which a is the slope and b intercept.
Q.3.5. Find the inverse matrix Σ^{-1} for $\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$.
A. $\Sigma^{-1} = \begin{pmatrix} 1 & -\rho \\ -\rho & 1 \end{pmatrix} / (1 - \rho^2)$.

3.2.6 *Linear Regression: Computation*

Regression is a technique for representing the correlation between x and y as a linear function (that is, a straight line on the plot), y = *slope* * x + *intercept* where *slope* and *intercept* are constants, the former expressing the change in y when x is added by 1 and the latter the level of y at $x = 0$. The best possible values of slope and intercept (that is, those minimizing the average square difference between real y's and those found as *slope* * x + *intercept*) are expressed in MatLab, according to formulas (3.6) and (3.8), as follows:

```
≫slope=corr(x,y)*std(y)/std(x);
≫intercept=mean(y) - slope*mean(x);
```

Here corr(x,y) is the MatLab command for computing Pearson correlation coefficient between x and y (3.7). There is another MatLab operation "corrcoef" which leads to an estimate of the matrix Σ above.

Project 3.1. 2D Analysis, Linear Regression and Bootstrapping
Let us take the Students data table as a 100 × 8 array a in MatLab, pick any two features of interest and plot entities as points on the Cartesian plane formed by the features. For instance, take Age as x and Computational Intelligence mark as y:

```
≫x=a(:,4); % Age is 4-th column of array "a"
≫y=a(:,8); % CI score is in 8-th column of "a"
```

Then student 1 (first row) will be presented by point with coordinates x = 28 and y = 90 corresponding to the student's age and CI mark, respectively. To plot them all, use command:

≫plot(x,y,‘k’)
% k refers to black colour, “.” dot graphics; ‘mp’ stands for magenta pentagram;
% see others by using “help plot”

Unfortunately, this gives a very tight presentation: some points are on the borders of the drawing. To make the borders stretched out, one needs to change the axis, for example, as follows:

≫d=axis; axis(1.2*d-10);

This transformation is presented on the right part of Fig. 3.9. To make both plots presented on the same figure, use “subplot” command of MatLab:

≫subplot(1,2,1)
≫plot(x,y,′k.′);
≫subplot(1,2,2)
≫plot(x,y,′k.′);
≫d=axis; axis(1.2*d-10);

Whichever presentation is taken, no regularity can be seen on Fig. 3.9 at all. Let’s try then whether anything better can be seen for different occupations. To do this, one needs to handle entity sets for each occupation separately:

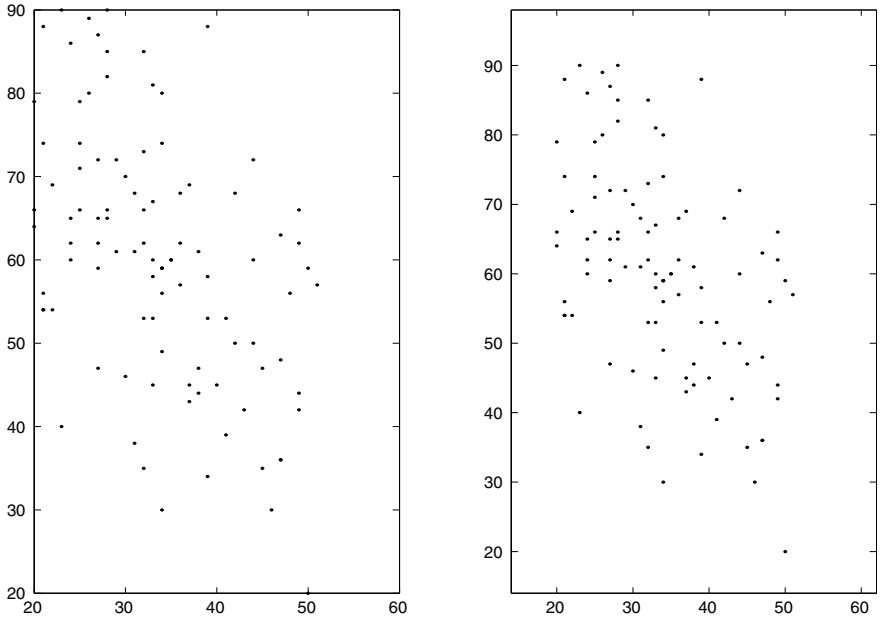

Fig. 3.9 Scatter plot of features “Age” and “CI score”; the display on the right is a rescaled version of that on the left

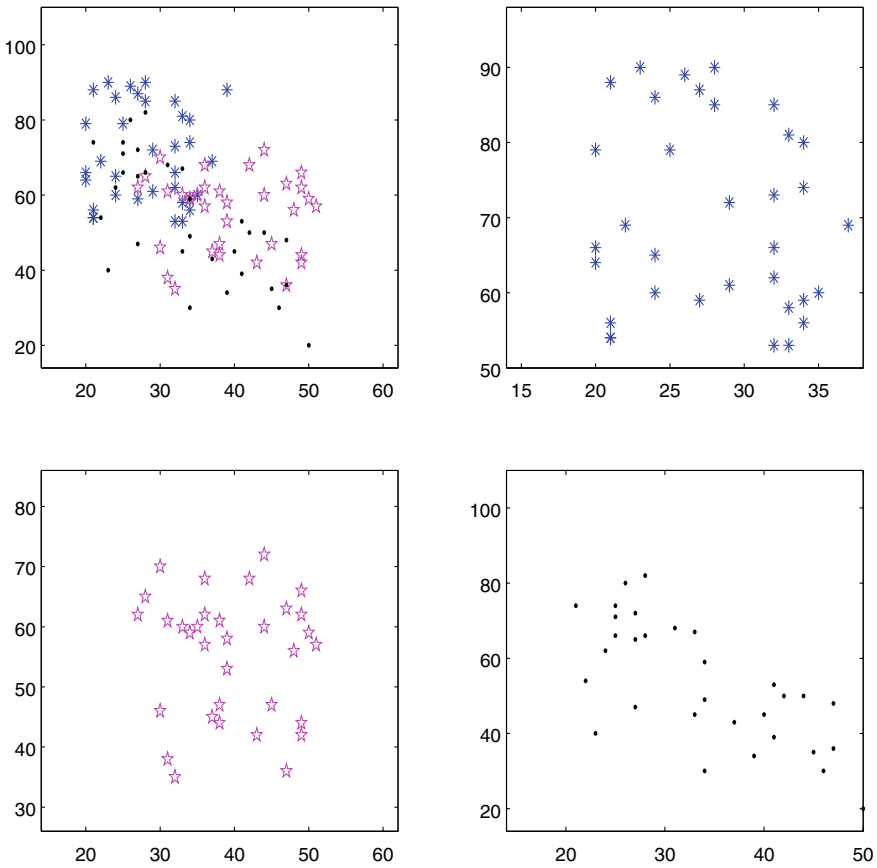

Fig. 3.10 Joint and individual displays of the scatter-plots at the occupation categories (IT star, BA pentagrams, AN dots)

≫o1=find(a(:,1)==1); % set of indices for IT
≫o2=find(a(:,2)==1); % set of indices for BA
≫o3=find(a(:,3)==1); % set of indices for AN
≫x1=x(o1);y1=y(o1); % the features x and y at IT students
≫x2=x(o2);y2=y(o2); % the features at BA students
≫x3=x(o3);y3=y(o3); % the features at AN students

Now we are in a position to put, first, all the three together, and then each of these three separately (again with the command "subplot", but this time with four windows organized in a two-by-two format, see Fig. 3.10).

≫subplot(2,2,1); plot(x1,y1, ′*b′,x2,y2,′pm′,x3,y3,′.k′);% all three
≫d=axis; axis(1.2*d-10);
≫subplot(2,2,2); plot(x1,y1, ′*b′); % IT plotted with blue stars

≫d=axis; axis(1.2*d-10);
≫subplot(2,2,3); plot(x2,y2,′pm′); %BA plotted with magenta pentagrams
≫d=axis; axis(1.2*d-10);
≫subplot(2,2,4); plot(x3,y3,′.k′); % AN plotted with black dots
≫d=axis; axis(1.2*d-10);

Of the three occupation groups, some potential relation can be seen only in the AN group: it is likely that "the greater the age the lower the mark" regularity holds in this group (black dots in the Fig. 3.10's bottom right). To check this, let us utilize the linear regression.

Linear regression equation, $y = slope * x + intercept$, is estimated by using MatLab, according to formulas (3.4)–(3.6), as follows:

≫cc=corrcoef(x3,y3);rho=c(1,2);% producing rho = -0.7082
≫slope=rho*std(y3)/std(x3); % this produces slope=−1.33;
≫intercept=mean(y3) - slope*mean(x3); % this produces intercept=98.2;

Since we are interested in group AN only, we apply these commands at AN-related values x3 and y3 to produce the linear regression as $y3 = 98.2 - 1.33 * x3$. The slope value suggests that every year added to the age, in general decreases the mark by 1.33, so that aging by 3 years would lead to the loss of 4 mark points. Obviously, care should be taken to draw realistic conclusions.

Altogether, the regression equation explains *rho*^2 = 0.50 = 50% of the total variance of y3—not too much, as is usual in social and human sciences.

Let us take a look at the reliability of the regression equation with bootstrapping, the popular computational experiment technique for validating data analysis results (see Sect. 2.2.3.3).

Bootstrapping is based on a pre-specified number of random trials, for instance, 5000. Each trial consists of the following steps:

(i) randomly selecting an entity N times, with replacement, so that the same entity can be selected several times whereas some other entities may be never selected in a trial. (As shown in Sect. 2.2.3.3, on average only 63% entities get selected into the sample.) A sample consists of N entities because this is the number of entities in the set under consideration. In our case, N = 31. One can use the following MatLab command:

≫N=31;ra=ceil(N*rand(N,1));
% rand(N,1) produces a column of N random real numbers, between 0 and 1 each.
% Multiplying this by N stretches them to (0,N) interval; the operation ceil rounds the numbers up to integers.

(ii) the sample ra is assigned with their data values according to the original data table:

≫xt=xx(ra);yt=yy(ra);
% here xx and yy represent the predictor and target, respectively;

```
% they are x3 and y3, respectively, which can be taken into account with
assignments
% xx=x3; and yy=y3.
```

so that coinciding entities get identical feature values.

(iii) a data analysis method under consideration, currently "linear regression", that basically computes the rho, the slope and the intercept, applies to this data sample to produce the trial result.

To do a number of trials (5000, in this case), one should run (i)–(iii) in a loop:

```
≫for k=1:5000; ra=ceil(N*rand(N,1));

    xt=xx(ra);yt=yy(ra);
    cc=corrcoef(xt,yt);
    rh(k)=cc(1,2);
    sl(k)=rh(k)*std(yt)/std(xt); inte(k)=mean(yt)-sl(k)*mean(xt);

end
% the results are 5000-strong set of columns rh (correlations), sl (slopes)
% and inte (intercepts)
```

Now we can check the mean and standard deviation of the obtained distributions. Commands

```
≫mean(sl); std(sl)
```

produce values: −1.33 and 0.23 for the mean and standard deviation, respectively. That means that the original value of slope = −1.33 is confirmed with the bootstrapping, but now we have obtained its standard deviation, 0.23, as well. Similarly mean/std values for the intercept and rho are computed. They are, respectively, 98.2/9.0 and −0.704/0.095.

We can plot the 5000 values found as 30-bin histograms (see Fig. 3.11):

```
≫subplot(1,2,1); hist(sl,30)
≫subplot(1,2,2); hist(in,30)
```

The command subplot (1,2,1) creates a pane with one row consisting of two windows for plots and puts the follow-up plot into the first window (that on the left). Command subplot (1,2,2) moves the action into the second window which is on the right of Fig. 3.11.

To derive the 95% confidence boundaries for the slope, intercept and correlation coefficient, one may use both pivotal and non-pivotal methods.

The pivotal method uses the hypothesis that the bootstrap sample is indeed a random sample from a Gaussian distribution. Parameters of this distribution for slope are determined with the following commands:

```
≫msl=mean(sl); ssl=std(sl);
```

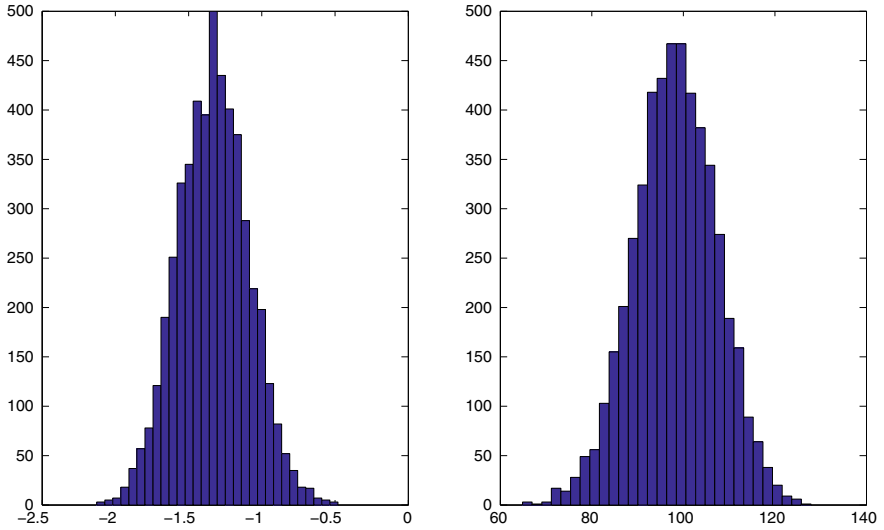

Fig. 3.11 30-bin histograms of the slope (left) and intercept (right) after 5000 bootstrapping trials

Since 95% of the Gaussian distribution fall within interval of ±1.96*std, the 95% confidence boundaries are derived, for the slope, as follows:

≫lbsl=msl – 1.96*ssl; rbsl=msl+1.96*ssl

The non-pivotal estimates require no such a hypothesis and are based on the bootstrap distribution as is. One just sorts all the values and takes 2.5% quantiles on both extremes of the range:

≫ssl=sort(sl); lbn=ssl(126);rbn=ssl(4875);

Indeed, we need to cut out 5% items from the sample, to make a 95% confidence interval. Since 5% of 5000 is 250, conventionally divided in two halves, this requires cutting off first 125 observations as well as the last 125 observations of the presorted list of the bootstrap values, which brings us to ssl(126) and ssl(4875) as the non-pivotal boundaries for the slope value.

All these estimates are presented in Table 3.3. The pivotal and non-pivotal estimates do not fall too far apart. Either can be taken as parameters of the boundary regressions.

Table 3.3 Parameters of the bootstrap distributions, as well as pivotal and non-pivotal boundaries

	Mean	St. dev.	Pivotal boundaries		Non-pivotal boundaries	
			Left	Right	Left	Right
Slope	−1.337	0.241	−1.809	−0.865	−1.800	−0.850
Intercept	98.510	9.048	80.776	116.244	80.411	116.041
Corr. coef.	−0.707	0.094	−0.891	−0.523	−0.861	−0.493

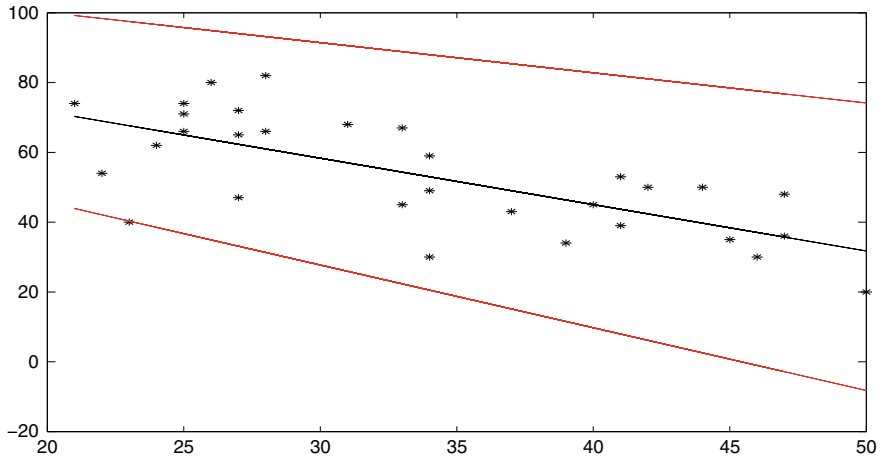

Fig. 3.12 Regression of CI score over Age (black line) within occupation category AN with boundaries covering 95% of potential biases due to sample fluctuations

This all can be visualized by, first, defining the three regression lines, the regular one and two corresponding to the lower and upper estimate boundaries, respectively, with

```
≫y3reg=slope*x3+intercept;
≫y3regleft=lbsl*x3+lbintercept;
≫y3regright=rbsl*x3+rbintercept;
```

and then plotting the four sets onto the same Fig. 3.12.

```
≫plot(x3,y3,'*k',x3,y3reg,'k',x3,y3regleft,'r',x3,y3regright,'r')
% x3,y3,'*k' presents student data as black stars; x3,y3reg,'k' presents the
% real regression line in black
% x3,y3regleft,'g' and x3,y3regright,'g' for boundary regressions in green
```

The lines on Fig. 3.12 show the boundaries of the regression line for 95% of trials.

Project 3.2. Non-linear and Linearized Regression: A Nature-Inspired Algorithm

In many domains the correlation between features is not necessarily linear. For example, in economics, processes related to the inflation over time are modeled by using the exponential function. A similar way of thinking applies to the processes of growth in biology. Variables describing climatic conditions obviously have a cyclic character; etc. The power law in social systems is nonlinear too.

Consider, for example, a power law function $y = ax^b$ where x is predictor and y target variables whereas a and b are unknown constant coefficients. Given the values of x_i and y_i on a number of observed entities $i = 1,\ldots, N$, the power law regression problem can be formulated as the problem of minimizing the summary

squared or absolute error over all possible pairs of coefficients a and b. There is no method that would straightforwardly lead to a globally optimal solution of the problem because minimizing a sum of many exponents is a complex problem. This is why conventionally the power law regression is fit by transforming it into a linear regression problem. Indeed, the equation of the power law regression, taken with no errors, is equivalent to the equation of linear regression with $\log(x)$ being predictor and $\log(y)$ the target: $\log(y) = b\log(x) + \log(a)$.This gives rise to the very popular strategy of linearization of the problem. First, transform x_i and y_i to $v_i = \log(x_i)$ and $z_i = \log(y_i)$ and fit the linear regression equation for given v_i and z_i; then convert the found coefficients into those of the original exponential function. This strategy seems especially suitable since the logarithm of a variable typically is much smoother so that the linear fit is better under the logarithm transformation.

There is one caveat, however: the fact that found coefficients are optimal in the linear regression problem does not necessarily imply that the converted exponents are necessarily optimal in the original problem. This we are going to explore in this project.

Nature-inspired optimization is a computational intelligence approach to minimize a non-linear function. Rather than look and polish a single solution to the optimization problem under consideration, this approach utilizes a population of solutions iteratively evolving from generation to generation, according to rules imitating a real-world evolutionary process and survival of the fittest. The rules typically include: (a) random changes from generation to generation such as "mutations" and "crossovers" in earlier, so-called "genetic", algorithms, and (b) policies for selecting and maintaining the best found solutions, the "elite". After a pre-specified number of iterations, the best solution among those observed in computations is reported as the outcome.

To start the evolutionary optimization process, one should first define a restricted area of admissible solutions so that no member of the population may leave the area. This warrants that the population will not explode by moving solutions to the infinity. There can be different ways for determining such an area. Let us follow this. Under the hypothesis of a power law relation $y = ab^x$, for any two entities i and j, the following equations should hold: $z_i = b * v_i + c$ and $z_j = b * v_j + c$ where $c = \log(a)$, $z_i = \log(y_i)$ and $v_i = \log(x_i)$. From these, b and c can be expressed as follows: $b = (z_i - z_j)/(v_i - v_j)$, $c = (v_i * z_j - v_j * z_i)/(v_i - v_j)$, which may lead to different values of b and c at different i and j. Denote bm and bM the minimum and the maximum of $(z_i - z_j)/(v_i - v_j)$, and cm and cM the minimum and maximum of $(v_i * z_j - v_j * z_i)/(v_i - v_j)$ over those i and j for which $v_i - v_j \neq 0$. One would expect that the admissible b and c should be within these boundaries, which means that the area of admissible solutions should be defined by the inequalities $(bm,cm) \leq (b,c) \leq (bM,cM)$. Since the optimal values of (b,c) should be around the averages of the ratios above, that is, lie deep inside the area between their maxima and minima, it helps to speed up the computation if one takes only those pairs (i,j) at which the values of v_i, v_j and z_i, z_j are not too close to 0 so that their logarithms are not that far

away from 0, and, similarly, the differences between them should be neither that small nor that high. This approach is implemented in MatLab code ddr.m in Appendix A.3.

For the step of producing the next generation, let us denote the population's $p \times 2$ array by f, at the current iteration, and by f', at the next iteration. The transition from f to f' is done in three steps. First, take the row of mean values within the columns of f and repeat it p times in a $p \times 2$ array mf. Then make a Gaussian random move:

$$\text{fn} = \text{f} + \text{randn}(\text{p}, 2). * \text{mf}/20$$

Here randn(p,2) is a p × 2 array of (pseudo) random numbers generated according to Gaussian distribution N(0,1) with 0 expectation and 1 variance. The symbol .* denotes the operation of multiplication of corresponding elements in matrices, so that (aij).*(bij) is a matrix whose (i,j)-th elements are products aij * bij. This random matrix is scaled down by $mf/20$ so that the move accounts for about 5% (one twentieth) of the average f values.

Since the move is to be restricted within the admissibility area, any a-element (first column of fn) which is greater than aM, is to be changed for aM, and any a-element smaller than am is to be changed for am. Similar trimming applies to b-elements. Denote the result by fr.

At the next step, take a $p \times 2$ array el whose rows are the same stored elite solution and arrive at the next generation f' by using the following "elite mix":

$$f' = 0.7fr + 0.3el$$

The elite mix moves all population members in the direction of the best solution found so far by 30%, which has been found work well in the examples of our interest.

This procedure is implemented in MatLab code nlr.m that relies on ddr.m at step 1 and a subroutine, delta, for evaluating the fitness (see A.4 in Appendix).

Consider now this experiment. Generate predictor x as a 50-long vector of random positive entries between 0 and 10, x = 10 * rand(1,50), and define $y = 2 * x^{1.07}$ with the normal additive noise 2 * $N(0,1)$ where 0 is the mean and 1 the variance, which is suppressed when overly negative, according to the Matlab code line

≫for ii=1:50;yy=2*x(ii)^1.07 +2*randn;y(ii)=max(yy,1.01);end;

When using the conventional linearized regression model by linearly mapping log(x) to log(y), to extract b and a (as the exponent of the found c) from this, the program llr.m implementing this approach produces $a = 3.0843$ and $b = 0.8011$ leading to the averaged squared error $y - ax^b$ equal to 3.41, so that the standard error is 3.10, about 20% of the mean y value, 10.1168. It is not only that the error is high, but also a wrong law is identified. The generated function y stretches x out ($b > 1$), whereas the found function stretches x in ($b < 1$).

Here are the results. Minimization of the averaged squared error $y - ax^b$ of the original model directly by using the code nlr.m, that implements the nature-inspired algorithm, the values are $a = 3.0293$ and $b = 1.0760$ leading to the average squared error of 0.0003 and the standard error of 0.0180. In contrast to the values found at the linearized scheme, the parameters a and b here are very close to those generated.

This obviously considerably outperforms the conventional procedure. Similar results can be found at different values of the noise variance.

Case-Study 3.1. Growth of Investment

Let us apply a similar approach to the following example involving variables x and y defined over a period of 20 time moments as presented in Table 3.4.

Variable x can be thought of as related to the time periods whereas y may represent the value of a fund. In fact, the components of x are numbers from 1 to 20 divided by 10, and y is obtained from them in MatLab according to formula $y = 2 * \exp(1.04 * x) + 0.6 * \text{randn}$ where randn is the normal (Gaussian) random variable with the mathematical expectation 0 and variance 1.

Let us, first, try a conventional approach of finding the average growth of the fund during all the period.

The average growth of the investment according to these data is conventionally expressed as the root 19, or power 1/19, of the ratio y_{20}/y_{01}, that is, 1.13. This estimates the average growth as 14% per period—which is by far greater than 4% in the data generating model.

Let us now try to make sense of the relation between x and y by applying the conventional linearization strategy to this data.

The strategy of linearization of the exponential equation outlined in Sect. 3.2.5 leads to values 1.1969 and 0.4986 for b and c, respectively, to produce $a = e^c = 1.6465$ and $b = 1.1969$ according to formulas there. As one can see, these differ from the original $a = 2$ and $b = 1.04$ by the order of 15–20%. The value of the squared error here is E = 13.90. See Fig. 3.13 representing the data.

Let us now apply the nature inspired approach to the original non-linear least-squares problem.

The program nlrm.m implementing the evolutionary approach described in Project 3.2 found a = 1.9908 and b = 1.0573. These are within 1–3% of the error from the original values a = 2 and b = 1.03. The summary squared error here is E = 7.45, which is by far smaller than that found with the linearization strategy.

The two found solutions can be represented on the scatter-plot graph, see Fig. 3.14. One can see that the linearized version has a much steeper exponent, which becomes visible at later periods.

Table 3.4 Data of investment y at time moments x from 0.10–3.00

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1.3	1.82	3.03	3.29	3.3	3.9	3.84	3.24	3.23	6.5
x	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	3
y	6.93	7.23	7.91	9.27	9.45	11.18	13.48	13.51	15.4	15.91

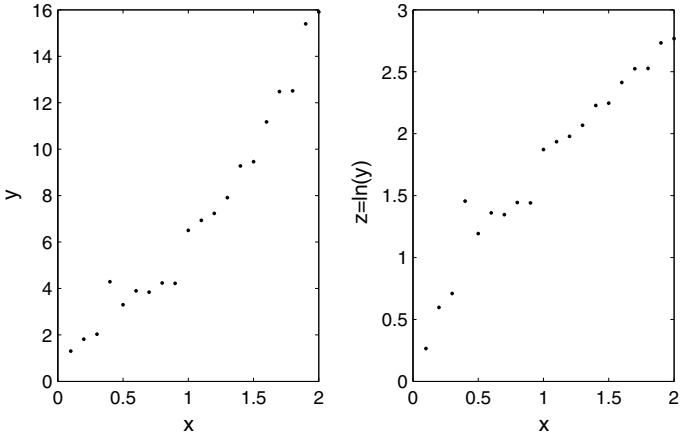

Fig. 3.13 Plot of the original pair (x,y) in which y is a noisy exponential function of x (on the left) and plot of the pair (x,z) in which $z = \ln(y)$. The plot on the right looks somewhat straighter indeed, though the correlation coefficients are rather similar, 0.970 for the plot on the left and 0.973 for the plot on the right

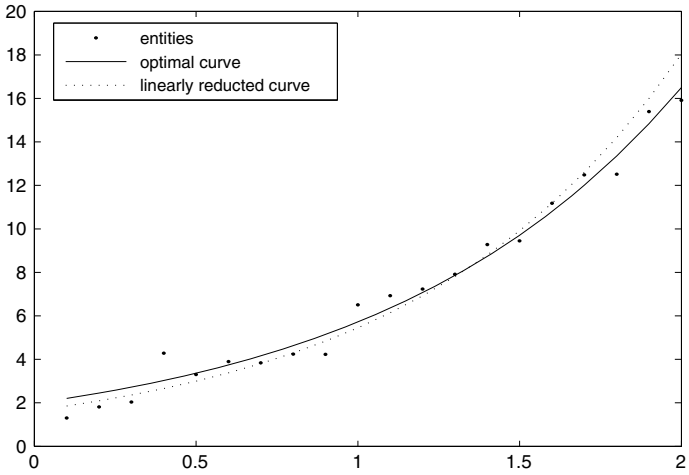

Fig. 3.14 Two fitting exponents are shown, with stars and dots, for the data in Case-Study 3.1

Q.3.6. Consider a binary feature defined on seven entities so that it is category A on the first three of them, and category B on the next four. Let us draw two dummy 1/0 variables, xA and xB, corresponding to each so that xA = 1 on the first three entities and xA = 0 on the rest, whereas xB = 0 on the first three entities and xB = 1 on the rest. What can be said of the correlation coefficient between xA and xB?

A. The correlation coefficient between xA and xB is -1 because xA + xB = 1 for all entities so that xA = −xB + 1.

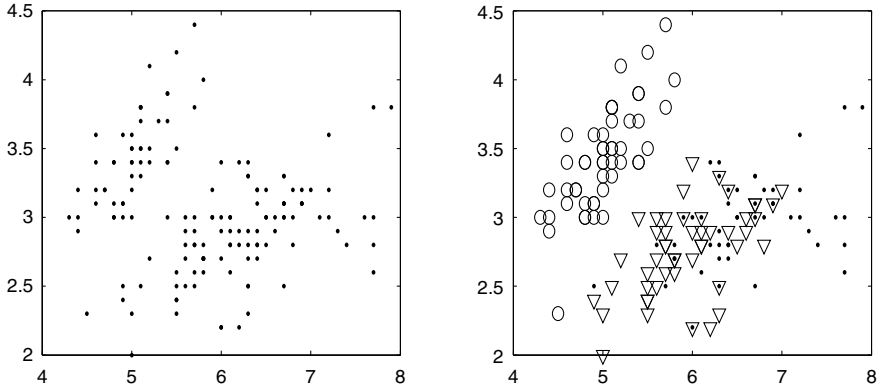

Fig. 3.15 Scatter plot of Sepal length and Sepal width from Iris data set (Table 1.1), as a whole on the left and taxon-wise on the right. Taxon 1 is presented by circles, taxon 2 by triangles, and taxon 3 by dots

Q.3.7. Extend the nature-inspired approach to the problem of fitting a linear regression with a nonconventional criterion such as the average relative error defined by formula $1/N \sum_{i=1}^{N} |e_i/y_i|$.

Case-Study 3.2. Correlation Between Iris Sepal Length and Width
Take x and y from the Iris set in Table 1.1 as the Sepal's length and width, respectively.

A scatter plot of x and y is presented on the left part of Fig. 3.15. This is a loose cloud of points which looks similar to that on the left part of Fig. 3.6, of no correlation. Indeed the correlation coefficient value here is not only very small, −0.12, but also negative, which is somewhat odd, because intuitively the features should be positively correlated as reflecting the size of the same flower.

To see a particular reason for the low, and negative, correlation, one should take into account that the sample is not homogeneous: the Iris set consists of 50 specimens of each of three different taxa. When the taxa are separated (see Fig. 3.15 on the right), the positive correlation is restored. The correlation coefficients are 0.74, 0.53 and 0.46 in taxon one, two and three, respectively. Here we see a nice example of the negative effect of the non-homogeneity of the sample on the data analysis results.

3.3 Multivariate Linear Regression

3.3.1 Formulation

Let us extend the notion of linear regression from the bivariate case to a multivariate case, when several features can be used as predictors for a target feature.

The problem of multivariate linear regression can be formulated as a particular case of the correlation learning problem with just one quantitative target variable u and linear admissible rules so that

$$u = w_1x_1 + w_2x_2 + \ldots + w_px_p + w_0$$

where w_0, w_1,..., w_p are unknown weights, parameters of the model.

For any entity $i = 1, 2, \ldots, N$, the rule-computed value of u

$$\hat{u}_i = w_1x_{i1} + w_2x_{i2} + \ldots + w_px_{ip} + w_0$$

differs from the observed one by $d_i = |\hat{u}_i - u_i|$, which may be zero—when the prediction is exact. To find w_1, w_2, …, w_p, w_0, one can minimize the summary square error

$$D^2 = \Sigma_i d_i^2 = \Sigma_i \left(u_i - w_1 * x_{i1} - w_2 * x_{i2} - \ldots - w_p * x_{ip} - w_0\right)^2 \tag{3.12}$$

over all possible parameter vectors $w = (w_0, w_1,\ldots,w_p)$.

To make the problem treatable in terms of linear operations, a fictitious feature x_0 is introduced such that all its values are 1: $x_{i0} = 1$ for all $i = 1, 2, \ldots, N$. Then criterion D^2 can be expressed as $D^2 = \Sigma_i(u_i - \langle w_i, x_i\rangle)^2$ using the inner products $\langle w, x_i\rangle$ where $w = (w_0, w_1,\ldots,w_p)$ and $x_i = (x_{i0}, x_{i1}, \ldots, x_{ip})$ are $(p + 1)$-dimensional vectors of which all x_i are known whereas w is not. From now on, the unity feature x_0 is assumed to be part of data matrix X in all correlation learning problems.

The criterion D^2 in (3.12) is but the squared Euclidean distance between the N-dimensional target feature column $u = (u_i)$ and vector $\hat{u} = Xw$ whose components are $\hat{u}i_i = \langle w, x_i\rangle$. Here X is $N\times(p + 1)$ matrix whose rows are x_i (augmented with the component $x_{i0} = 1$, thus being $(p + 1)$-dimensional) so that Xw is the matrix product of X and w. Vectors defined as Xw for all possible w's form $(p + 1)$-dimensional vector space, referred to as X-span.

Thus, the problem of minimization of (3.12) can be reformulated as follows: given a target vector u, find its projection $\hat{u}$ in the X-span space. The global solution to this problem is well-known: it is provided by a matrix P_X applied to u:

$$\hat{u} = P_Xu \tag{3.13}$$

where P_X is the so-called orthogonal projection operator, an $N \times N$ matrix, defined as:

$$P_X = X\left(X^TX\right)^{-1}X^T \tag{3.14}$$

so that

$$\hat{u} = X(X^TX)^{-1}X^Tu \text{ and } w = (X^TX)^{-1}X^Tu. \quad (3.15)$$

Matrix P_X projects every N-dimensional vector u to its nearest match in the $(p + 1)$-dimensional X-span space. The inverse $(X^TX)^{-1}$ does not exist if the rank of X, as it may happen, is less than the number of columns in X, $p + 1$, that is, if matrix X^TX is singular or, equivalently, the dimension of X-span is less than $p + 1$. In this case, the so-called pseudo-inverse matrix $(X^TX)^+$ can be used as well. This is not a big deal computationally: for example, in MatLab one just puts pinv(X^TX) instead of inv(X^TX).

The quality of approximation is evaluated by the minimum value D^2 in (3.12) averaged over the number of entities and related to the variance of the target variable. Its complement to 1, the determinacy coefficient, is defined by the equation

$$\rho^2 = 1 - D^2/(N\sigma^2(u)) \quad (3.16)$$

The determinacy coefficient shows the proportion of the variance of u explained by the linear regression. Its square root, ρ, is referred to as the coefficient of multiple correlation between u and $X = \{x_0, x_1, x_2, \ldots, x_p\}$.

3.3.2 Case Studies

Case-Study 3.3. Linear Regression for Market Town Data

Consider feature Post expressing the number of post offices in Market towns (Table 1.2 on p. 15) and try to relate it to other features in the table. It obviously relates to the population. For example, towns with population of 15,000 and greater are those and only those where the number of post offices is 5 or greater. This correlation, however, is not as good as to give us more guidance in predicting Post from the Population. For example, at the seven towns whose population is from 8000 to 10,000 any number of post offices from 1 to 4 may occur, according to the table. This could be attributed to effects of services such as a bank or hospital present at the towns. Let us specify a set of features in Table 1.2 that can be thought of as affecting the feature Post, to include in addition to Population some other features—PS—Primary schools, Do—General Practitioners, Hos—Hospitals, Ba—Banks, Sst—Superstores, and Pet—Petrol Stations; seven features altogether, taken as the set of input variables (predictors).

What we want is to establish a linear relation between this set and target feature Post. A linear relation is an equation representing Post as a weighted sum of input features plus a constant intercept; the weights can be any reals, not necessarily positive. If the relation is supported by the data, it can be used for various purposes such as analysis, prediction and planning.

Table 3.5 Weight coefficients of input features at Post Office as target variable for Market towns data

Feature	Weight
Pop_Res	0.0002
PSchools	0.1982
Doctors	0.2623
Hospitals	−0.2659
Banks	0.077
Superstores	0.0028
Petrol	−0.3894
Intercept	0.5784

In the example of seven Market town features used for linearly relating them to Post Office feature, the least-squares optimal weight coefficients are presented in Table 3.5. Each weight coefficient shows how much the target variable would change on average if the corresponding feature is increased by a unity, while the others do not change. One can see that increasing population by a thousand would give a similar effect as adding a primary school, about 0.2, which may seem absurd in the example as Post Office variable can have only integer values. Moreover, the linear function format should not trick the decision maker into thinking that increasing different input features can be done independently: the features are obviously not independent so that increase of, say, the population will lead to respectively adding new schools for the additional children. Still, the weights show relative effects of the features—according to Table 3.5, adding a doctor's surgery in a town would lead to maximally possible increase in post offices. The maximum value is assigned to the intercept in this case. What this may mean? Is it the number of post offices in an empty town with no population, hospitals or petrol stations? Certainly not. The intercept expresses that part of the target variable which is relatively independent of the features taken into account. It should be also pointed out that the weight values are relative not to just feature concepts but specific scales in which features measured. Change of a feature scale, say 10-fold, would result in a corresponding, inverse, change of its weight (due to the linearity of the regression equation). This is why in statistics the relative weights are considered for the scales expressed in units of the standard deviation. To find them, one should multiply the weight for the current scale by the feature's standard deviation (see Table 3.6).

The standardized weights are well justified when input features are mutually uncorrelated—indeed, they show the pair-wise correlation with the target feature. Yet in a situation of correlated features, like this, they seem to have much less definite interpretation, except for showing the changes of the target in units of the standard deviations, although some claim that they also reflect feature's correlation with the target or even importance for predicting the target. An argument against their usage as a correlation measure is that, in fact, a regression coefficient multiplied by the standard deviation loses its "purity" as a measure of correlation to the target at constant levels of the other features because the standard deviation does not pertain to constant features. An argument against their usage as measures of

Table 3.6 Standardized weight coefficients of input features at Post Office as target variable for Market towns

Feature	Weights in natural scales, w	Standard deviations, s	Weights in standardized scales, w *s
Pop_Res	0.0002	6193.2	1.3889
PSchools	0.1982	3.7344	0.5419
Doctors	0.2623	1.3019	0.3414
Hospitals	−0.2659	0.58	−0.1542
Banks	0.077	3.384	0.3376
Superstores	0.0028	1.7242	0.0048
Petrol	−0.3894	1.637	−0.6375

importance for prediction is that the standardized coefficient has nothing to do with the change of the coefficient of determinacy when the corresponding feature is removed from the equation of regression.

Bring (1994) proposes to kill two birds with one stone: to clean up the standard deviations from the non-constancy of the other features, which are claimed to reflect the changes in the coefficients of determinacy. Specifically, take the variance of a feature and take off the proportion of it unexplained by the linear regression of it through the other features. The square root of the result represents the partial standard deviation, which is proportional to the so-called "t-value", and, in the original squared form, to the change of the coefficient of determinacy inflicted by the removal of the feature from the list of the explanatory variables (Bring 1994). Unfortunately, this is not that simple, as the next Case-Study 3.4 shows.

Case-Study 3.4. Using Feature Weights Standardized

Table 3.7 presents the feature weights standardized with both the original and partial standard deviations as well as the absolute reductions of the original coefficient of determinacy 0.8295 after removal of the corresponding variables. There is a general agreement between the absolute values of the first column and those in the third column, but the second column has little in common with either of them.

Table 3.7 Different indexes to express the idea of importance of a feature in the Post regression problem

Feature	Weights expressed in standard deviations	Weights with partial standard deviations	Determinacy coefficient reduction
Pop_Res	1.3889	1603	0.0247
PSchools	0.5419	1.02	0.0077
Doctors	0.3414	0.64	0.0055
Hospitals	−0.1542	0.41	0.0023
Banks	0.3376	3.27	0.0059
Superstores	0.0048	1.07	0
Petrol	-0.6375	0.96	0.0251

Table 3.8 Weight coefficients for reduced set of features at Post Office as target variable for Market towns data

Feature	Weight
POP_RES	0.0003
PSchools	0.1823
Hospitals	−0.3167
Banks	0.0818
Petrol	−0.4072
Intercept	0.5898

A general analysis of a simpler problem of relation between the regression coefficients and correlation coefficients between the target and input features can be found in Waller and Jones (2010).

Amazingly, the convenient standardization involves negative weights, specifically at features Petrols and Hospitals. This can be an artifact of the method, related to the effect of "replication" of features. One can think of Hospitals being a double for Doctors, and Petrol, for Superstores. Thus, before jumping to conclusions, one should check whether the minus disappears if the "replicas" are removed from the set of features. As Table 3.8 shows, not in this case: the negative weights remain, though they slightly change, as well as other weights. This illustrates that the interpretation of linear regression coefficients as weights should be cautious and restrained.

In our example, coefficient of determinacy $\rho^2 = 0.83$, that is, the seven features explain 83% of the variance of Post Office feature, and the multiple correlation is $\rho = 0.91$. Curiously, the reduced set of five features (see Table 3.8) contributes almost the same, 83.4% of the variance of the target variable. This may make one wonder whether just one Population feature could suffice for doing the regression. This can be tested with the 2D method described in Sect. 3.1 or with the nD method of this section.

According to the formulation of the multivariate linear regression method, the estimated parameters must be feature weight coefficients—no room for an intercept in the formula. To accommodate the intercept, a fictitious feature whose all values are unities is introduced. That is, an input data matrix X with two columns is to be used: one for the Population feature, the other for the fictitious variable of all ones. According to (3.10), this leads to the slope 0.0003 and intercept 0.4015, though with somewhat reduced coefficient of determinacy, which is $\rho^2 = 0.78$ in this case. From the prediction point of view this may be all right, but the reduced feature set looses on interpretation.

3.4 Linear Discrimination and SVM

3.4.1 Linear Discrimination

Discrimination is an approach to address the problem of drawing a rule to distinguish between two classes of entity points in the feature space, a "yes" class and "no" class, such as for instance a set of banking customers in which a, typically very small, subset of fraudsters constitutes the "yes" class and that of the others the "no" class. On Fig. 3.16, entities of "yes" class are presented by circles and of "no" class by squares.

The problem is to find a function $u = f(x)$ that would separate the two classes in such a way that $f(x)$ is positive for all entities in the "yes" class and negative for all the entities in the "no" class. When the discriminant function $f(x)$ is assumed to be linear, the problem is of linear discrimination. It differs from that of the linear regression in that aspect that the target values here are binary, either "yes" or "no", so that this is a classification rather than regression, problem.

The classes on Fig. 3.16 can be discriminated by a straight—dashed—line indeed. The dotted vector w, orthogonal to the "dashed line" hyperplane, represents a set of coefficients at the linear classifier represented by the dashed line. Vector w also shows the direction at which function $f(x) = \langle w, x \rangle - b$ grows. Specifically, $f(x)$ is 0 on the separating hyperplane, and it is positive above, and negative beneath, that. With no loss of generality, w can be assumed to have its length equal to unity. Then, for any x, the inner product $\langle w, x \rangle$ would express the length of the projection of vector x along the direction of w.

To find an appropriate w, even in the case when "yes" and "no" classes are linearly separable, various criteria can be utilized. A most straightforward classifier is defined as follows: put 1 for "yes" and −1 for "no" and apply the least-squares criterion of linear regression. This produces a theoretically sound solution approximating the best possible—Bayesian—solution in a conventional statistics model. Yet, in spite of its good theoretical properties, least-squares solution may be

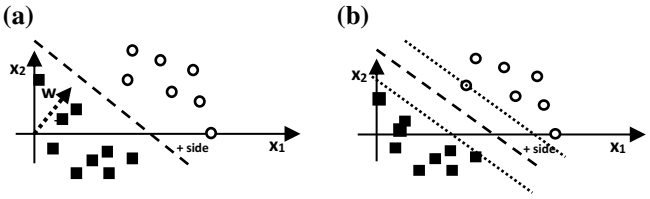

Fig. 3.16 A geometric illustration of a separating hyper-plane between classes of circles and squares. The dotted vector w on **a** is orthogonal to the hyper-plane: its elements are hyper-plane coefficients, so that it is represented by equation $\langle w, x \rangle - b = 0$. Vector w points at the direction: at all points above the dashed line, the circles included, function $f(x) = \langle w, x \rangle - b$ is positive. The dotted lines on, **b** show the margin, and the squares and circle on them are support vectors

not necessarily the best at some data configurations. In fact, it may even fail to separate the positives from negatives when they are linearly separable. Consider the following example.

Worked Example 3.4. A Failure of Fisher Discrimination Criterion
Let there be 14 2D points that are presented in Table 3.9 (first line) and displayed in Fig. 3.17a. Points 1,2,3,4,6 belong to the positive class (dots on Fig. 3.17a), and the others to the negative class (stars on Fig. 3.17a). Another set, obtained by adding to each of the components a random number, according to the normal distribution with zero mean and 0.2 the standard deviation, is presented in the bottom line of Table 3.9 and Fig. 3.17b. The class assignment for the disturbed points remains the same.

The optimal vectors w according to formula (3.15) are presented in Table 3.10 as well as that for the separating, dotted, line in Fig. 3.17d.

Note that the least-squares solution depends on the values assigned to classes, leading potentially to an infinite number of possible solutions under different numerical codes for "yes" and "no". A popular discriminant criterion of minimizing the ratio of a "within-class error" over "out-of-class error", proposed by R. Fisher in his founding work of 1936, in fact, can be expressed with the least-squares criterion as well. Just change the target as follows: assign N/N_1, rather than +1, to "yes" class

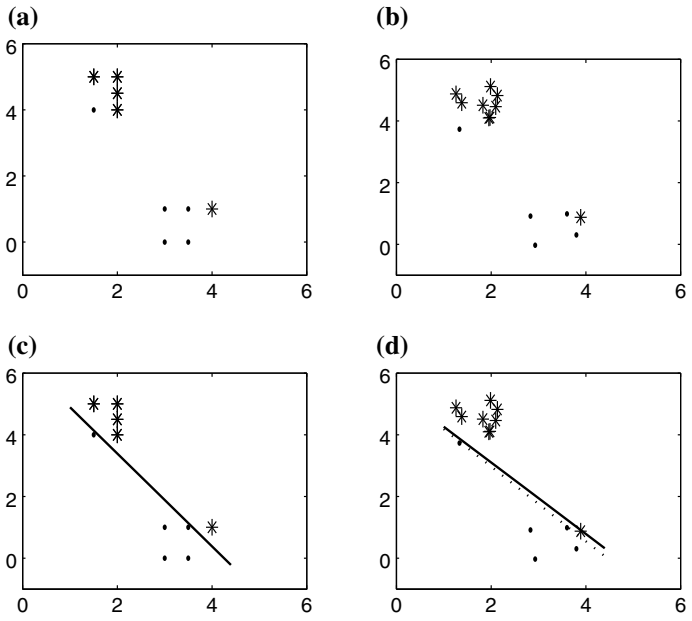

Fig. 3.17 **a** and **b** represent the original and perturbed data. The least squares optimal separating line is added in **c** and **d** shown by solid. Entity 5 is wrongly assigned to the "dot" class according to the solid line in (**d**); a separating line is shown dotted there

Table 3.9 *x-y* coordinates of 14 points as given originally and perturbed with a white noise of standard deviation 0.2, that is, generated from the Gaussian distribution N(0,0.2)

Entity #		1	2	3	4	5	6	7	8	9	10	11	12	13	14
Original data	x	3.00	3.00	3.50	3.50	3.00	1.50	3.00	3.00	3.00	1.50	3.00	3.00	3.00	1.50
	y	0.00	1.00	1.00	0.00	1.00	3.00	3.00	5.00	3.50	5.00	3.00	5.00	3.50	5.00
Perturbed data	x	3.93	3.83	3.60	3.80	3.89	1.33	1.95	3.13	1.83	1.26	1.98	1.99	3.10	1.38
	y	−0.03	0.91	0.98	0.31	0.88	3.73	3.09	3.82	3.51	3.87	3.11	5.11	3.46	3.59

Table 3.10 Coefficients of straight lines on Fig. 3.17

	Coefficients at		
	x	*y*	Intercept
LSE at original data	−1.2422	-0.8270	5.2857
LSE at perturbed data	−0.8124	−0.7020	3.8023
Dotted at perturbed data	−0.8497	−0.7020	3.7846

and $-N/N_2$ to "no" class, rather than −1 (see Duda et al. 2001, pp. 242–243). This means that Fisher's criterion may also lead to a failure in a linearly separable situation (see Fig. 3.17d).

By far the most popular set of techniques, Support Vector Machine (SVM), utilize a different criterion—that of maximum margin. The margin of a point x, with respect to a hyperplane, is the distance from x to the hyperplane along its perpendicular vector w (Fig. 3.17a), which is measured by the absolute value of inner product $\langle w, x \rangle$. The margin of a class is defined by the minimum value of the margins of its members. Thus the criterion requires, like L_∞, finding such a hyperplane that maximizes the minimum of class margins, that is, crosses the middle of line between the nearest entities of two classes. Those entities that fall on the margins, shown by dotted lines on Fig. 3.17b, are referred to as support vectors; this explains the method's title.

It should be noted that the classes are not necessarily linearly separable; moreover, in most cases they are not. Therefore, the SVM technique is accompanied with a non-linear transformation of the data into a high-dimensional space which is more likely to make the classes linear-separable. Such a non-linear transformation is provided by the so-called kernel function. The kernel function imitates the inner product in the high-dimensional space and is represented by a between-entity similarity function such as that defined by formula (3.20) on p. 202.

The intuition behind the SVM approach is this: if the population data—those not present in the training sample—concentrate around training data, then having a wide margin would keep classes separated even after other data points are added (see Fig. 3.18). One more consideration comes from the Minimum Description Length principle: the wider the margin, the more robust the separating hyperplane

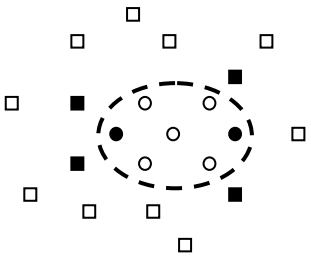

Fig. 3.18 Illustrative example of 2D entities belonging to two classes, circles and squares. The separating line in the space of Gaussian kernel is shown by the dashed oval. The support entities are shown by black

is and the less information of it needs to be stored. A criticism of the SVM approach is that the support vector machine hyperplane is based on the borderline objects—support vectors—only, whereas the least-squares hyperplanes take into account all the entities so that the further away an entity is the more it may affect the solution, because of the quadratic nature of the least-squares criterion. Some may argue that both borderline and far away entities can be rather randomly represented in the sample under investigation so that neither should be taken into account in distinguishing between classes: it is some "core" entities of the patterns that should be separated—however, there has been no such an approach taken in the literature so far.

3.4.2 Support Vector Machine (SVM) Criterion

Another criterion would put the separating hyperplane just in the middle of an interval drawn through closest points of the different patterns. This criterion produces what is referred to as the support vector machine since it heavily relies on the points involved in the drawing of the separating hyperplane (as shown on the right of Fig. 3.16). These points are referred to as support vectors. A natural formulation would be like this: find a hyperplane $H : \langle w, x\rangle = b$ with a normed w to maximize the minimum of absolute values of distances $|\langle w, x_i\rangle - b|$ to H from points x_i belonging to each of the classes. This, however, is rather difficult to associate with a conventional formulation of an optimization problem because of the following irregularities:

(i) an absolute value to maximize,
(ii) the minimum over points from each of the classes, and
(iii) w being of the length 1, that is, normed.

However, these all can be successfully tackled. The issue (i) is easy to handle, because there are only two classes, on the different sides of H. Specifically, the distance is $\langle w, x_i\rangle - b$ for "yes" class and $-\langle w, x_i\rangle + b$ for "no" class—this removes the absolute values. The issue (ii) can be taken care of by uniformly using inequality constraints

$\langle w, x_i\rangle - b \geq \lambda$ for x_i in "yes" class and
$-\langle w, x_i\rangle + b \geq \lambda$ for x_i in "no" class

and maximizing the margin λ with respect to these constraints. The issue (iii) can be addressed by dividing the constraints by λ so that the norm of the weight vector becomes $1/\lambda$, thus inversely proportional to the margin λ. Moreover, one can change the criterion now because the norm of the ratio w/λ is minimized when λ is

maximized. Denote the "yes" class by $u_i = 1$ and "no" class by $u_i = -1$. Then the problem of deriving a hyperplane with a maximum margin can be reformulated, without the irregularities, as follows: find b and w such that the norm of w or its square, $\langle w, w \rangle$, is minimum with respect to constraints

$$u_i(\langle w, x_i \rangle - b) \geq 1 \quad (i = 1, 2, \ldots, N) \tag{3.17}$$

This is a problem of quadratic programming with linear constraints, which is easier to analyze in the format of its dual optimization problem. The dual problem can be formulated by using the so-called Lagrangian, a common concept in optimization, that is, the original criterion penalized by the constraints weighted by the so-called Lagrange multipliers that are but penalty rates. Denote the penalty rate for the violation of i-th constraint by α_i. Then the Lagrangian can be expressed as

$$L(w, b, \alpha) = \langle w, w \rangle / 2 + \Sigma_i \alpha_i (u_i(\langle w, x_i \rangle - b) - 1), \tag{3.18}$$

where $\langle w, w \rangle$ has been divided by 2 with no loss of generality, just for the sake of convenience. The optimum solution minimizes L over w and b, and maximizes L over non-negative α. The first order optimality conditions require that all partial derivatives of L are zero at the optimum, which leads to equations $\Sigma_i \alpha_i u_i = 0$ and $w = \Sigma_i \alpha_i u_i x_i$. Multiplying the latter expression by itself leads to equation $\langle w, w \rangle = \Sigma_{ij} \alpha_i \alpha_j u_i u_j \langle x_i, x_j \rangle$. The second item in Lagrangian L becomes equal to $\Sigma_i \alpha_i u_i \langle w, x_i \rangle - \Sigma_i \alpha_i u_i b - \Sigma_i \alpha_i = \langle w, w \rangle - 0 - \Sigma_i \alpha_i$. This leads us to the following, dual, problem of optimization regarding the Lagrangian multipliers, which is equivalent to the original problem: Maximize criterion

$$\Sigma_i \alpha_i - \Sigma_{ij} \alpha_i \alpha_j u_i u_j \langle x_i, x_j \rangle / 2 \tag{3.19}$$

subject to $\Sigma_i \alpha_i u_i = 0$ and $\alpha_i \geq 0$.

Support vectors are defined as those x_i for which penalty rates are positive, $\alpha_i > 0$, in the optimal solution—only they contribute to the optimal vector $w = \Sigma_i \alpha_i u_i x_i$; the others have zero coefficients and disappear.

It should be noted that the margin constraints can be violated, which is not difficult to take into account—by using non-negative values η_i expressing the extent of violations:

$$u_i(\langle w, x_i \rangle - b) \geq 1 - \eta_i \quad (i = 1, 2, \ldots, N)$$

in such a way that they are minimized in a combined criterion $\langle w, w \rangle / 2 + C \Sigma_i \eta_i$ where C is a large "reconciling" coefficient that is a user-defined parameter. The dual problem for the combined criterion remains almost the same as above, in spite of the fact that an additional set of dual variables, β_i, needs to be introduced as

corresponding to the constraints $\eta_i \geq 0$. Indeed, the Lagrangian for the new problem can be expressed as

$$L(w, b, \alpha, \beta) = \langle w, w \rangle / 2 - \Sigma_i \alpha_i (u_i (\langle w, x_i \rangle - b) - 1) - \Sigma_i \eta_i (\alpha_i + \beta_i - C),$$

which differs from the previous expression by just the right-side item. This implies that the same first-order optimality equations hold, $\Sigma_i \alpha_i u_i = 0$ and $w = \Sigma_i \alpha_i u_i x_i$, plus additionally $\alpha_i + \beta_i = C$. These latter equations imply that $C \geq \alpha_i \geq 0$ because β_i are non-negative.

Since the additional dual variables are expressed through the original ones, $\beta_i = C - \alpha_i$, the dual problem can be shown to remain unchanged and it can be solved by using quadratic programming algorithms (see Vapnik 2006; Schölkopf and Smola 2005). Recently, approaches have appeared for solving the original problem as well (see Groenen et al. 2008).

3.4.3 Kernels

Situations at which classes are linearly separable are very rare; in real data, classes are typically well intermingled with each other. To attack these typical situations with linear approaches, the following trick can be applied. The data are nonlinearly transformed into a much higher dimensional space in which, because of both nonlinearity and higher dimension, the classes may be linearly separable. The transformation can be performed only virtually because of specifics of the dual problem: dual criterion (3.19) depends not on individual entities but rather just inner products between them. This property obviously translates to the transformed space, that is, to the transformed entities. The inner products in the transformed space can be computed with the so-called kernel functions $K(x,y)$ so that in criterion (3.19) inner products $\langle x_i, x_j \rangle$ are substituted by the kernel values $K(x_i,x_j)$. Moreover, by substituting the expression $w = \Sigma_i \alpha_i u_i x_i$ into the original discrimination function $f(x) = \langle w, x \rangle - b$ we obtain its different expression $f(x) = \Sigma_i \alpha_i u_i \langle x, x_i \rangle - b$, also involving inner products only, which can be used as a kernel-based decision rule in the transformed space: x belongs to "yes" class if $\Sigma_i \alpha_i u_i K(x, x_i) - b > 0$.

It is convenient to define a kernel function over vectors $x = (x_v)$ and $y = (y_v)$ through the squared Euclidean distance $d^2(x,y) = (x_1 - y_1)^2 + \ldots + (x_V - y_V)^2$ because matrix $(K(x_i,x_j))$ in this case is positive definite—a defining property of matrices of inner products. Arguably, the most popular is the Gaussian kernel defined by:

$$K(x, y) = \exp\left(-d^2(x, y)\right) \tag{3.20}$$

Q.3.8. Consider a full set B_n of 2^n binary 1/0 vectors of length n like those presented by columns below for $n = 3$:

1	0	0	0	0	1	1	1	1
2	0	0	1	1	0	0	1	1
3	0	1	0	1	0	1	0	1

These columns can be considered as integers coded in the binary number system; moreover, they are ordered from 0 to 7. Prove that this set shutters any subset of n (or less) points.

A. Indeed, let S be a set of elements $i_1, i_2, \ldots, i_n$ in B_n that are one-to-one labeled by numbers from 1 to n. Consider any partition of S in two classes, S_1 and S_3. Assign 0 to each element of S_1 and 1 to each element of S_3. The partition follows that vector of B_n that corresponds to the assignment.

Q.3.9. Consider set B_n defined above. Prove that its rank is n, that is, there are n columns in matrix B_n that form a base of the space of n-dimensional vectors.

A. Take, for example, n columns e_p that contain unity at p-th position whereas other n-1 elements are zero ($p = 1, 2, \ldots n$). These obviously are mutually orthogonal and any vector $x = (x_1,\ldots,x_n)$ can be expressed as a linear combination $x = \Sigma_p x_p e_p$, which proves that vectors e_p form a base of the n-dimensional space.

Q.3.10. What is VC-dimension of the linear discrimination problem at an arbitrary dimension $p \geq 2$?

A. The VC = dimension in this case is $p + 1$, because each subset of p points can be separated from the others by a hyperplane, but there can be such $(p + 1)$-point configurations that cannot be shattered using linear separators.

Worked Example 3.5. SVM for the Iris Dataset

Consider Iris dataset standardized by subtracting, from each feature column, its midrange and dividing the result by the half-range.

Take the Gaussian kernel in (3.20) to find a support vector machine surface separating Iris class 3 from the rest. The resulting solution embraces 21 supporting entities (see Table 3.11), along with their "alpha" prices reaching into hundreds and even, on two occasions, to the maximum boundary 500 embedded in the algorithm.

There is only one error with this solution, entity 78 wrongly recognized as belonging to taxon 3. The errors increase when we apply a cross-validation techniques, though. For example, "leave-all-one-out" cross-validation leads to nine errors: entities 63, 71, 78, 82 and 83 wrongly classified as belonging to taxon 3 (false positives), while entities 127, 133, 135 and 139 are classified as being out of taxon 3 (false negatives).

Table 3.11 List of support entities in the problem of separation of taxon 3 (entities 101 to 150) in Iris data set from the rest (thanks to V. Sulimova for the computation)

N	Entity	Alpha	N	Entity	Alpha
1	18	0.203	12	105	3.492
2	28	0.178	13	106	15.185
3	37	0.202	14	115	53.096
4	39	0.672	15	118	15.724
5	58	13.63	16	119	449.201
6	63	209.614	17	127	163.651
7	71	7.137	18	133	500
8	78	500	19	135	5.221
9	81	18.192	20	139	16.111
10	82	296.039	21	150	26.498
11	83	200.312			

Q.3.11. Why only 10, not 14, points are drawn on Fig. 3.17b?

A. Because each of the points 11–14 doubles a point 7–10.

Q.3.12. What would change if the last four points are removed so that only points 1–10 remain?

A. The least-squares solution will be separating again

3.5 Learning Correlation with Neural Networks

3.5.1 Artificial Neuron and Neural Network: Presentation

Neural network is one of the most popular structures used for predictions of target features. It is a network of artificial neurons modeling the neuron cell in a living organism. A neuron cell fires an output when its summary input becomes higher than a threshold. Dendrites bring signal in, axons pass it out, and the firing occurs via a synapse, a gap between neurons, that makes the threshold (see Fig. 3.19).

This is modeled in the concept of artificial neuron as follows (see Fig. 3.20). A neuron model is drawn as a set of input elements connected to an output. The connections are assigned with wiring weights.

The input signals are data features or other neurons' outputs. The output element receives a combined signal, the sum of feature values weighted by the wiring weights. The output compares this with a firing threshold, otherwise referred to as a bias, and fires an output depending on the result. Ideally, the output is 1 if the combined signal is greater than the threshold, and −1 if it is smaller than that. This is, in fact, what is called the sign function of the difference, $sign(x)$, which is 1, 0 or −1 if x is positive, zero or negative, respectively. This activation function is overly straightforward sometimes. Instead, the so-called sigmoid and symmetric sigmoid

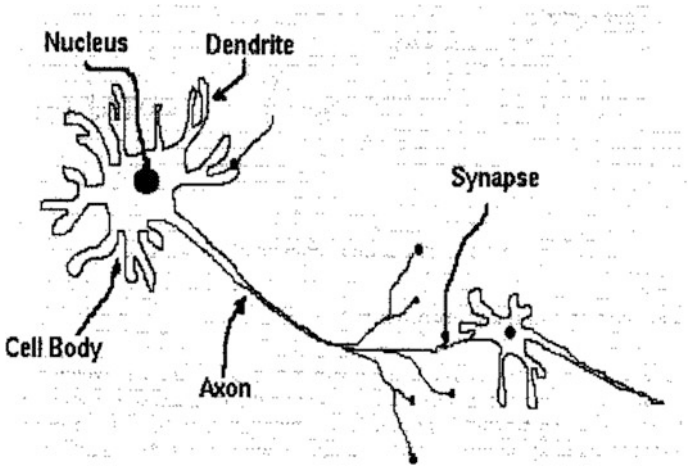

Fig. 3.19 Scheme of a neural cell

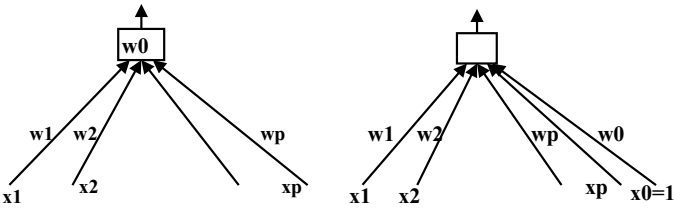

Fig. 3.20 A scheme of an artificial neuron, on the left. The same neuron with the firing threshold shown as a wiring weight on the fictitious input always equal to 1 is on the right

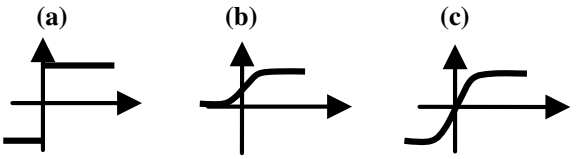

Fig. 3.21 Graphs of sign (**a**), sigmoid (**b**) and symmetric sigmoid (**c**) functions

functions are considered as smooth exponent-based counterparts to sign(x). Their graphs are shown alongside with that for *sign*(*x*) on Fig. 3.21. Sometimes the output element is assumed as doing no transformation at all, just passing the combined signal as the neuron's output, which is referred to as a linear activation function.

The firing threshold, or bias, hidden in the box in neuron on the left on Fig. 3.20, can be made explicit if one more, fictitious, input is added to the neuron.

This input is always equal to 1 so that its wiring weight is always added to the combined input to the neuron. It is assumed to be equal to minus the bias so that the total sum is the difference between the combined signal and the bias. In the

remainder, we assume that the bias, with the minus sign, is always explicitly present among the wiring weights in this way (see Fig. 3.20 on the right).

Artificial neurons can be variously combined in neural networks. There have been defined many specific types of neural network structures, referred to as architectures, of which the most generic is a three-layer structure with no feedback connections, referred to as a feedforward neural network. Such a network is presented on Fig. 3.22. There are two outbound layers, the input and output ones, and one intermediate layer which is referred to as a hidden layer. This is why such a structure is referred to as a one hidden-layer neural network (NN).

Network on Fig. 3.22 is designed as a one-hidden-layer NN for predicting petal sizes of Iris features from their sepal sizes. Recall that in Iris data set, each of 150 specimens is presented with four features which are the length and width of petals (features w3 and w4) and sepals (features w1 and w2). It is likely that the sepal sizes and petal sizes are related.

In fact, the further material can be used for building an NN for modeling correlation between any inputs and outputs—the only possible difference, in numbers of input and/or output units, plays no role in the organization of computations.

This neural network consists of the following layers:

(a) Input layer that accepts three inputs: a bias input $x_0 = 1$ as explained above (see Fig. 3.20 on the right) as well as sepal length and width; these are combined to be inputs to each of the neurons at the hidden layer.
(b) Output layer producing an estimate for petal length and width with a linear activation function. Its input is the output signals from the hidden layer. No fictitious input $x_0 = 1$ is assumed here because the activation function here just passes the combined signal through without a threshold.

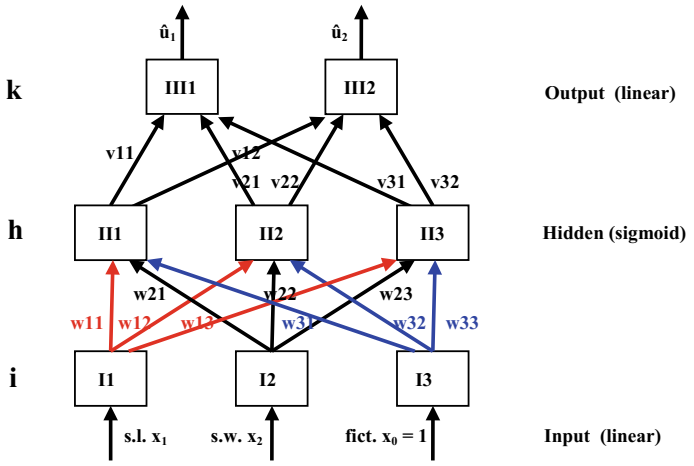

Fig. 3.22 A feed-forward network with two input and two output features (no feedback loops). Layers: Input (I, indexed by *i*), Output (III, indexed by *k*) and Hidden (II, indexed by *h*)

(c) Hidden layer consisting of three neurons. Each of them takes a combined input from the first layer and applies to it its sigmoid activation function. The output signals of these three neurons constitute inputs to the output layer. The architecture allows for any number of hidden neurons with no changes in the computations.

The one-hidden-layer structure is generic in NN theory. It has been proven, for instance, that such a structure can exactly learn any subset of the set of entities. Moreover, any pre-specified mapping of inputs to outputs can be approximated up to a pre-specified precision with such a one-hidden-layer network, if the number of hidden neurons is large enough (Haykin 1999). This property, for many years, served as a justification for the specialists to consider one-layer neural networks only. The past decade saw an explosion of research on deep neural networks, that is, networks with a dozen or more layers (see, for example, LeCun et al. 2015). It appears deep neural networks are capable to capture non-linear hidden features and, in this way, dramatically improve precision of the neural network decision rules. Deep learning is an area of intense research efforts which should bring novel breakthroughs in machine learning. There is a caveat though. The hidden features are hidden indeed. There is no way to explicitly describe relations between given features and the hidden one. That brings forward the I. Asimov's controversy pointed to in his "I, Robot" series: unlike in data analysis, machine learning may generate unexpected and unpredictable consequences.

3.5.2 Activation Functions and Network Function

Two popular activation functions, besides the *sign* function $\mathring{u}_i = sign(\hat{u}_i)$, are the *linear* activation function, $\mathring{u}_i = \hat{u}_i$ and *sigmoid* activation function $\mathring{u}_i = s(\hat{u}_i)$ where

$$s(x) = (1 + e^{-x})^{-1} \tag{3.21}$$

is a smooth analogue to the sign function, except for the fact that its output is between 0 and 1 rather than −1 and 1 (see Fig. 3.21b). To imitate the perceptron with its $sign(x)$ output, between −1 and 1, we first double the output interval and then subtract 1 to obtain what is referred to as a symmetric sigmoid or hyperbolic tangent:

$$th(x) = 2s(x) - 1 = 2(1 + e^{-x})^{-1} - 1 \tag{3.21′}$$

This function, illustrated on Fig. 3.21c, in contrast to sigmoid $s(x)$, is symmetric: $th(-x) = -th(x)$, like $sign(x)$, which can be useful in some contexts.

The sigmoid activation functions have nice mathematical properties; they are not only smooth, but their derivatives can be expressed through the functions themselves, see Q.3.13 and (3.28) further on.

Let us express now the function of the one-hidden-layer neural network presented on Fig. 3.22. Its wiring weights between the input and hidden layer form a matrix $W = (w_{ih})$, where i denotes an input, and h a hidden neuron, $h = 1, 2,\ldots, H$ where H is the number of hidden neurons. The wiring weights between the hidden and output layers form matrix $V = (v_{hk})$, where h denotes a hidden neuron and k an output.

Layers I and III are assumed to be linear giving no transformation to their inputs; all of the hidden layer neurons will be assumed to have a symmetric sigmoid as their activation function.

To find out an analytic expression for the network's output, let us work it out layer by layer. Neuron h in the hidden layer receives, as its input, a combined signal

$$z_h = w_{1h}x_1 + w_{2h}x_2 + w_{3h}x_0$$

which is h-th component of vector $z = \Sigma_i x_i * w_{ih} = x * W$ where x is a 1×3 input vector. Then its output will be $th(zh)$. These constitute an output vector $th(z) = th(x * W)$ that is input to the output layer. Its k-th node receives a combined signal $\Sigma_j v_{jk} * th(z_j)$ which is k-th component of the matrix product $th(z) * V$, that is passed as the NN output $\hat{u}$. Therefore, the NN on Fig. 3.22 transforms input x into output $\hat{u}$ according to the following formula

$$\hat{u} = th(x * W) * V \tag{3.22}$$

which combines linear operations of matrix multiplication with a nonlinear symmetric sigmoid transformation. If matrices W, V are known, (3.22) computes the function $u = F(x)$ in terms of th, W, and V. The problem is to fit this model with training data provided, at this instance, by the Iris data set.

3.5.3 *Learning a Multi-layer Network*

Given all the wiring weights W, between the input and hidden layers, and wiring weights V, between the hidden and output layers, as well as pre-specified hidden layer activation functions, the NN on Fig. 3.22 takes an input of the sepal length and width and transforms it into estimates of the corresponding petal length and width.

The quality of the estimates can be measured by the average squared error. The better adapted weights W and V are, the smaller the error. Where the weights come from? They are learnt from the training data.

Thus, the problem is to estimate weight matrices W and V at the training data in such a way that the average squared error is minimized.

The machine learning paradigm is based on the assumption that a learning device adapts itself incrementally by facing entities one by one. This means that the full sample is assumed to be never known to the device so that global solutions,

such as the orthogonal projection used in linear discrimination, are not applicable. In such a situation an optimization algorithm that processes entities one by one should be applied. Such is the gradient method, also referred to as the steepest descent.

This method relies on the so-called gradient of the function to be optimized (see Sect. A.2 in Appendix). The gradient is a vector that can be derived or estimated at any admissible solution, that is, matrices W and V. This vector shows the direction of the steepest ascent over the optimized function considered as a surface. Its elements are the so-called partial derivatives of the optimized function that can be derived according to rules of calculus. The gradient is useful for maximizing a criterion, but how one can do minimization with the steepest ascent? Easily, by moving in the opposite direction, that is, taking minus gradient.

Assume, we have some estimates of matrices W and V as well as their gradients, that is, matrices gW and gV, whose components express the steepest ascent direction of changes in W and V. Then, according to the method of steepest descent, the matrices V and W should be moved in the direction of $-gW$ and $-gV$ with the control of the length of the step by a factor referred to as the learning rate. The equations expressing the move from the old state to the new one are as follows:

$$V(new) = V(old) - \mu * gV, W(new) = W(old) - \mu * gW \tag{3.23}$$

where μ is the learning rate (step size). The importance of properly choosing the step size is illustrated on Fig. 3.23.

The gradient of the criterion of squared error is defined by: (a) the matrices W and V, (b) the error value itself, and (c) the input feature values. This is why it is convenient to apply this approach when entities come in a sequence so that each individual entity gives an estimate of the gradient and, accordingly, the move to a new state of matrices W and V according to Eq. (3.23). The sequence of entities is natural when the learning is done on the fly by processing entities in the order of their arrival. In the situations when all the entities have been already collected in a data set, as the Iris data set, the sequence is organized artificially in a random order. Moreover, as the number of entities is typically rather small (as it is in the case of just 150 Iris specimens) and the gradient process is rather slow, it is usually not enough to process all the entities just once. The processing of all the entities in a

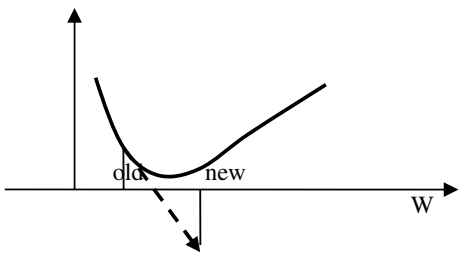

Fig. 3.23 The importance of properly choosing the step in the steepest descent process: if the leap is too big, the new state may be worse than the old one

Table 3.12 Relative error values in the predicted petal dimensions with full Iris data after 5000 epochs

Number of hidden neurons	Relative error, %	
	Petal length	Petal width
3	5.36	8.84
6	3.99	8.40
10	3.98	8.15

random order constitutes *an epoch*. A number of epochs need to be executed until the matrices V and W are more or less stabilized.

Worked Example 3.6. Learning Iris Petal Sizes
Consider, at any Iris specimen, its two sepal sizes as the input and its two petal sizes as the output. We are going to find a decision rule relating them in the format of a one-hidden-layer NN.

At the Iris data, the architecture presented on Fig. 3.22 and program nnn.m implementing the error back-propagation algorithm leads to the average errors at each of the output variables presented in Table 3.12 at different numbers of hidden neurons h. Note that the errors are given relative to feature ranges.

The number of elements in matrices V and W here are five-fold of the number of hidden neurons, thus ranging from 15 at the current setting of three hidden neurons to 50 when this grows to 10. One can see that the increase in the numbers of hidden neurons does bring some improvement, but not that great—probably not worth doing.

Here are a few suggestions for further work on this example:

1. Find values of E for the errors reported in Table above.
2. Take a look at what happens if the data are not normalized.
3. Take a look at what happens if the learning rate is increased, or decreased, ten times.
4. Extend the table above for different numbers of hidden neurons.
5. Try petal sizes as input with sepal sizes as output.
6. Try predicting only one size over all input variables.

Worked Example 3.7. Predicting Marks at Student Dataset
Let us embark on an ambitious task of predicting student marks at the Students data—we partially dealt with this in Sect. 3.3. The nnn.m program leads to the average errors in predicting student marks over three subjects, as presented in Table 3.13 at different numbers of hidden neurons h. Surprisingly, the prediction works rather well: the errors are on the level of 3 points only, more or less independently of the number of hidden neurons utilized.

Table 3.13 Average absolute error values in the predicted student marks over all three subjects, with full Student data after 5,000 epochs

H	|e1|	|e2|	|e3|	# param.
3	3.65	3.16	3.17	27
6	3.29	3.03	3.75	54
10	3.17	3.00	3.64	90

3.5.4 *Steepest Descent for the Square Error Criterion with Linear Rules*

In machine learning, the assumption is that the decision rule is learnt incrementally by using entities one by one. That is, the global solutions involving the entire sample are not applicable. In such a situation an optimization algorithm that processes entities one by one should be applied. The most popular is the gradient method, also referred to as the steepest descent.

This method relies on the gradient of the function to be optimized. If we are to minimize function $f(x)$ over x spanning a subspace D of the n-dimensional vector space R^n, we can utilize its gradient gf for this purpose. The gradient gf at $x \in D$ is a vector consisting of the f's partial derivatives over all components of x, under the assumption that a full derivative, geometrically corresponding to the tangential hyperplane, does exist. This vector shows the direction of the steepest ascent of $f(x)$, so that its opposite vector $-gf$ shows the opposite direction which is considered as that of the steepest descent of $f(x)$. The method of steepest descent produces a sequence of points $x(0)$, $x(1)$, $x(2)$, … starting from an arbitrary $x(0)$ by using recursive equation

$$x(t+1) = x(t) - \mu_t * gf(x(t)) \qquad (3.23')$$

where parameter μ_t denotes the length of the step to go from $x(t)$ in the direction of the steepest descent, referred to as the learning rate in machine learning. The sequence $x(t)$ is guaranteed to converge to the minimum point at a constant $\mu_t = \mu$ if $f(x)$ is strictly convex, so that there is a sphere of a finite radius such that $f(x)$ is always greater than its lower part, as shown on the right of Fig. 3.24 (see Polyak 1987).

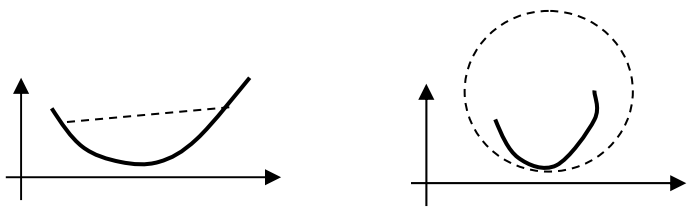

Fig. 3.24 A convex function, on the left and strictly convex function, on the right

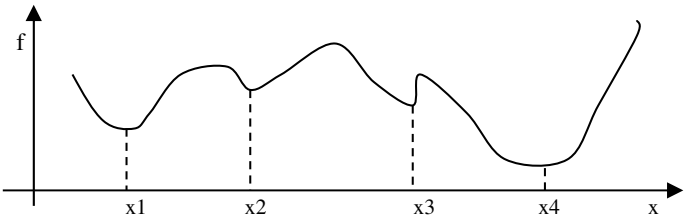

Fig. 3.25 Points x1–x4 are points of local minimum for the function whose graph is drawn with the line. The global minimum is only one of them, x4

The process converges if $f(x)$ is a convex function and μ_t tends to 0 when t grows to infinity, but not too fast so that the sum of the series $\Sigma_t \mu_t$ is infinity. This guarantees that the moves from $x(t)$ to $x(t + 1)$ are small enough to not over-jump the point of minimum but not that small to stop the sequence short of reaching the optimum by themselves.

If $f(x)$ is not convex however, the sequence reaches just one of the local optima depending on the starting point $x(0)$ (see Fig. 3.25). Luckily, the square error in the problem of linear discriminant analysis is strictly convex so that the steepest descent sequence converges to the optimum from any initial point. This gives rise to the algorithm described in the following section.

3.5.5 *Learning Wiring Weights with Error Back-Propagation*

The problem of learning a neural network is to find weight matrices W and V minimizing the squared difference between u observed and $\hat{u}$ computed:

$$E = d(u, \hat{u}) = \langle u - th(x * W) * V, u - th(x * W) * V \rangle / 2, \tag{3.24}$$

over the training entity set. The division by 2 is made to avoid factor 2 in the derivatives of E.

Specifically, with just two outputs in Fig. 3.22, the error function is

$$E = \left[(u_1 - \hat{u}_1)^2 + (u_2 - \hat{u}_2)^2 \right] / 2 \tag{3.24'}$$

where $u_1 - \hat{u}_1$ and $u_2 - \hat{u}_2$ are differences between the actual and predicted values of the two outputs.

Steepest descent equations (3.23) for learning V and W can be written component-wise:

$$\begin{aligned} v_{hk}(t+1) &= v_{hk}(t) - \mu * \partial E/\partial v_{hk}, \\ w_{ih}(t+1) &= w_{ih}(t) - \mu * \partial E/\partial w_{ih} (i \in I,\ h \in II,\ k \in III) \end{aligned} \tag{3.25}$$

To make these computable, let us express the derivatives explicitly; first those at the output, over v_{hk}:

$$\partial E/\partial v_{hk} = -(u_k - \hat{u}_k) * \partial \hat{u}_k/\partial v_{hk}.$$

To advance, notice that $\partial \hat{u}_k/\partial v_{hk} = th(zh)$, since $\hat{u}_k = \Sigma_j th(zh) * v_{hk}$. Putting this into equation above makes

$$\partial E/\partial v_{hk} = -(u_k - \hat{u}_k) * th(zh). \tag{3.26}$$

Regarding the second layer, of W, let us find the derivative $\partial E/\partial w_{ih}$ which requires more chain-based derivations. Specifically,

$$\partial E/\partial w_{ij} = \Sigma_k \left[-(u_k - \hat{u}_k) * \partial \hat{u}_k/\partial w_{ij}\right].$$

Since $\hat{u}_k = \Sigma_j th\left(\Sigma_i x_i * w_{ij}\right) * v_{jk}$, this can be expressed as

$$\partial \hat{u}_k/\partial w_{ij} = v_{jk} * th'\left(\Sigma_i x_i * w_{ij}\right) * x_i.$$

The derivative $th'(z)$ can be expressed through $th(z)$ as explained in Q.3.13 later, which leads to the following final expression for the partial derivatives:

$$\partial E/\partial w_{ij} = -\Sigma_k \left[-(u_k - \hat{u}_k) * v_{jk}\right] * \left(1 + th(z_j)\right)\left(1 - th(z_j)\right) * x_i/2 \tag{3.27}$$

Equations (3.23), (3.26) and (3.27) lead to the following rule for processing an entity, or instance, in the error back-propagation algorithm as applied to neural network in Fig. 3.22.

1 *Forward computation (of the output $\hat{u}$ and error).* Given matrices V and W, upon receiving an instance (x,u), the estimate $\hat{u}$ of vector u is computed according to the neural network as formalized in Eq. (3.22), and the error $e = u - \hat{u}$ is calculated.
2 *Error back-propagation (for estimation of the gradient elements).* Each neuron receives the relevant error estimate, which is
   - $e_k = -(u_k - \hat{u}_k)$, for (3.26) at output neurons k ($k = III1, III2$) or
   - $\Sigma_k[(u_k - \hat{u}_k) * v_{hk}]$, for (3.27) at hidden neurons h ($j = II1, II2, II3$) [the latter can be seen as the sum of errors arriving from the output neurons according to the corresponding synapse weights].

These are used to adjust the derivatives (3.26) and (3.27) by multiplying them with local data depending on the input signal, which is $th(zh)$, for neuron k's source h in (3.26), and $th'(zh)x_i$ for neuron h's source i in (3.27).

3 *Weights update.* Matrices V and W are updated according to formula (3.23).

What is nice in this procedure is that the computation can be done locally, so that every neuron processes only the data that are available to this neuron, first from the input layer, then backwards, from the output layer. In particular, the algorithm does not change if the number of hidden neurons is changed from $h = 3$ on Fig. 3.22, to any other integer $h = 1, 2, ...$; nor does it change if the number of inputs and/or outputs changed.

3.5.6 Error Back-Propagation: Computation

For a data set available as a whole, "offline", due to the specifics of the binary target variables and activation functions, such as $th(x)$ and $sign(x)$, which have −1 and 1 as their boundaries, the data in the NN context are frequently pre-processed to make every feature's range to lie between −1 and 1 and the midrange to be 0. This can be done by using the conventional shifting and rescaling formula for each feature v, $yi_v = (xi_v - a_v)/b_v$, at which b_v is equal to the half-range, $b_v = (M_v - m_v)/2$, and shift coefficient a_v, to the mid-range, $a_v = (M_v + m_v)/2$. Here M_v denotes the maximum and m_v the minimum of feature v.

The practice of digital computation, with a limited number of digits used for representation of reals, shows that it is a good idea to further expand the ranges into a [−10,10] interval by multiplying afterwards all the entries by 10: in this range, digital numbers stored in computer arguably lead to smaller computation errors than in the range [−1,1], if they are closer to 0.

The implementation of the method of gradient descent for learning neural networks cannot be straightforward because the minimized squared error depends both on the wiring weight matrices V and W and input/output pairs (x,u), yet there is no way to freely change the latter—the process is bound by the set of observations. This is why the observed pairs (x_i,u_i), the instances, are used as triggers to the steepest descent changes in matrices V and W. Specifically, given V and W, the instances are put one by one, in a random order, to see what are the discrepances between the observed u and computed $\hat{u}$. When all of the instances have been entered, their order is randomly changed and they are ready to be put all over again—this is referred to as a new "epoch". The matrices V and W are changed either at each (x_i,u_i) instance, using the errors $\hat{u}–u$ locally, or after an epoch, using the accumulated errors.

The error back-propagation algorithm, with the local changes of matrices V and W, can be formulated as follows.

A. Initialize weight matrices $W = (w_{ih})$ and $V = (v_{hk})$ by using random normal distribution N(0,1) with the mean at 0 and the variance 1.
B. Standardize data to [−10,10] ranges and 0 averages as described above.
C. Formulate halting criterion as explained below and run a loop over epochs.
D. Randomize the order of entities within an epoch and run a loop of the error back-propagation *instance processing procedure*, below, in that order.
E. If Halt-criterion is met, end the computation and output results: W, V, $\hat{u}$, e, and E. Otherwise, execute D again.

The best halting criterion, according to the nature of the steepest descent process should be at

(i) Matrices V and W stabilized. Unfortunately, in real world computations this criterion requires by far too many iterations, so that in practice the matrices fail to converge. Thus, other stopping criteria are used.
(ii) The difference between the average values (over iterations within an epoch) of the error function becomes smaller than a pre-specified threshold, such as 0.0001.
(iii) The number of epochs performed reaches a pre-specified threshold such as 5000.

Instance Processing Procedure

Specifics of the NN structure and function provide for simple and effective rules for processing individual entities in the procedure of the steepest descent. Before updating the wiring weights according to Eqs. (3.23), two following steps are executed:

1. *Forward computation of the estimated output and its error*. Upon receiving a training instance input feature values, they are processed by the neural network to produce an estimate of the output, after which the error is computed as the difference between real and estimated output values.
2. *Error back-propagation for estimation of the gradient.* The computed error of the output is back-propagated through the network. Each neuron of the output layer corresponds to a specific output feature and, thus, receives the error in this feature. Each neuron of the hidden layer receives a combined error signal from all output neurons weighted by the corresponding wiring weights. These are used to adjust the gradient elements by using the hidden neuron activation function.

In the Appendix A.4, a Matlab code nnn.m is presented for learning NN weights with the error back-propagation algorithm according to the NN of Fig. 3.22. Two parameters of the algorithm, the number of neurons in the hidden layer and the learning rate, are its input parameters. The output is the minimum level of error achieved and the corresponding weight matrices V and W.

The code includes the following steps:

1. *Loading data*. It is assumed that all data are in subfolder Data. According to the task, this can be either iris.dat or stud.dat or any other dataset.
2. *Normalizing data.* This is done by shifting each column to its midrange with the follow-up dividing it by the half-range, after which all data set is multiplied by 10, to have them in [−10,10] scale as described above.
3. *Preparing input and output* training sub-matrices. This is done after the decision has been made of what features fall in the former and what features fall in the latter categories. In the case of Iris data, for example, the target is petal data (features w3 and w4) and input is sepal measurements (features w1 and w2) as described. In the case of Students data, the target can be students' marks on all three subjects (CI, SP and OOP), whereas the other variables (occupation categories, age and number of children), input.
4. *Initializing the network.* This is done by: (a) specifying the number of hidden neurons H, (b) filling in matrices W and V with random (0,1) normally distributed values, and (c) setting a loop over epochs with the counter initialized at zero.
5. *Organizing a loop over the entities.* For setting a random order of entities to be processed, the Matlab command randperm(n) for making a random permutation of integers 1, 2,…, n can be used.
6. *Forward pass.* Given an entity, the output is calculated, as well as the error, using the current V, W and activation functions. The program uses the symmetric sigmoid (3.21) as the activation function of hidden neurons.
7. *Error back-propagation*. Gradient matrices for V and W according to formulas (3.26) and (3.27) are computed.
8. *Weights V and W update*. Having the gradients computed and learning rate accepted as the input, updated W and V are computed according to (3.23).
9. *Halt-condition*. This includes both the level of precision, say 0.01, and a threshold to the number of epochs, say, 5,000. If either is reached, the program halts.

Q.3.13. Prove that the derivatives of sigmoid (3.21) or hyperbolic tangent (3.21) functions appear to be simple polynomials of selves. Specifically,

$$s'(x) = \left((1+e^{-x})^{-1}\right)' = (-1)(1+e^{-x})^{-2}(-1)e^{-x} = s(x)(1-s(x)), \quad (3.28)$$

$$\begin{aligned} th'(x) &= [2*s(x)-1]' = 2*s(x)' = 2*s(x)*(1-s(x)) \\ &= (1+th(x))*(1-th(x))/2 \end{aligned} \quad (3.28')$$

Q.3.14. Find a way to improve the convergence of the process, for instance, with adaptive changes in the step size values.

Q.3.15. Use k-fold cross validation to provide estimates of variation of the results regarding the data change.
Q.3.16. Develop a scoring function for learning a category by using the contribution of the partition to be built to the category.

3.6 Association Between Nominal Features: Elementary and Linear Modeling

3.6.1 Elementary Analysis: Quetelet Index and Chi-Squared

3.6.1.1 Conceptual Relations from Statistics

To analyze interrelations between two nominal features, they are cross-classified in the so-called contingency table. A contingency table has its rows corresponding to categories of one feature and columns to categories of the other feature, while the entries are counts of entities falling in the overlap of the corresponding row and column categories. The contingency table's structure may well serve for the analysis of association between the nominal features summarized in the table.

Worked Example 3.8. Contingency Table on Market Towns Data
To cross-classify features Banks and Farmer's Market on Market towns data, we first need to categorize the quantitative feature Banks. Consider, for example, the four-category partition of the range of Banks feature at Market towns set presented in Table 3.14.

These categories are cross-classified with FM "yes" and "no" categories in Table 3.15. Besides the cross-classification counts, the table also contains summary within category counts, the totals, on the margins of the table, the last row and last

Table 3.14 Definition of Ba categories on the Market towns dataset

Category	Definition	Notation
1	Ba ≥ 10	10+
2	$10>$Ba ≥ 4	4+
3	$4>$Ba ≥ 2	2+
4	Ba = 0 or 1	1−

Table 3.15 Cross classification of the Ba categories with FM categories

FarmMarket	Bank/building society categories				Total
	10+	4+	2+	1−	
Yes	2	5	1	1	9
No	4	7	13	12	36
Total	6	12	14	13	45

Table 3.16 BA/FM cross-classification relative frequencies, percent

FM | Ba	10+	4+	2+	1−	Total
Yes	3.44	11.11	3.22	3.22	20
No	8.89	15.56	28.89	26.67	80
Total	13.33	26.67	31.11	28.89	100

Table 3.17 Protocol/attack contingency table for Intrusion data

Category	Apache	Saint	Smurf	Norm	Total
Tcp	23	11	0	30	64
Udp	0	0	0	26	26
Icmp	0	0	10	0	10
Total	23	11	10	56	100

column—this is why they are referred to as marginal frequencies. The total count balances the sheet in the bottom-right corner.

The same contingency data converted to relative frequencies by relating them to the total number of entities are presented in Table 3.16.

Q.3.17. Build a contingency table for features "Protocol-type" and "Attack type" in Intrusion data. **A.** See Table 3.17

A contingency table can be used for assessment of correlation between two sets of categories. The highest level of correlation is that of a conceptual association. A conceptual association may exist if a row, k, has all its entries, not marginal of course, except just one, say l, equal to 0, which would mean that all of the extent of category k belongs to the column category l. The data, thus, indicate that the category k may logically imply the category l.

Worked Example 3.9. Equivalence and Implication from a Contingency Table

Such are rows "Udp" and "Icmp" in Table 3.17. There is a perfect match in this table: a row category k = "Icmp" and a column category l = "Smurf", that contains the only non-zero count. No other combination (k, l′) or (k′, l) is possible according to the table. In such a situation, one may claim that, subject to the sampling error, category l may occur if and only if k does, that is, k and l are equivalent.

A somewhat weaker, but still very much valuable is the case of "Udp" row in Table 3.17. It appears, Udp protocol implies "Norm" column category—a no-attack situation, though there is no equivalence here because the "Norm" column contains another positive count, in row "Tcp".

Case-Study 3.5. Trimming Contingency Data: A Bad Option

Unfortunately, there are no zeros in Table 3.15, and thus, no conceptual relation between the number of Banks and the presence of a Farmer's market. But some of the entries are really close to 0, which may make us tempted to trim the data a bit. Imagine, for example, that in row "Yes" of Table 3.15, two last entries are 0, not 1s.

Table 3.18 A trimmed BA/FM cross classification "cleaned" of 13 towns, to sharpen the view

	Number of banks/build, societies				
FMarket	10+	4+	2+	1−	Total
Yes	2	5	0	0	7
No	0	0	13	12	25
Total	2	5	13	12	32

This would imply that a Farmers Market may occur only in a town with 4 or more Banks. A logical implication, that is, a production rule, "If BA is 4 or more, then a Farmer's market must be present", could be derived then from thus modified table. One may try taking this path and cleaning the data of smaller entries, by removing corresponding entities from the table, of course, to not obscure our "vision" of the pattern of correlation. Thus trimmed Table 3.18 is obtained from Table 3.15 by removing just 13 entities from "less popular" entries. This latter table expresses, with no exception, a very simple conceptual statement "A town has a Farmer's market if and only if the number of Banks in it is 4 or greater". However nice the rule may sound, let us not forget the cost of the trimming which is the 13 towns, almost 30% of the sample, that have been removed as those not fitting the stated perspective. Such a data doctoring borders with forgery—one of the reasons for a famous quip usually attributed to B. Disraeli, a celebrated British politician of XIX century: "There are three gradations of lies: lies, damned lies and statistics." The issue of sample adjustment so far has received no reasonable solution, even with respect to outliers—values falling way beyond the feature range one would expect normally. Anyway, the conclusion of the trimming exercise is that one should try finding ways of expressing conceptual relations without doctoring the sample.

3.6.1.2 Capturing Relationships with Quetelet Indexes

Quetelet index provides for a strategy for visualization of correlation patterns in contingency tables without removal of "not-fitting" entities. In 1832, A. Quetelet (1796–1874), a founding father of statistics, proposed to measure the extent of association between row and column categories in a contingency table by comparing the local count with an average one.

Let us consider correlation between the presence of a Farmer's Market and the category "10 or more Banks" according to Table 3.15. We can see that their joint probability/ frequency is the entry in the corresponding row and column: P(Ba = 10 + & FM = Yes) = 2/45 = 4.44% (joint probability/frequency rate). Of the 20% entities that fall in the row "Yes", this makes the proportion of "Ba = 10+" under condition "FM = Yes" equal to P(Ba = 10+ /FM = Yes) = P(Ba = 10+ & FM = Yes)/P(FM = Yes) = 0.0444/0.20 = 0.222 = 22.2%. Such a ratio expresses the conditional probability/rate.

Is this high or low? Hard to tell without comparing this with the unconditional rate, that is, with the frequency of category "Ba = 10+" in the whole dataset, which is P(Ba = 10+) = 13.33%. Let us compute the (relative) difference between the two, which is referred to as Quetelet index q:

$$\begin{aligned} q(\text{Ba} = 10+/\text{FM} = \text{Yes}) &= [\text{P}(\text{Ba} = 10+/\text{FM} = \text{Yes}) \\ &\quad -\text{P}(\text{Ba} = 10+)]/\text{P}(\text{Ba} = 10+) \\ &= [0.2222 - 0.1333]/0.1333 = 0.6667 = 66.7\%. \end{aligned}$$

That means that condition "FM = Yes" raises the frequency of the Bank category by 66.7%.

This logic concurs with our everyday intuition. Consider, for example, the risk of getting a serious illness, say tuberculosis, which may be, say, about 0.1%, one in a thousand, in a given region. Take a condition such as "Bad housing" and count the rate of tuberculosis under this condition, amounting to, say 0.5%—which is very small by itself, yet a five-fold increase over the average tuberculosis rate. This is exactly what Quetelet index measures: $q(l/k) = (0.5 - 0.1)/0.1 = 400\%$ to show that the change of the average rate is 4 times.

Worked Example 3.10. Quetelet Index in a Contingency Table
Let us apply the general Quetelet index formula (3.29) to entries in Table 3.15. This leads to Quetelet index values presented in Table 3.19. By highlighting positive values in the table, we obtain the same pattern as on the "purified" data as in Case-Study 3.5, but this time in a somewhat more realistic manner, keeping the sample intact. Specifically, one can see that "Yes" FM category provides for a strong increase in the probabilities, whereas "No" category leads to much weaker changes.

Table 3.19 BA/FM Cross classification Quetelet coefficients, % (positive entries highlighted using bold font)

FMarket	10+	4+	2+	1−
Yes	**66.67**	**108.33**	−63.29	−61.54
No	−16.67	−27.08	**16.07**	**15.38**

Q.3.18. Compute Quetelet coefficients for Table 3.17.
A. See Table 3.20 in which positive entries are highlighted in bold

Table 3.20 Quetelet indices for the protocol/attack contingency Table 3.17, %; positive values are highlighted using bold font

Category	Apache	Saint	Surf	Norm
Tcp	**56.25**	**56.25**	−100.00	−16.29
Udp	−100.00	−100.00	−100.00	**78.57**
Icmp	−100.00	−100.00	**900.00**	−100.00

Case-Study 3.6. Has There Been Any Bias in S'n'S' Policy?
Take on the case of Stop-and-Search policy in England and Wales 2005 represented according to race (B—black, A—Asian and W—white), by numbers in Table 1.12 in Sect. 1.3.2—these are overwhelmingly in category W. The criticism of this policy came out of comparison of this distribution with the distribution of the entire population. Such a distribution, according to the latest pre-2005 census 2001, can be easily found on web. By subtracting from that the numbers of Stop-and-Search occurrences, under the assumption that nobody has been subjected to this more than once, Table 3.21 has been drawn. Its last column gives the numbers that were used for the claim of a racial bias: indeed category B members have been subjects of the policy six times more frequently than category W members. A similar picture emerges when Quetelet coefficients are used (see Table 3.22). Category B is subject to Stop-and-Search policy 400% more frequently than on average, whereas category W, 15% less.

Yet some would consider drawing a table like Table 3.21, and of course the derived Table 3.22, as something nonsensical, because it is based on an implicit assumption that the Stop-and-Search policy applies to the population randomly. They would argue that police apply the policy only when they deem it necessary, so that the comparison should involve not all of the total population but only that criminal. Indeed, the distribution of subjects to Stop-and-Search policy by race has been almost identical to that of the imprisoned population of the same year. Therefore, the claim of a racial bias by police should be declared incorrect, provided that the system of justice is not biased overall.

Table 3.21 Distribution of stop-and-search policy cross-classified with race

	S'n'S	Not S'n'S	Total	S'n'S-to-Total
Black	131,723	1,377,493	1,509,216	0.0873
Asian	70,252	2,948,179	3,018,431	0.0233
White	676,178	46,838,091	47,514,269	0.0142
Total	878,153	51,163,763	52,041,916	0.0169

Table 3.22 Relative Quetelet coefficients for cross-classification in Table 3.21, %

	S'n'S	Not S'n'S
Black	417.2	−7.2
Asian	37.9	−0.6
White	−15.7	0.3

3.6.1.3 Conditional Probabilities, Quetelet Indexes and Pearson's Chi-Squared

Consider two sets of disjoint categories on an entity set I, K and L. Using $k = 1,\ldots, K$ (for example, occupation of individuals constituting I) and $l = 1,\ldots, L$ (say, family or housing type) as category indices should not bring any confusion here. Both, K and L, make a partition of the entity set I; these partitions are crossed to see if there is any correlation between them. For a a pair of categories $(k,l) \in K \times L$, count the number of entities that fall in both. The (k,l) co-occurrence count is denoted by N_{kl}. Obviously, these counts sum to N because the categories are not overlapping and cover the entire dataset. A table housing these counts, N_{kl}, or their relative values, relative frequencies $p_{kl} = N_{kl}/N$, is referred to as a contingency table or just cross-classification. The totals, that is, within-row sums $N_{k+} = \Sigma_l N_{kl}$ and within-column sums $N_{+l} = \Sigma_k N_{kl}$ (as well as their relative frequency counterparts) are referred to as marginals (because they are located on the margin of the contingency table).

The (empirical) probability that category l occurs under condition k can be expressed as $P(l/k) = p_{kl}/p_{k+} = N_{kl}/N_{k+}$. The probability $P(l)$ of the category l with no condition is just $p_{+l} = N_{+l}/N$. Similar notation is used when l and k are swapped. The relative difference between the conditional and unconditional probabilities is referred to as the (relative) Quetelet index (Mirkin 2001):

$$q(l/k) = \frac{P(l/k) - P(l)}{P(l)} \tag{3.29}$$

where $P(l) = N_{+l}/N$, $P(k) = N_{k+}/N$, $P(l/k) = N_{kl}/N_{k+}$. That is, Quetelet index expresses correlation between categories k and l as the relative change in the probability of l when k is taken into account.

With little algebra, one can derive a simpler—and symmetric—expression

$$\begin{aligned} q(l/k) &= [N_{kl}/N_{k+} - N_{+l}/N]/(N_{+l}/N) \\ &= N_{kl}N/(N_{k+}N_{+l}) - 1 = \frac{p_{kl}}{p_{k+}p_{+l}} - 1 \end{aligned} \tag{3.29'}$$

Highlighting high positive and negative values in a Quetelet index table, such as Tables 3.19 and 3.22, visualizes the pattern of association between the two sets of categories.

This visualization can be extended to a theoretically sound presentation. Let us define the average Quetelet association index Q as the sum of pair-wise Quetelet indexes weighted by their frequencies/probabilities:

$$Q = \sum_{k=1}^{K}\sum_{l=1}^{L} p_{kl} q(l,k) = \sum_{k=1}^{K}\sum_{l=1}^{L} p_{kl}\left(\frac{p_{kl}}{p_{k+}p_{+l}} - 1\right) = \sum_{k=1}^{K}\sum_{l=1}^{L} \frac{p_{kl}^2}{p_{k+}p_{+l}} - 1 \tag{3.30}$$

The right-hand expression for Q in (3.30) is popular in the statistical analysis of contingency data. In fact, this Q is equal to the chi-squared correlation coefficient proposed by Pearson (1900) in a very different context—as a measure of deviation of the contingency table entries from the statistical independence.

To explain this in more detail, let us first introduce the concept of statistical independence. The sets of k and l categories are said to be statistically independent if $p_{kl} = p_{k+}\, p_{+l}$ for all k and l. Obviously, such a condition is hard to fulfill in reality K. Pearson suggested using relative squared errors to measure the deviations of observed frequencies from the statistical independence. Specifically, he introduced the following coefficient usually referred to as Pearson's chi-squared association coefficient:

$$X^2 = N \sum_{k=1}^{K} \sum_{l=1}^{L} \frac{(p_{kl} - p_{k+} p_{+l})^2}{p_{k+} p_{+l}} = N \left(\sum_{k=1}^{K} \sum_{l=1}^{L} \frac{p_{kl}^2}{p_{k+} p_{+l}} - 1 \right) \tag{3.31}$$

The equation on the right can be proven with little algebra. Consider, for example, this part of the expression on the left in (3.31):

$$\sum_{l=1}^{L} \frac{(p_{kl} - p_{k+} p_{+l})^2}{p_{k+} p_{+l}} = \sum_{l=1}^{L} \frac{p_{kl}^2 - 2 p_{kl} p_{k+} p_{+l} + (p_{k+} p_{+l})^2}{p_{k+} p_{+l}}$$
$$= \sum_{l=1}^{L} \frac{p_{kl}^2}{p_{k+} p_{+l}} - 2 \sum_{l=1}^{L} p_{kl} + \sum_{l=1}^{L} p_{k+} p_{+l} = \sum_{l=1}^{L} \frac{p_{kl}^2}{p_{k+} p_{+l}} - p_{k+}$$

The expression on the right in the above is derived by using equations $\Sigma_l p_{kl} = p_{k+}$ and $\Sigma_l p_{+l} = 1$. Summing all these equations over k will produce (3.31). On the other hand, the expression on the right in (3.31) is obviously equal to $\Sigma_l p_{kl} q(l/k)$ so that

$$\sum_{l=1}^{L} \frac{(p_{kl} - p_{k+} p_{+l})^2}{p_{k+} p_{+l}} = \sum_{l=1}^{L} p_{kl} q(l/k) \tag{3.32}$$

By comparing the right-hand parts of (3.30) and (3.31), it is easy to see that $X^2 = NQ$. The same follows from summing all the equations (3.32) over k.

The popularity of X^2 index in statistics and related fields rests on the following theorem proven by K. Pearson: if the contingency table is based on a sample of entities independently drawn from a population in which the statistical independence holds (so that all deviations are due to just randomness in the sampling), then the probabilistic distribution of X^2 converges to the chi-squared distribution (when N tends to infinity) introduced by Pearson earlier for similar analyses. The probabilistic chi-squared distribution (with p degrees of freedom) is defined as the distribution of the sum of squares of p random variables, each distributed according to the standard Gaussian $N(0,1)$ distribution.

This theorem is not always of interest to a computational data analyst, because they analyze data that are not necessarily random or not necessarily independently sampled. However, Pearson's chi-squared coefficient is frequently used just for scoring correlation in contingency tables. The equation $X^2 = NQ$ gives a credible support to this practice. According to this equation, the value of X^2 is not only a measure of deviation from the statistical independence. It also has a different meaning as a measure of association between categories: that of the averaged Quetelet coefficient. If, for example $X^2/N = 0.25$, one may credibly claim that the knowledge of L-category at an object improves the chance of its corresponding K-category by 25% on average.

To get more intuition on the underlying correlation concept, let us take a look at the extreme values that X^2 can take and situations at which the extreme values are reached (Mirkin 2001). It appears that at $K \leq L$, that is, if the number of columns is not smaller than that of rows, X^2 ranges between 0 and $K - 1$. It reaches 0 if there is a statistical independence at all (k,l) entries so that all $q_{kl} = 0$, and it reaches $K - 1$ if each column l contains only one non-zero entry $p_{k(l)l}$, which is thus equal to p_{+l}. Such a structure of the contingency table can be interpreted as an empirical evidence that the logical implication $l(k) \rightarrow k$ has place for all $k = 1,2,\ldots, K$.

Representation of the chi-squared through Quetelet coefficients,

$$X^2 = \sum_{k=1}^{K}\sum_{l=1}^{L} Np_{kl}q(l/k) \tag{3.33}$$

amounts to decomposition of X^2 into the sum of $N_{kl}\ q(l/k)$ items and allows for visualization of the items within the contingency table format, such as that presented in Table 3.23.

In fact, not only the total sum of these items coincide with that of the original chi-squared items $N(p_{kl} - p_{k+}p_{+l})^2/p_{k+}p_{+l}$, but also the within-column and within-row sums coincide too, as the derivation of (3.32) above clearly demonstrates for the latter case.

However all the original chi-squared items in (3.31) are positive and cannot show whether the correlation expressed by an individual entry is positive or negative. To overcome this shortcoming, another visualization of X^2 is in use. That visualization involves the square roots of the chi-squared items

$$r(k,l) = \frac{p_{kl} - p_{k+}p_{+l}}{\sqrt{p_{k+}p_{+l}}} \tag{3.34}$$

Table 3.23 Four-fold contingency table between binary features

		Feature Y		Total
		Yes	No	
Feature X	Yes	a	b	$a + b$
	No	c	d	$c + d$
Total		$a + c$	$b + d$	$N = a + b+c + d$

that are convenient to refer to as Pearson indexes. Obviously, $X^2 = N\Sigma_{k,l} r(k,l)^2$. Pearson indexes indeed have the same signs as $q(l/k)$, and in fact are closely related: $q(l/k) = r(k,l)[(p_{k+}p_{+l})]^{1/3}$. It is less clear what interpretation of its own the values $r(k,l)$ may have, although they are useful in the Correspondence analysis of contingency tables (Sect. 3.6.3), see also normalized Laplacian in Sect. 5.2.

Q.3.19. Take two binary features presented as 1/0 variables and build their contingency table, sometimes referred to as a four-fold table (Table 3.23) in which symbols a, b, c, and d are used to denote the co-occurrence counts.

Prove that Quetelet coefficient $q(Yes/Yes)$ expressing the relative difference between $a/(a + c)$ and $(a + b)/N$ is equal to

$$q(Yes/Yes) = \frac{ad - bc}{(a+c)(a+b)},$$

and the summary Quetelet coefficient Q, or Pearson's X^2/N, is equal to

$$Q = \frac{(ad - bc)^2}{(a+c)(b+d)(a+b)(c+d)}.$$

Q.3.20. Prove that the correlation coefficient between two 1/0 binary features can be expressed in terms of the four-fold table as $\rho = \sqrt{Q}$, that is,

$$\rho = \frac{ad - bc}{\sqrt{(a+c)(b+d)(a+b)(c+d)}}.$$

Q.3.21. Given a $K \times L$ contingency table P and a pair of categories, $k \in K$ and $l \in L$, consider an absolute Quetelet index $a(l/k) = P(l/k) - P(l)$ − the change from the frequency of $l \in L$ on the whole entity set I to the frequency of l on entities falling in category $k \in K$. In terms of P, $P(l) = p_{+l}$ and $P(l/k) = p_{kl}/p_{+l}$. Prove that the summary Quetelet index $A = \Sigma_{k,l} p_{kl} a(l/k) = \Sigma_{k,l} p_{kl}^2/p_{k+} - \Sigma_l p_{+l}^2$ is equal to the following expression, an asymmetric analogue to Pearson chi-squared:

$$A = \sum_{k=1}^{K} \sum_{l=1}^{L} \frac{(p_{kl} - p_{k+}p_{+l})^2}{p_{k+}} \tag{3.35}$$

which also is the numerator of the so called Goodman-Kruskal "tau-b" association index (Kendall and Stewart 1967).

A. Indeed, by taking the square of the numerator, expression in (3.35) becomes equal to $\Sigma_{k,l}(p_{kl}^2 - 2p_{kl}p_{k+}p_{+l} + p_{k+}^2 p_{+l}^2)/p_{k+}$, which is $\Sigma_{k,l}p_{kl}^2/p_{k+} - 2\Sigma_{k,l}p_{kl}p_{+l} + \Sigma_{k,l}p_{k+}p_{+l}^2 = \Sigma_{k,l}p_{kl}^2/p_{k+} - 2\Sigma_{k,l}p_{+l}^2 + \Sigma_l p_{+l}^2$ because $\Sigma_k p_{kl} = p_{+l}$ and $\Sigma_k p_{k+} = 1$. This is obviously $\Sigma_{k,l}p_{kl}^2/p_{k+} - \Sigma_l p_{+l}^2 = \Sigma_{k,l}p_{kl}a(l/k) = A$, which proves the statement.

Worked Example 3.11. Visualization of Contingency Table Using Weighted Quetelet Coefficients

Let us multiply Quetelet coefficients in Table 3.19 by the frequencies of the corresponding entries in Table 3.15. Quetelet coefficients in Table 3.19 are taken relative to unity, not per cent. This leads us to Table 3.24 whose entries sum to the value of Pearson's chi-square coefficient for Table 3.15, 6.86. Note that entries in Table 3.24 can be both positive and negative; those with absolute value greater than 6.86/4 = 1.72 are highlighted in bold—they show the entries of an extraordinary deviation from the average. Of them, column 4+ supplies the highest positive impact and the highest negative impact.

Worked Example 3.12. A Conventional Decomposition of Chi-square Coefficient

Let us consider a conventional way of visualization of contingency tables, by putting Pearson indexes, the square roots $r(k,l)$ of the chi-square coefficient items in (3.34) as the table's elements. These are in Table 3.25. The table does show a similar pattern of positive and negative associations. However, it is not the entries of the table that sum to the chi-square coefficient but rather the squares of the entries. The fact that the summary values on the margins in Tables 3.24 and 3.25 are the same is not by chance: it exemplifies a mathematical property (see Eq. (3.32)).

Table 3.24 BA/FM chi-squared (NQ = 6.86) and its decomposition according to (3.33); extreme values are highlighted using bold font

FMarket	10+	4+	2+	1−	Total
Yes	1.33	**5.41**	−0.64	−0.62	5.48
No	−0.67	**−1.90**	**3.09**	**1.85**	1.37
Total	0.67	3.51	1.45	1.23	**6.86**

Table 3.25 Square roots of the items in Pearson chi-squared (X^2 = 6.86); the items themselves are in parentheses; those positive are highlighted using bold font

FMarket	10+	4+	2+	1-	Total
Yes	**0.73**(0.53)	**1.68(3.82)**	−1.08 (1.16)	−0.99 (0.98)	(5.49)
No	−0.36 (0.13)	−0.84 (0.70)	**0.54 (0.29)**	**0.50 (0.25)**	(1.37)
Total	(0.67)	(3.52)	(1.45)	(1.23)	(6.86)

Q.3.22. In Table 3.24, all marginal values, the sums of rows and columns, are positive, in spite of the fact that many within-table entries are negative. Is this just due to specifics of the distribution in Table 3.15 or a general property?
A. A general property: the within-row or within-column sums of the elements, $N_{lk}\ q(l/k)$, must be positive, see (3.32).
Q.3.23. Find a similar decomposition of chi-squared for OOPmarks/Occupation in Student data. Hint: First, categorize quantitative feature OOPmarks somehow: you may use equal bins, or conventional boundary age points such as 35, 65 and 75, or any other considerations.
Q.3.24. Can any logical production rules come from the columns of Table 3.17?
A. Yes, both Apache and Saint attacks may occur at the tcp protocol only.
Q.3.25. Of 100 Christmas shoppers in Q.2.25, 50 spent £60 each, 20 spent £100 each, and 30 spent £150 each. Those who spent £60 each are males only and those who spent £100 each are females only, whereas among the rest 30 individuals half are men and half are women. Build a contingency table for the two features, gender and spending. Find and interpret the value of Quetelet coefficient for females who spent £100 each.
A. The contingency table (of co-occurrence counts):

	Spending, £			
Gender	60	100	150	Total
Female	0	20	15	35
Male	50	0	15	65
Total	50	20	30	100

This table of absolute co-occurrence counts coincides with that of proportions expressed per cent because the number of shoppers is 100.

Quetelet coefficient for (Female/£100) entry is

$$Q = 100 * 20/(20 * 35) - 1 = 3.86 - 1 = 1.86$$

This means that being female in this category of spending is more likely than the average, by 186%.

3.6.1.4 Different Association Measures

Given two nominal features represented by partitions $S = \{S_1, S_2,..., S_K\}$ and $T = \{T_1, T_2,...., T_L\}$ of the entity set I, summarized in a $K \times L$ contingency table $P = (p_{kl})$ where p_{kl} is the proportion of entities in $S_k \cap T_l$. Let us describe a few approaches to scoring the association between S and T that are used in popular data analysis programs.

One idea for assessing the extent of association is to use a correlation measure over the contingency table entries, such as averaged Quetelet coefficients, Q and A, or chi-squared X^2, as discussed above in Sect. 3.6.1.

Seemingly another idea is to score the extent of reduction of uncertainty over T obtained when S becomes available. This idea works like this: take a measure of uncertainty of a feature, in this case partition, $\upsilon(T)$, and evaluate it within each of S-classes, $\upsilon(T(S_k))$, $k = 1,\ldots, K$. Then the average uncertainty on these classes will be $\sum_{k=1}^{K} p_{k+}\upsilon(T(S_k))$, where p_{k+} are proportions of categories k, that is, classes S_k, so that the total reduction of uncertainty is equal to

$$\upsilon(T/S) = \upsilon(T) - \sum_{k=1}^{K} p_{k+}\upsilon(T(S_k)) \tag{3.36}$$

Of course, a function like (3.36) can be considered a measure of association over the contingency table P as well, but a nice feature of this approach is that it can be extended from nominal features to quantitative ones—by using an uncertainty index over quantitative T-features.

Two very popular measures defined according to (3.36) are the so-called impurity function (Breiman et al. 1984) and information gain (Quinlan 1993).

The impurity function builds on Gini coefficient as a measure of variance (see Sect. 2.3.1). Let us recall that Gini index for partition T is $G(T) = 1 - \sum_{l=1}^{L} p_l^2$ where p_l is the proportion of entities in T_l. If I is partitioned in clusters S_k, $k = 1,\ldots, K$, partitions T and S form a contingency table of relative frequencies $P = (p_{kl})$. Then the reduction (3.36) of the value of Gini coefficient due to partition S is equal to

$$\Delta(T, S) = G(T) - \sum_{k=1}^{K} p_{k+}G(T(S_k)). \tag{3.37}$$

This index $\Delta(T, S)$ is referred to as impurity of S over partition T. The greater the impurity, the better the split S.

It is not difficult to prove that $\Delta(T, S)$ relates to Quetelet indexes from Sect. 3.6.1. Indeed, $\Delta(T, S) = A(T, S)$ where $A(T, S)$ is the average absolute Quetelet index defined by Eq. (3.35) in Q.3.21. This implies indeed that $\Delta(T, S) = \sum_l p_{kl}^2/p_{k+} - \sum_l p_{+l}^2$, which proves the following statement: The impurity function is equal to the average absolute Quetelet index.

The information gain function builds on entropy as a measure of uncertainty (see Sect. 2.3.1). Let us recall that entropy of partition T is $H(T) = -\sum_{l=1}^{L} p_l \log(p_l)$ where p_l is the proportion of category l, that is, part T_l. If the entity set is partitioned in clusters S_k, $k = 1,\ldots, K$, partitions T and S form a contingency table of relative frequencies $P = (p_{kl})$. Then the reduction (3.36) of the value of entropy due to partition S is equal to $I(T, S) = H(T) - \sum_{k=1}^{K} p_{k+}H(T(S_k))$. This index $I(T, S)$ is referred to as the information gain due to S. In fact, it is equal to a popular characteristic of the cross-classification of T and S, the mutual information defined as $I(T, S) = H(T) + H(S) - H(ST)$ where $H(ST)$ is entropy of the bivariate distribution represented by the contingency table P. Please note that the mutual information is symmetric with regard to S and T, in contrast to the impurity function.

To prove the statement let us just put forward the definition of the information gain and use the property of logarithm that $\log(a/b) = \log(a) - \log(b)$:

$$I(T,S) = H(T) - \sum_k p_k H(T(S_k)) = H(T) + \sum_k p_{k+} \sum_l p_{kl} \log(p_{kl}/p_{k+})$$
$$= H(T) - \sum_k p_{k+} \log(p_{k+}) + \sum_{k,l} p_{kl} \log(p_{kl}) = H(T) + H(S) - H(ST),$$

which completes the proof.

The reduction of uncertainty measures are absolute differences that much depend on the measurement scale and, also, on values of $\upsilon(T)$ and $\upsilon(S)$. This is why it can be of advantage to use relative versions of the reduction of uncertainty measures normalized by $\upsilon(T)$ or $\upsilon(S)$ or both. For example, the popular program C4.5 (Quinlan 1993) uses the information gain normalized by $H(S)$ and referred to as the information gain ratio.

3.6.2 *Least-Squares Analysis of Association Between Dummy Matrices*

3.6.2.1 Linear Regression of One Dummy Matrix Over the Other One

Consider a nominal feature over an entity set I of cardinality N represented by partition $T = \{T_l\}$, and another nominal feature represented by partition $S = \{S_k\}$. This can be a cluster partition derived from available features to approximate T. Rather than directly concentrating on their contingency table $P = (p_{kl})$, let us take a look at the association between T and S from a different perspective.

Let us define an $N \times L$ dummy matrix X corresponding to partition T by assigning each category T_l with a binary variable x_l, a dummy, which is just a 1/0 N-dimensional vector whose elements $x_{il} = 1$ if i $\in T_l$ and $x_{il} = 0$, otherwise ($l = 1,\ldots, L$). Similarly define an $N \times K$ dummy matrix Y whose columns y_k are 0/1-vectors corresponding to categories k of S.

Let us consider linear regression of y_k over set of all X-categories which is achieved by using the orthogonal projector $P_X = X(X^TX)^{-1}X^T$, so that $\hat{y}_k = P_X y_k$. Let us take a look at the components of the computed vector $\hat{y}_k$. Let us recall that the projector's matrix P_X consists of diagonal blocks with (i,j)-th elements $1/N_l$ for $i,j \in T_l$. whereas all the other elements are zero. Then the inner product of its i-th row and vector is the number of unities in it multiplied by $1/N_l$ for that l for which $i \in T_l$. That is exactly N_{lk}/N_l where N_{lk} is the cardinality of intersection $S_k \cap T_l$. In other words, i-th element of $\hat{y}_k$ is the conditional probability $N_{lk}/N_l = p(k/l)$ where l is the T-category of object i, $i = 1,2,\ldots, N$.

This gives the structure of regression of the dummy matrix Y over dummy matrix X, $\hat{Y} = P_X Y$ in terms of the conditional frequency contingency table.

Let us now standardize each feature y_k by a scale shift a_k and rescaling factor $1/b_k$, according to the conventional formula $y'_k = (y_k - a_k)/b_k$. This will correspondingly change the $\hat{Y}$-values, so that the conditional probabilities will change to $c_{kl} = (p(k/l) - a_k)/b_k$. In mathematical statistics, the issue of standardization is just a routine transforming the probabilistic density function to a standardized format. Things are different in data analysis, since no density function is assigned to data usually. The scale shift is considered as positioning the data against a backdrop of the "norm", whereas the act of rescaling is to balance feature "weights" (see Sect. 2.2 for discussion). Therefore, choosing the feature means as the 'norm" should be reasonable. The mean of feature y_k is obviously the proportion of unities in it, which is p_{k+} in notations related to the contingency table P. In fact, the remainder of this section can be considered as another reason for using $a_k = p_{k+}$. The choice of rescaling factors is somewhat less certain, though using all $b_k = 1$ should seem reasonable too because all the dummies are just 1/0 variables measured in the same scale. Incidentally, 1 is the range of any dummy. Some other values related to y_k's dispersion could be used as well. Especially suitable *is* $b_k = (p_{k+})^{1/2}$ which is the standard deviation of the Poisson distribution of a 1/0 variable (see Sect. 2.3.3), as will be seen in the end of this section. With the scale shift value specified, the standardized conditional probabilities in $\hat{Y}$ can be expressed as

$$p'(k/l) = \frac{p_{lk} - p_{k+}p_{+l}}{p_{+l}b_k} \tag{3.38}$$

Let us compute the sum of squares of all the $\hat{Y}$-elements (3.38). Within the k-th column, there are N_{+l} values p(k/l) (3.38), which leads us to the value

$$B_{lk} = N_{+l}p'(k/l)^2 = N\frac{(p_{lk} - p_{k+}p_{+l})^2}{b_k^2 p_{+l}} \tag{3.39}$$

as the contribution of (k,l)-pair to the sum of squares, $\langle \hat{Y}', \hat{Y}' \rangle$, where symbol '" refers to the fact that the data matrix Y has been pre-standardized.

Accordingly, the total contribution of partition S to the total scatter of the set of standardized dummies representing partition T is equal to

$$B(S,T) = \left\langle \widehat{\mathrm{Y}}', \widehat{\mathrm{Y}}' \right\rangle = \sum_{l=1}^{L}\sum_{k=1}^{K} B_{lk}^2 = N\sum_{l=1}^{L}\sum_{k=1}^{K}\frac{(p_{lk} - p_{k+}p_{+l})^2}{b_k^2 p_{+l}} \tag{3.40}$$

The term "contribution" comes from the regression model under consideration:

$$\mathrm{Y}' = \mathrm{P_X Y} + \mathrm{E} = \widehat{\mathrm{Y}}' + \mathrm{E},$$

which satisfies the Pythagorean decomposition property because of the orthogonality of the projector $\mathrm{P_X}$:

$$\langle \mathrm{Y}', \mathrm{Y}' \rangle = \langle \widehat{\mathrm{Y}}', \widehat{\mathrm{Y}}' \rangle + \langle \mathrm{E}, \mathrm{E} \rangle,$$

so that $\langle \widehat{\mathrm{Y}}', \widehat{\mathrm{Y}}' \rangle$ is the contribution indeed.

It should be noted that the value $\langle \mathrm{E}, \mathrm{E} \rangle$ can be considered a measure of distance, Euclidean squared, as usual, between partitions T and S corresponding to the nominal features and their dummy matrices X and Y under consideration

$$\delta(\mathrm{X}, \mathrm{Y}) = \langle \mathrm{E}, \mathrm{E} \rangle = \| \mathrm{Y}' - \mathrm{P}_{\mathrm{X}} \mathrm{Y}' \|^2 \tag{3.41}$$

This can be referred to as the dummy matrix regression distance.

The total contribution (3.40) reminds us of both the averaged relative Quetelet coefficient Q in (3.30), as well as the impurity function Δ(T, S) in (3.37) and the averaged absolute Quetelet coefficient in (3.35). The latter two indexes, up to the constant N of course, emerge at the rescaling factors being unity, $b_k = 1$. The former emerges when rescaling factors $b_k = \sqrt{p_{k+}}$. The square root of the frequency has an appropriate meaning—this is a good estimate of the standard deviation in Poisson model of the variable: according to this model, N_{k+} unities are thrown randomly into the fragment of memory assigned for the storage of vector y_k. In fact, at this scaling system, $\mathrm{B(T/S)} = \mathrm{X}^2$, the Pearson chi-squared!

Let us summarize the proven facts.

Statement 3.6.2.1 The impurity function in (3.37) can be equivalently expressed as

(a) The reduction of Gini uncertainty index of partition T when partition S is taken into account;
(b) The averaged absolute Quetelet index $a(l/k) = p_{kl}/p_{k+} - p_{+l}$ of the same effect;
(c) The total contribution of partition S to the summary data scatter of the set of dummy 1/0 features corresponding to classes of T and standardized by subtracting the mean with no rescaling.

Statement 3.6.2.2 The Pearson chi-squared association index can be equivalently expressed as

(a) A measure of statistical independence between partitions T and S;
(b) The averaged relative Quetelet index $q(l/k) = (p_{kl}/p_{k+} - p_{+l})/p_{+l}$ between partitions T and S;
(c) The total contribution of partition S to the summary data scatter of the set of dummy 1/0 features x_l corresponding to classes T_l and standardized by subtracting the mean and dividing the result by $b_l = \sqrt{p_l}$.

The claims of equivalence in Statements 3.6.2.1 and 3.6.2.2, although having been described by this author earlier (see, for example Mirkin 1996, 2012) remain virtually unknown, in spite of the fact that they point to useful relations between different association measures and, as well, between statistical association measures and preferred data normalization options.

3.6.2.2 Canonical Analysis of Dummy Matrices: Dual Scaling, "L'analyse Des Correspondences" and the Chi-Squared

Consider two nominal features over an entity set I of cardinality N represented by partitions $T = \{T_l\}$ and $S = \{S_k\}$. Define $N \times L$ dummy matrix X and $N \times K$ dummy matrix Y corresponding to partitions T and S, respectively.

Consider the linear subspaces $L(X)$ and $L(Y)$ spanning matrices X and Y. The problem of canonical correlation is to find in $L(X)$ and $L(Y)$ normed vectors x and y maximally oriented towards each other so that $\langle x, y \rangle$ is maximum with respect to $x \in L(X)$ and $y \in L(Y)$ such that $\langle x, x \rangle = a^T X^T X a = 1$ and $\langle y, y \rangle = b^T Y^T Y b = 1$. In fact the problem is of finding vectors a and b maximizing $a^T X^T Y b$ such that $a^T X^T X a = 1$ and $b^T Y^T Y b = 1$. Since matrices $X^T X$ and $Y^T Y$ are diagonal with diagonal elements equal to N_{+l} and N_{k+}, respectively, the normalizing constraints can be reformulated as

$$\sum_{l=1}^{L} N_{+l} b_l^2 = 1, \quad \sum_{k=1}^{K} N_{k+} a_k^2 = 1,$$

A mathematical solution to this problem is described in Sect. A.3.3 of Appendix. It is related to spectral analysis of the product $P_X P_Y$ where P_X and P_Y are $N \times N$ orthogonal projector matrices, defined as $P_X = X(X^T X)^{-1} X^T$, $P_Y = Y(Y^T Y)^{-1} Y^T$. The general (i,j)-th element of $P_X P_Y$ is the inner product of i-th row of P_X and j-th row/column of P_Y. The former consists of $1/N_{+l}$ for all the objects belonging to the same l-th category of T as i, and zeros at other objects. The latter consists of $1/N_{k+}$ for all the objects belonging to the same k-th category of S as j, and zeros at other objects. Non-zero values meet at objects belonging to both T_{+l} and S_{k+}, that is, $S_k \cap T_l$. Therefore the inner product is equal to $N_{kl}/(N_{k+}N_{+l})$.

The spectrum of matrix $P_X P_Y$ is the same as of matrix AB where $A = (X^T X)^{-1} X^T Y$, $B = (Y^T Y)^{-1} Y^T X$ as defined in Sect. A.3.3. The matrices A and B are of sizes $L \times K$ and $K \times L$, respectively, and their entries are conditional probabilities. It is easy to find out that (l,k)-th element of A is equal to N_{lk}/N_{+l}, and (k,l)-th element of B is equal to N_{kl}/N_{k+}. Here $N_{lk} = N_{kl}$ is the number of objects in the intersection $S_k \cap T_l$.

A version of the power method for finding the maximum eigenvalue of $P_X P_Y$ or of AB, which is the same, may be defined by using matrices A and B separately. The method begins with any K-dimensional a = a(0) and finds L-dimensional $b' = Aa(0)$, which is then normalized by computing $\beta = (b'^T Y^T Y b')^{1/2}$ and taking $b(1) = b'/\beta$. Now next a-vector is computed as $a' = Bb(1)$ and then normalized similarly as $a(1) = a'/(a'^T X^T X a')^{1/2}$. Next iterations run similarly. The convergence is warranted if the maximal eigenvalue is greater than the other eigenvalues indeed. The maximum eigenvalue is computed as the product of the norms of a' and b'. The corresponding eigenvectors a and b can be considered as the best numerical codes for the categories; they are mutually oriented towards each other.

Moreover, the total of the canonical eigenvalues is related to the Pearson chi-squared coefficient between the two nominal variables. Since both spaces, $L(X)$ and $L(Y)$, contain the bisector subspace of all N-dimensional vectors with equal elements, this generates a trivial eigenvalue 1, which should not be taken into account when considering the total of canonical eigenvalues. As is well known, the sum of all eigenvalues of a square matrix is invariant under linear transformations of the space; it is equal to the sum of the diagonal elements. As was shown above the (i,i)-th element of matrix $P_X P_Y$ is equal to $N_{kl}/(N_{k+}N_{+l})$, where k and l are those indices for which $i \in T_l$ and $i \in S_k$. The number of objects i such that $i \in T_l$ and $i \in S_k$, is equal to N_{kl}. Therefore, the total of diagonal elements in matrix $P_X P_Y$ is equal to

$$\sum_{k=1}^{K}\sum_{l=1}^{L} N_{kl}\frac{N_{kl}}{N_{k+}N_{+l}} = \sum_{k=1}^{K}\sum_{l=1}^{L}\frac{N_{kl}^2}{N_{k+}N_{+l}} = \sum_{k=1}^{K}\sum_{l=1}^{L}\frac{p_{kl}}{p_{k+}p_{+l}} = X^2/N + 1,$$

according to Eq. (3.31). By subtracting the trivial eigenvalue 1, we conclude that the total of non-trivial canonical eigenvalues is equal to X^2/N.

The phenomenon of canonical correlation between nominal features has attracted considerable attention of researchers. In particular, there are two techniques of numerical analysis of nominal features based on the canonical correlation. One is referred to as dual scaling (see Nishisato 2014); that utilizes a version of the iterations according to the power method, as described above. Another, referred to as the correspondence analysis, builds on the simultaneous consideration of both spaces, $L(X)$ and $L(Y)$, and processes related to their interrelation (Benzecri 1992; Greenacre 2017; Lebart et al. 1995). Specifically, an attention is given to equations relating the canonical vectors, $a = \mu Bb$ and $b = \mu Aa$, the so-called transition formulas. Since elements of A and B are conditional probabilities $p(k/l) = p_{lk}/p_{+l}$, and $p(l/k) = p_{kl}/p_{k+}$, respectively, a and b appear to be averaged versions of each other (up to the singular value μ), which leads to a joint display of both S-categories and T-categories.

This author developed a symmetric version of the correspondence analysis which involves no dual spaces; this is described in the next Sect. 3.6.3.

3.6.3 Correspondence Analysis

3.6.3.1 Correspondence Analysis: Presentation

Correspondence Analysis is an extension of PCA to contingency tables taking into account the specifics of co-occurrence data: they are not only comparable across the table but also can be meaningfully summed together across the table. This leads to a unique way of standardization of such data—by using the Quetelet coefficients

rather than the original frequencies, which is an advantage over the common situations in which the data standardization is rather arbitrary.

Correspondence Analysis (CA) is a method for visually displaying both row and column categories of a contingency table $P = (p_{ij})$, $i \in I$, $j \in J$, in such a way that distances between the presenting points reflect the patterns of co-occurrences in P. This method is usually introduced as a set of dual heuristics applied simultaneously to rows and columns of the contingency table (see, for example, Lebart et al. 1995; Greenacre 2017). Yet there is a way for introducing CA as an encoder-decoder based data recovery technique similar to that used for introducing PCA above. According to this perspective (Lebart and Mirkin 1993; Mirkin 1996), CA is a version of PCA differing from PCA due to the specifics of contingency data, in the following aspects:

(i) The CA method obtains hidden factors representing the relative Quetelet indexes rather than the original frequency data;
(ii) Both rows and columns are not equivalently contributing; each is assigned with a weight reflecting its frequency; the greater the frequency, the greater the weight. These weights are used in the approximation problem throughout; both in the least-squares criterion and the mutual orthogonality conditions;
(iii) Both rows and columns are visualized on the same display, thus referred to as a biplot, in such a way that the geometric distances between the representing points reflect the so-called chi-square distances between row (or column) conditional frequency profiles;
(iv) The data scatter is defined as the sum of squared entries weighted by the products of the row and column weights, that is equal to the Pearson chi-squared association coefficient.

Worked Example 3.13. Correspondence Analysis of a Theft/Age Contingency Table

Consider Table 3.26, cross-classifying cases of attempted theft from shops and supermarkets in the Netherlands 1979 from the book by Israëls 1987; see also Lombardo et al. (2016).

Two classification bases are: categories of stolen goods and age groups of perpetrators. There are 13 goods categories, from closing to household items to perfumes; see the list in the left column of Table 3.26. There are 9 age groups (less than 12 years old, 12–14 years old, 15–17, 18–20, 21–29, 30–39, 40–49, 50–64, 65 and over years old) coded accordingly in the first row of Table 3.26.

To apply the method to data in Table 3.26, we first transform that into Quetelet coefficients format (see Table 3.27).

This standardization does make the data structure somewhat. It suffices to mention pairs (tobacco, AGold) and (toys, AGmin) whose q-values exceed 200% increase from the average. But the transformation $p \Rightarrow q$ is not alone in CA. It is coupled with the weighting of each row and column by its corresponding marginal

Table 3.26 Cross-classification of theft cases over stolen goods and age groups

Category	AGmin	AG13	AG16	AG19	AG25	AG35	AG45	AG57	AGold	Total
Clothing	81	138	304	384	942	359	178	137	45	2568
Accessories	66	204	193	149	297	109	53	68	28	1167
Tobacco	150	340	229	151	313	136	121	171	145	1756
Stationery	667	1409	527	84	92	36	36	37	17	2905
Books	67	259	258	146	251	96	48	56	41	1222
Records	24	272	368	141	167	67	29	27	7	1102
Household	47	117	98	61	193	75	50	55	29	725
Candy	430	637	246	40	30	11	5	17	28	1444
Toys	743	684	116	13	16	16	6	3	8	1605
Jewelry	132	408	298	71	130	31	14	11	10	1105
Perfumes	32	57	61	52	111	54	41	50	28	486
Hobby	197	547	402	138	280	200	152	211	111	2238
Other	209	550	454	252	624	195	88	90	34	2496
Total	2845	5622	3554	1682	3446	1385	821	933	531	20,819

Table 3.27 Quetelet indexes for the goods/age contingency Table 3.26

Category	AGmin	AG13	AG16	AG19	AG25	AG35	AG45	AG57	AGold	Total
Clothing	−0.77	−0.80	−0.31	0.85	1.22	1.10	0.76	0.19	−0.31	0.1233
Accessories	−0.59	−0.35	−0.03	0.58	0.54	0.40	0.15	0.30	−0.06	0.0561
Tobacco	−0.37	−0.28	−0.24	0.06	0.08	0.16	0.75	1.17	2.24	0.0843
Stationery	0.68	0.80	0.06	−0.64	−0.81	−0.81	−0.69	−0.72	−0.77	0.1395
Books	−0.60	−0.22	0.24	0.48	0.24	0.18	0.00	0.02	0.32	0.0587
Records	−0.84	−0.09	0.96	0.58	−0.08	−0.09	−0.33	−0.45	−0.75	0.0529
Household	−0.53	−0.40	−0.21	0.04	0.61	0.56	0.75	0.69	0.57	0.0348
Candy	1.18	0.63	0.00	−0.66	−0.87	−0.89	−0.91	−0.74	−0.24	0.0694
Toys	2.39	0.58	−0.58	−0.90	−0.94	−0.85	−0.91	−0.96	−0.80	0.0771
Jewelry	−0.13	0.37	0.58	−0.20	−0.29	−0.58	−0.68	−0.78	−0.65	0.0531
Perfumes	−0.52	−0.57	−0.26	0.32	0.38	0.67	1.14	1.30	1.26	0.0233
Hobby	−0.36	−0.09	0.05	−0.24	−0.24	0.34	0.72	1.10	0.94	0.1075
Other	−0.39	−0.18	0.07	0.25	0.51	0.17	−0.11	−0.20	−0.47	0.1199
Total	0.1367	0.27	0.1707	0.0808	0.1655	0.0665	0.0394	0.0448	0.0255	1.0000

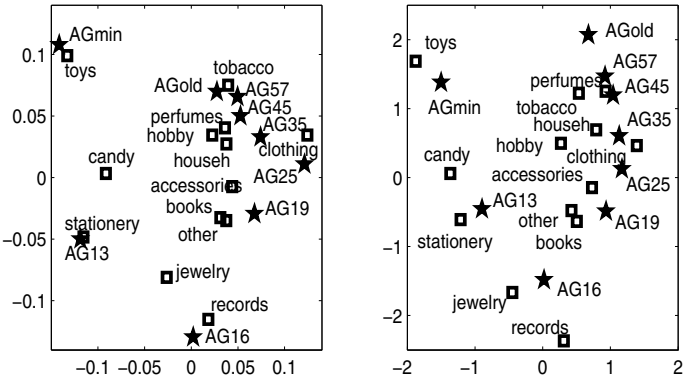

Fig. 3.26 Visualization of thefts/age contingency table in Table 3.26 using Correspondence Analysis, on the left, and PCA, on the right. Squares stand for good types and stars for thief categories

probability so that the squared errors in the criterion are weighted by products of the marginal probabilities. Moreover, the vector norm is weighted by them too.

Figure 3.26 represents a CA visualization of Table 3.26 derived as described above, on the left, and PCA, on the right. The visualizations do not differ that much, although the CA display gives a more clear picture. They clearly show which good types an age group tends to steal. AGmin is close to toys, whereas AG16 is near jewelry and records.

3.6.3.2 Correspondence Analysis: Formulation

Correspondence Analysis (CA) is a method for visually displaying both row and column categories of a contingency table $P = (p_{ij})$, $i \in I$, $j \in J$, in such a way that distances between the presenting points reflect the pattern of co-occurrences in P. To be specific, let us take on the issue of visualization of P on a 2D plane so that we are looking for just two approximating factors, $u_1 = (v_1, w_1)$ where $v_1 = (v_1(i))$ and $w_1 = (w_1(j))$ and $u_2 = (v_2, w_2)$ where $v_2 = (v_2(i))$ and $w_2 = (w_2(j))$, with $I \cup J$ as their domain, such that each row $i \in I$ is displayed as point $u(i) = (v_1(i), v_2(i))$ and each column $j \in J$ as point $u(j) = (w_1(j), w_2(j))$ on the plane as shown in Fig. 3.26.

The $|I|$-dimensional vectors v_t and $|J|$-dimensional vectors w_t constituting u_t ($t = 1, 2$) are calculated to approximate the relative Quetelet coefficients $q_{ij} = p_{ij}/(p_{i+}p_{+j}) - 1$ rather than the co-occurrences p_{ij} themselves, according to equations:

$$q_{ij} = \mu_1 v_1(i) w_1(j) + \mu_2 v_2(i) w_2(j) + e_{ij} \tag{3.42}$$

where μ_1 and μ_2 are positive reals, by minimizing the weighted least-squares criterion

$$E^2 = \sum_{i \in I} \sum_{j \in J} p_{i+} p_{+j} e_{ij}^2 \tag{3.43}$$

with regard to μ_t, v_t, w_t, subject to conditions of weighted orthonormality:

$$\sum_{i \in I} p_{i+} v_t(i) v_{t'}(i) = \sum_{j \in J} p_{+j} w_t(j) w_{t'}(j) = \begin{cases} 1 & \text{if } t = t' \\ 0 & \text{otherwise} \end{cases} \tag{3.44}$$

where t, $t' = 1, 2$.

The weighted criterion E^2 is equivalent to the unweighted least-squares criterion L^2 applied to the matrix R with Pearson indexes $r_{ij} = q_{ij}(p_{i+}p_{+j})^{1/2} = (p_{ij} - p_{i+}p_{+j})/(p_{i+}p_{+j})^{1/2}$ as its entries. To be exact, let us consider model (3.42′) below:

$$r_{ij} = \alpha_1 f_1(i) g_1(j) + \alpha_2 f_2(i) g_2(j) + \varepsilon_{ij} \tag{3.42'}$$

and try minimize the sum of squared residuals $L^2 = \sum_{i,j} \varepsilon_{ij}^2$. According to the SVD theory, the solution to this problem is constituted by the two maximal singular values α_1 and α_2 of matrix R = (r_{ij}) and corresponding pairs of the normed singular vectors (f_1, g_1) and (f_2, g_2), respectively. Let us put $r_{ij} = q_{ij}(p_{i+}p_{+j})^{1/2}$ in (3.42′) and divide the equation by $(p_{i+}p_{+j})^{1/2}$, which leaves it invariant:

$$q_{ij} = \alpha_1 f_1(i) g_1(j) / (p_{i+} p_{+j})^{1/2} + \alpha_2 f_2(i) g_2(j) / (p_{i+} p_{+j})^{1/2} + \varepsilon_{ij} / (p_{i+} p_{+j})^{1/2}$$

This can be equivalently rewritten as:

$$q_{ij} = \alpha_1 v_1(i) w_1(j) + \alpha_2 v_2(i) w_2(j) + e_{ij} \tag{3.42''}$$

where $v_t(i) = f_{it}/(p_{i+}^{1/2})$, $w_t(j) = g_{jt}/(p_{+j}^{1/2})$, and $e_{ij} = \varepsilon_{ij}/(p_{i+}p_{+j})^{1/2}$. Equation (3.42″) is similar to Eq. (3.42). What is about criterion? One can see that $\varepsilon_{ij} = e_{ij}(p_{i+}p_{+j})^{1/2}$. Therefore, the $L^2 = E^2$. That means that the α_1 and α_2 are the maximal singular values of R, and the above definitions provide for the solutions of the problem in (3.42), (3.43), and (3.44).

Therefore, the factors v and w are determined by the singular-value decomposition of matrix $R = (r_{ij})$. More explicitly, the two maximal singular values μ_t and corresponding singular vectors $f_t = (f_{it})$ and $g_t = (g_{jt})$ of matrix R, defined by equations $Rg_t = \mu_t f_t$, $R^T f_t = \mu_t g_t$ $(t = 1, 2)$ determine the optimal values μ_t and optimal solutions to the problem of minimization of (3.43)–(3.44), according to equations $v_t(i) = f_{it}/(p_{i+}^{1/2})$ and $w_t(j) = g_{jt}/(p_{+j}^{1/2})$.

The singular triplet equations can be rewritten in terms of v_t and w_t, as follows:

$$\sum_{j \in J} \frac{p_{ij}}{p_{i+}} w_t(j) = \mu_t v_t(i), \quad \sum_{i \in I} \frac{p_{ij}}{p_{+j}} v_t(i) = \mu_t w_t(j) \tag{3.45}$$

To prove the left-hand equation, take equation $Rg_t = \mu_t f_t$ in its component-wise form, $\sum_{j\in J} r_{ij} g_j = \mu f_i$ (index t omitted for the sake of convenience) and substitute by vectors v and w defined above: $\sum_{j\in J} r_{ij} \left(\frac{p_{+j}}{p_{i+}}\right)^{1/2} w(j) = \mu v(i)$. This is equivalent to $\sum_{j\in J} \left(\frac{p_{ij}}{p_{i+}} - p_{+j}\right) w(j) = \mu v(i)$. To complete the proof, equation $\sum_j p_{+j} w(j) = 0$ is to be proven. To do that, let us first prove that vector g_0 whose components are $p_{+j}^{1/2}$ is a singular vector of R corresponding to singular value 0 (the other component of the singular triplet is equal to $f_0 = (p_{i+}^{1/2})$). Indeed,

$$\sum_{j\in J} r_{ij} p_{+j}^{1/2} = (1/p_{i+}^{1/2}) \sum_{j\in J} (p_{ij} - p_{i+} p_{+j}) = (1/p_{i+}^{1/2})(p_{i+} - p_{i+}) = 0.$$

Then the equation $\sum_j p_{+j} w(j) = 0$ follows from the fact that all the singular vectors are mutually orthogonal so that singular vector g corresponding to w is orthogonal to g_0, which proves the statement. The right-hand equation can be proven in a similar way, from equation $R^T f_t = \mu_t g_t$.

Equations (3.45) are referred to as transition equations and considered to justify the joint display of rows and columns because the row-points $v_t(i)$ appear to be averaged column-points $w_t(j)$ and, vice versa, the column-points appear to be averaged versions of the row-points, up to the singular value of μ_t course.

The mutual location of the row-points is considered as justified by the fact that between-row-point squared Euclidean distances $d^2(u(i), u(i'))$ approximate the chi-square distances between corresponding rows of the contingency table. Specifically, chi-square distance is defined as a weighted squared Euclidean distance:

$$\chi^2(i, i') = \sum_{j\in J} p_{+j} (q_{ij} - q_{i'j})^2 = \sum_{j\in J} (p_{ij}/p_{i+} - p_{i'j}/p_{i'+})^2 / p_{+j}. \quad (3.46)$$

Here $u(i) = (v_1(i), v_2(i))$ for v_1 and v_2 rescaled in such a way that their norms are equal to μ_1 and μ_2, respectively. A similar property holds for columns j, j'. In fact, it is the right-hand item in (3.46) which is used to define the chi-squared distance between either columns or rows of a contingency table (Lebart et al. 1995), but the definition in terms of Quetelet coefficients in the middle of (3.46) (Mirkin 1996) looks more natural. The distance is dubbed chi-square distance because of its links to the chi-square coefficient for the table P. First of all, if we take the weighted chi-square summary distance to 0, $\sum_{i\in I} p_{i+} c^2(i, 0)$ where 0 is put instead of $q_{i'j}$ in (3.46), it is easy to see that this is the Pearson chi-squared coefficient, without the factor N of course, which is simultaneously the expression for the data scatter according to criterion E^2 in (3.43):

$$\sum_{i\in I} p_{i+} c^2(i, 0) = \sum_{i\in I} \sum_{j\in J} p_{i+} p_{+j} e_{ij}^2 = X^2/N \quad (3.47)$$

The weighted data scatter is equal to the scatter of R, the sum of its squared entries $T(R)$, which can be easily proven from the definition of R. Indeed, $T(R) = \sum_{i \in I} \sum_{j \in J} \left(p_{ij} - p_{i+}p_{+j}\right)^2 / (p_{i+}p_{+j}) = X^2/N$. This implies that

$$X^2/N = \mu_1^2 + \mu_2^2 + E^2 \tag{3.48}$$

which can be seen as a decomposition of the contingency data scatter, expressed by X^2, into contributions of the individual factors, μ_1^2 and μ_2^2, and unexplained residuals, E^2. (Only two factors are considered here, but the number of factors to be found can be raised up to the rank of matrix R with no other changes).

In a common situation, the first two singular values account for a major part of X^2, thus justifying the use of the plane of the first two factors for visualization of the interrelations between I and J.

3.6.3.3 Correspondence Analysis: Computation

Given a contingency table P, the computation of correspondence analysis factors can go in three steps: (a) computing Pearson index matrix R, (b) finding the singular decomposition of R and the two first correspondence analysis factors, and (c) visualization of the joint display of rows and columns of P. Here are MatLab commands for these.

(a) Computing Pearson index matrix R

```
≫Pc=sum(P); Pr=sum(P′); total=sum(Pc);
≫P=P/total; %relative frequencies
≫Pc=Pc/total; %column relative frequencies
≫Pr=Pr/total; %row relative frequencies
≫Prod=Pr′*Pc; % matrix of products
≫ rProd=Prod.^(0.5); % square roots of products
≫ r=(P-Prod)./rProd; % Pearson index matrix
```

(b) Finding the correspondence analysis factors:

```
≫[a,mu,b]=svd(r);
≫% finding first factor
≫x1=a(:,1)./sqrt(Pr′);
≫y1 = b(:,1)./sqrt(Pc′);
≫% finding second factor
≫x2= a(:,2)./sqrt(Pr′);
≫y2= b(:,2)./sqrt(Pc′);
```

As a bonus, one can estimate the proportion of data scatter, the chi-squared, taken into account by the factors, and display it on the screen:

```
≫yy=r.*r; chi=sum(sum(yy))% data scatter
```

≫ccn=(mu(1,1)^2+mu(2,2)^2)*100/chi;
%contribution of the first two
≫disp('Contribution of the solution:'); ccn

(c) Visualization of the joint display of rows and columns of *P*. The plot is easy to do with command

≫plot(x1,x2,'ks', y1,y2,'kp');

Yet to make the points annotated with row and column names, which are to be available in a string cell termed say 'names', the joint set of rows and columns should get their x-coordinate and y-coordinate vectors, z1 and z2 below:

≫z1=[x1' y1']; z2=[x2' y2']; text(z1,z2,names);
≫v=axis; axis(1.5*v);

The last line is to make the plot to look tighter by extending its boundaries.

3.6.4 Correlation Between Projection Matrices

Consider two nominal features over an entity set I of cardinality N represented by partitions $T = \{T_l\}$ and $S = \{S_k\}$. Define the $N \times L$ dummy matrix X and $N \times K$ dummy matrix Y corresponding to partitions T and S, respectively.

Consider the linear subspaces $L(X)$ and $L(Y)$ spanning matrices X and Y, as well as corresponding orthogonal projection matrices $P_X = X(X^TX)^{-1}X^T$ and $P_X = Y(Y^TY)^{-1}Y^T$. Such a matrix expresses the orientation of the space $L(X)$ or $L(Y)$ in a concise way, like the normal vector a for a hyperplane defined by the equation $\langle a, x \rangle = 0$. Recall that the spaces $L(X)$ or $L(Y)$ overlap over the unidimensional bisector line. To take this parasitic subspace out, one should subtract the parasitic subspace from the matrices—by simply subtracting 1/N from all the elements of the matrices, which transforms them to $P_X^-$and P_Y^-.

The similarity between the corrected subspaces $L(X)^-$ and $L(Y)^-$ can be measured by the inner product, or even a correlation index, between P_X^- and P_Y^-.

The inner product is equal to

$$\langle P_X^-, P_Y^- \rangle = \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{i \in S_k} \sum_{j \in T_l} \left(\frac{1}{N_{k+}} - \frac{1}{N} \right) \left(\frac{1}{N_{+l}} - \frac{1}{N} \right)$$
$$= \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{i \in S_k} \sum_{j \in T_l} \left(\frac{1}{N_{k+}N_{+l}} - \frac{1}{NN_{+l}} - \frac{1}{NN_{k+}} + \frac{1}{N^2} \right).$$

The latter expression is the result of multiplication of the expressions in the parentheses in the former one. Let us figure out what are the summation results for

each of the four items. The item $1/(N_{k+}N_{+l})$ appears in those (i,j) at which $i \in S_k$ and $j \in S_l$. that is, at those (i,j) at which both $i,j \in S_k \cap S_l$. That implies that the sum of these items is equal to $\sum_{k=1}^{K}\sum_{l=1}^{L}\frac{N_{kl}^2}{N_{k+}N_{+l}}$. The second item does not depend on i, so that there are N of them to sum. Within any category T_l, there are N_l items leading to the summary value $-1/N$ within each $l = 1,2\ldots, N$. Summing these over l leads to -1 as the total. Similarly, the total of summation of the third item is -1 as well. The item number four, $1/(N^2)$, summed N^2 times, over all i and all j, will produce 1 as the total. Altogether, the inner product of the projection matrices is equal to

$$\langle P_X^-, P_Y^- \rangle = \sum_{k=1}^{K}\sum_{l=1}^{L}\frac{N_{kl}^2}{N_{k+}N_{+l}} - 1 = Q. \tag{3.49}$$

That is, rather unexpectedly, the inner product of orthogonal projector's matrices is the average Quetelet index, or Pearson chi-squared related to N.

Q.3.26. Prove that the sum of elements of matrix P_X^- considered as an $N \times N$ vector is 0; so is the mean of P_X^-.

Q.3.27. Prove that the sum of squares of elements of matrix P_X^- considered as an $N \times N$ vector is $K - 1$ where K is the number of X-categories. **Hint**: Follow the way of the previous paragraph. Consider the sum of all the elements of P_X^- to be multiplied by themselves, in real, and see what are the results of summation of them over all i,j = 1,…, N.

Now we are ready to see what is the correlation coefficient between matrices P_X^- and P_Y^-:

$$\rho(P_X^-, P_Y^-) = \frac{Q}{\sqrt{(K-1)(L-1)}} = \frac{\sum_{k=1}^{K}\sum_{l=1}^{L}\frac{(N_{kl}-N_{k+}N_{+l})^2}{N_{k+}N_{+l}}}{\sqrt{(K-1)(L-1)}} \tag{3.50}$$

That is an interesting index. According to Kendall and Stewart (1967, Eq. (33.64)), that is the squared value of the so-called Tschuprow association coefficient. A. Tschuprow (1874–1926) proposed it in early 20th century based on the fact that the product $(K-1)(L-1)$ is the mean value for Pearson's chi-squared under the hypothesis that the features are statistically independent.

Q.3.28. Give an analytic expression for the quadratic distance between P_X^- and P_Y^-.

A. That is $d(P_X^-, P_Y^-) = \langle P_X^-, P_X^- \rangle + \langle P_Y^-, P_Y^- \rangle - 2\langle P_X^-, P_Y^- \rangle = K - 1 + L - 1 - 2Q = 2((K + L)/2 - 1 - Q)$ where Q is the average Quetelet coefficient. In the case when $K = L$, this works fine. However, the greater the difference between L and K, the greater the minimum value of d differs from 0, which perhaps correctly reflects the mismatch between the two features.

3.7 Distance Between Relations Corresponding to Tied Rankings and Partitions

This material follows that by Mirkin and Fenner (2019). The topic of comparing rankings was initiated by Charles Spearman (1863−1945) (a junior collaborator of the founding fathers of multivariate statistics, Francis Galton and Karl Pearson), who was hired to further pursue the golden dream of Galton, a proof that human talent is inherited, mainly from one's parents and, partly, from even more distant ancestors. Although ranking is non-quantitative, Spearman proposed using ranks as numerical values, so that the Pearson correlation coefficient could be employed. This is straightforward when the observations being compared are linearly ordered. However, different observations can sometimes be assigned the same numerical rank value, which then led to the introduction of the term "tied observations", subsequently replaced by "tied rankings". Formally, a *tied ranking* can be represented as an ordered partition $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$, that is, a partition whose parts are linearly ordered by their indices 1, 2, …, p. We say that an element i *precedes* an element j in the ranking $\boldsymbol{R}$ if the part containing i precedes the part containing j. A clear-cut case of an ordered partition is given by the rank features in social surveys. A ranked feature asks respondents to classify alternatives using an ordered set of categories, such as "strongly agree", "agree", "neutral", "disagree", "strongly disagree". The term ranking is used here as a synonym of the ordered partition; when considering a series of objects with no ties, that is referred to as a strict ranking.

In 1938, a British statistician Maurice Kendall (1907–1983) introduced a different representation for rankings by using the relation of precedence between ranks rather than the ranks themselves. Given a tied ranking $\boldsymbol{R}$, we define a square observation-to-observation matrix (the *Kendall matrix*) in which the (i, j) entry is +1 if i precedes j in $\boldsymbol{R}$, 0 if i and j have the same rank, or −1 if j precedes i. The Kendall rank correlation coefficient between two tied rankings is the Pearson correlation coefficient between the corresponding Kendall matrices, considered as vectors in an N×N-dimensional space.

In the 1950s, John Kemeny (1926–1992) approached the issue of comparing rankings from a social consensus perspective. Given a set of ordered partitions, a consensus ordered partition should represent the major tendency in the set. A conventional approach, the majority rule, may fail when determined by voting on pairs of alternatives.

Specifically, the so-called Condorcet paradox holds: if there are three parties at a meeting, each supporting one of three cyclically related linear orderings of three alternatives, say, (a) [i,j,k], (b) [j,k,i], and (c) [k,i,j], respectively, then the majority rule would lead to a cycle in the precedence relation: i would precede j because this is so for the majority, (a) and (c); similarly, j would precede k, and k would precede i. This contradicts the requirement that the precedence relation corresponding to the majority consensus ranking should be transitive. This paradox is a basis of the celebrated "social choice impossibility theorem" by an American economist Kenneth Arrow (1921–2017), a Nobel Prize winner.

John Kemeny proposed a different definition for consensus ranking using a distance measure between rankings. Rather than defining any specific distance measure ab initio, he formulated four axioms that should hold for any admissible distance measure. These axioms led Kemeny to derive the unique distance measure satisfying them. The *Kemeny distance* turned out to be the L_1-*distance* between the Kendall matrices [see (Kemeny and Snell 1962) for a convincing exposition].

Here, we are going to describe a joint geometric space of ordered and unordered partitions using the corresponding weak order and equivalence relations on the set of observations (as it is done in (Mirkin 1979; Mirkin and Fenner 2019).

a. **Weak orders and equivalence relations**

Given a finite set I of N elements, a collection of its subsets $\boldsymbol{R} = \{R_1, R_2, \ldots, R_p\}$ is referred to as a *partition* if the subsets R_s are all non-empty, non-overlapping, and cover the entire set I, so that each $i \in I$ belongs to a unique subset R_s, $1 \leq s \leq p$. The subsets are called the parts of the partition $\boldsymbol{R}$. A partition is said to be *ordered* if there is a linear order relation of precedence between its parts, $R_s < R_t$, that is transitive, anti-reflexive and complete. If the order coincides with the natural order between indices 1, 2, …, p, we use parentheses to denote this, viz. $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$. In Decision Theory, an ordered partition is referred to as a *ranking*.

Each ordered partition $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$ generates a *binary preference relation*

$$\rho = \{(i,j) : i \in R_s, j \in R_t, \text{and } s \leq t\}. \tag{3.51}$$

Usually, two non-overlapping binary relations are defined with respect to a ranking $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$: the *strict preference* relation $P = \{(i, j): i \in R_s, j \in R_t, \text{and } s < t\}$ and the *indifference* relation $E = \{(i, j): i, j \in R_s \text{ for some } s\}$. The indifference relation E here is transitive, reflexive and symmetric, thus E is the equivalence relation corresponding to the unordered partition $\check{\boldsymbol{R}}$ having the same parts as $\boldsymbol{R}$. Obviously, $\rho = P \cup E$, that is, ρ in (3.51) is a non-strict preference relation in which the strict preference and indifference relations are merged together. Usually, researchers try to avoid such a "mix"; but we will see later that there is no problem with this merger. The next part of this section is a brief reminder of some conventional concepts and facts about preference relations [see, for example, (Steele and Stefánsson 2015)].

If ρ is a binary relation, its inverse ρ^{-1} is defined as $\rho^{-1} = \{(i, j): (j, i) \in \rho\}$. If ρ is the preference relation corresponding to a tied ranking $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$, then its inverse ρ^{-1} corresponds to the reverse tied ranking $\boldsymbol{R}^{-1} = (R_p, \ldots, R_2, R_1)$. It is easy to see that the indifference relation E corresponding to any tied ranking $\boldsymbol{R}$ satisfies $E = \rho \cap \rho^{-1}$. Thus, the strict preference relation is the difference $P = \rho - E = \rho - \rho^{-1}$.

It is clear that ρ in (3.51) is

– Reflexive, that is, $(i, i) \in \rho$ for any $i \in I$,

– Transitive, that is, if (i, j) ∈ ρ and (j, k) ∈ ρ, then (i, k) ∈ ρ for any i, j, k ∈ *A*, and
– Complete, that is, (i, j) ∈ ρ or (j, i) ∈ ρ, or both, for any i, j ∈ *A*.

Of course, reflexivity can be considered as a special case of completeness for which i = j. A binary relation satisfying these properties is usually referred to as a *weak order*. In fact, a converse statement also holds: A preference relation ρ corresponds to an ordered partition $\boldsymbol{R}$ if and only if it is a weak order.

To prove that, assume ρ to be a binary relation on the set *I* that is reflexive, transitive and complete. Consider any i ∈ *I* and define the subset ρ(i) = {j∈ *I*: (i, j) ∈ ρ}. Then, for any pair i, k ∈ *I*, if (i, k) ∈ ρ then ρ(k) ⊆ ρ(i). This holds because whenever j ∈ ρ(k), i.e. (k, j) ∈ ρ, then (i, j)∈ ρ also, because ρ is transitive. Therefore, since ρ is complete, for any pair i, k ∈ *I*, either ρ(k) ⊆ ρ(i) or ρ(i) ⊆ ρ(k), or both. It follows that the collection of sets ρ(i) is linearly ordered by set-theoretic inclusion, so they can be ordered as a sequence of sets S_t for t = 1, 2, …, p, where $S_1 \supset S_2 \supset \ldots \supset S_p$. Then the subsets $R_t = S_t - S_{t+1}$, t = 1, 2, …, p-1, and $R_p = S_{p,}$, form a ranking $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$. It is quite easy to check that its corresponding preference relation (3.51) coincides with the given relation ρ. The reverse implication, that the relation (3.51) corresponding to an ordered partition is reflexive, transitive and complete, has already been established above. This completes the proof.

The subsets $R_t = S_t - S_{t+1}$ in the proof each satisfy $R_t = \rho(i) \cap \rho^{-1}(i)$ for some i ∈ *I*. This establishes that a binary relation ρ is a weak order if and only if its strict part *P* is anti-reflexive and transitive, its indifference part *E* is an equivalence relation, and *P*, P^{-1}, *E* form a partition of the Cartesian product *I*×*I*.

b. **Refinement and betweenness**

A ranking $\boldsymbol{R}'$ is a *refinement* of a ranking $\boldsymbol{R}$ if it is obtained from the latter by subdividing some of its parts into smaller ones, and some ordering is defined between the smaller parts of each subdivided part of $\boldsymbol{R}$. The corresponding preference relations, ρ′ and ρ, are related by set-theoretic inclusion: A tied ranking $\boldsymbol{R}'$ is a refinement of a tied ranking $\boldsymbol{R}$ if and only if ρ′ ⊂ ρ.

Indeed, if $\boldsymbol{R}'$ is a refinement of a tied ranking $\boldsymbol{R}$ then, for some pairs i, j of elements of *A* such that both (i,j) ∈ ρ and (j,i)∈ ρ, only one of these holds for ρ′. Conversely, suppose that ρ and ρ′ correspond to tied rankings $\boldsymbol{R}$ and $\boldsymbol{R}'$, respectively, and that ρ′ ⊂ ρ. Then ρ′(i) ⊆ ρ(i) for any i ∈ *I*, and, moreover, the inclusion is proper for some i ∈ *I*. Consider any such i. Let {i1, i2, …, ik} be a maximal subset of *I* such that $\rho(i) \supset \rho'(i1) \supset \rho'(i2) \supset \ldots \supset \rho'(ik)$. Then, by the previous analysis, every equivalence class $R'_{tu} = \rho'(iu) \cap \rho'^{-1}(iu)$ will be part of the equivalence class $R_t = \rho(i) \cap \rho^{-1}(i)$, which completes the proof.

We say that ρ is *coarser* than ρ′, if ρ′ is a refinement of ρ.

A binary relation τ on *I* is said to be *between* binary relations ρ and ρ′ if and only if $\rho \cap \rho' \subseteq \tau \subseteq \rho \cup \rho'$. A ranking $\boldsymbol{T}$ is said to be between tied rankings $\boldsymbol{R}$ and $\boldsymbol{R}'$ if, for any i, j ∈ *I*, the ordering between them in $\boldsymbol{T}$ is compatible with their ordering in

both $\boldsymbol{R}$ and $\boldsymbol{R}'$: that is, (i) if i precedes j in both $\boldsymbol{R}$ and $\boldsymbol{R}'$ then i precedes j in $\boldsymbol{T}$; (ii) if i precedes j in one of $\boldsymbol{R}$ and $\boldsymbol{R}'$, and i and j are indifferent in the other, then i either precedes j or is indifferent to j in $\boldsymbol{T}$; (iii) if i and j are indifferent in both $\boldsymbol{R}$ and $\boldsymbol{R}'$, then i and j are indifferent in $\boldsymbol{T}$; lastly, (iv) if i precedes j in $\boldsymbol{R}$ but j precedes i in $\boldsymbol{R}'$, then anything can be true of the ordering between i and j in $\boldsymbol{T}$: i may precede j, or j may precede i, or i and j may be indifferent in $\boldsymbol{T}$ (Kemeny and Snell 1962). It is easy to prove that $\boldsymbol{T}$ is between $\boldsymbol{R}$ and $\boldsymbol{R}'$ if and only if the same is true for their weak orders.

In the general case of two arbitrary tied rankings $\boldsymbol{R}$ and $\boldsymbol{R}'$, the relation $\rho \cap \rho'$ is a partial preference relation because there can be i, j $\in I$ such that i strictly precedes j in $\boldsymbol{R}$, whereas j strictly precedes i in $\boldsymbol{R}'$, so that neither (i, j) nor (j, i) belongs to $\rho \cap \rho'$.

Such a case, which is not uncommon, is exemplified by a proverbial question: "What is better: being poor but healthy or being rich but ill?" (with a proverbial answer that to be both rich and healthy is better indeed.)

What is appealing about $\rho \cap \rho'$ is that its indifference relation is always an equivalence relation, thus corresponding to the partition that is just the intersection of the unordered partitions $\check{\boldsymbol{R}}$ and $\check{\boldsymbol{R}}'$ that correspond to the ordered partitions $\boldsymbol{R}$ and $\boldsymbol{R}'$, respectively. The intersection $\check{\boldsymbol{R}} \cap \check{\boldsymbol{R}}'$ is the partition of I in which the parts are the intersections $R_s \cap R_t'$ of some part R_s of $\boldsymbol{R}$ and some part R_t' of $\boldsymbol{R}'$ for which R_s and R_t' are not disjoint.

Both ordered and unordered intersections can be visualized as a block matrix in which the blocks are formed by the subsets of rows and columns corresponding to the parts of the ordered partitions $\boldsymbol{R}'$ and $\boldsymbol{R}$, respectively (see Fig. 3.27). Of course, the blocks of the intersections are only partially ordered so that, for example, blocks $R_2' \cap R_3$ and $R_3' \cap R_2$ are not comparable. However, a linear order can be imposed naturally by ordering the blocks first by rows and then by columns, so that any block of the first row precedes the blocks in all other rows. This is the so-called lexicographic product $\boldsymbol{R}' * \boldsymbol{R}$ introduced in (Mirkin 1979). Similarly, an alternative

	R_1	R_2	R_3	R_4	R_5
R_1'	$R_1' \cap R_1$	$R_1' \cap R_2$	$R_1' \cap R_3$	$R_1' \cap R_4$	$R_1' \cap R_5$
R_2'	$R_2' \cap R_1$	$R_2' \cap R_2$	$R_2' \cap R_3$	$R_2' \cap R_4$	$R_2' \cap R_5$
R_3'	$R_3' \cap R_1$	$R_3' \cap R_2$	$R_3' \cap R_3$	$R_3' \cap R_4$	$R_3' \cap R_5$
R_4'	$R_4' \cap R_1$	$R_4' \cap R_2$	$R_4' \cap R_3$	$R_4' \cap R_4$	$R_4' \cap R_5$

Fig. 3.27 A visual representation of the intersection of two rankings $\boldsymbol{R}' \cap \boldsymbol{R}$, where $\boldsymbol{R}'$ relates to rows and $\boldsymbol{R}$ to columns. It is assumed that the rows and columns are permuted according to the rankings $\boldsymbol{R}'$ and $\boldsymbol{R}$, respectively

lexicographic product $\boldsymbol{R} * \boldsymbol{R}'$ is defined by ordering blocks first by columns and then by rows. Curiously, in the ordered series $\boldsymbol{R}'$, $\boldsymbol{R}' * \boldsymbol{R}$, $\boldsymbol{R} * \boldsymbol{R}'$ and $\boldsymbol{R}$, the middle term of each triplet is between the other two (Mirkin 1979). A similar statement holds for the corresponding relations ρ', $\rho' * \rho$, $\rho * \rho'$ and ρ.

Q.3.29. Consider two rankings on an 8-element set, $\boldsymbol{R}'$= (1–2–3–4, 5–6–7–8) and $\boldsymbol{R}$ = (1–4–5, 2–3–7, 6–8). Here symbol – joins elements of the same part. Prove that $\boldsymbol{E}' \cap \boldsymbol{E}$ = {1–4, 2–3, 5, 7, 6–8} and give examples of rankings between $\boldsymbol{R}'$and $\boldsymbol{R}$.
A. For example, any ordering of $\boldsymbol{E}' \cap \boldsymbol{E}$ which is compatible with both $\boldsymbol{R}$ and $\boldsymbol{R}'$ as well as its further aggregations, say, $\boldsymbol{S}$ = (1-4, 2-3, 5, 6-8,7) and $\boldsymbol{T}$ = (1-4-2-3,5-6-8,7).

c. Correlation by Spearman and Kendall

Consider the Spearman rank correlation, that is, the Pearson correlation coefficient between ranks taken as numerical values. To deal with the case of tied rankings, each element of an equivalence class of the indifference relation is assigned with the average within-class rank. The average rank of the elements in part R_s of the tied ranking $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$ is $L + (|R_s| + 1)/2$, where L is the cardinality of $R_1 \cup R_2 \cup \ldots \cup R_{s-1}$, and $|\cdot|$ denotes the number of elements in a set. The *Kendall rank correlation* is based on the representation of tied rankings on I by N×N matrices. Given a tied ranking $\boldsymbol{R}$ and the corresponding preference relation $\rho = P \cup E$, we now define a skew-symmetric matrix $\mathbf{K} = (k_{ij})$, for $i, j \in I$, such that $k_{ij} = 1$ if $(i, j) \in P$, $k_{ij} = 0$ if $(i, j) \in E$, and $k_{ij} = -1$, if $(j, i) \in P$. The Kendall rank correlation coefficient between $\boldsymbol{R}$ and $\boldsymbol{R}'$ is the correlation coefficient between their Kendall matrices, $\mathbf{K}$ and $\mathbf{K}'$, considered as vectors in an N^2-dimensional space. This is compatible with the non-quantitative nature of tied rankings, especially since the mean of a skew-symmetric matrix is always 0.

It should be noted that, soon after the Kendall matrix was defined, a somewhat similar skew-symmetric representation for quantitative features was proposed by Daniels (1944), who proved that, given a quantitative feature x on I, the matrix $\mathbf{X} = (x_{ij})$, where $x_{ij} = x_i - x_j$, can be used to represent the feature in statistical computations. For example, the inner product of the matrices $\mathbf{X}$ and $\mathbf{X}'$ corresponding to features $\mathbf{x}$ and $\mathbf{x}'$ is proportional to the inner product of $\mathbf{x}$ and $\mathbf{x}'$ after they have been centered by subtracting their means, viz. $\langle \mathbf{X}, \mathbf{X}' \rangle = 2N\langle \mathbf{x} - m(\mathbf{x}), \mathbf{x}' - m(\mathbf{x}') \rangle$, where $m(\mathbf{x})$ is the mean of $\mathbf{x}$. This implies that the correlation coefficient between $\mathbf{X}$ and $\mathbf{X}'$ is equal to the correlation coefficient between $\mathbf{x}$ and $\mathbf{x}'$. Therefore, the Spearman rank correlation coefficient can also be defined as the Pearson correlation coefficient between the corresponding matrices of rank differences (r_{ij}), where $r_{ij} = r_i - r_j$. The Kendall matrix is then just the matrix of signs in the Daniels matrix $\mathbf{X} = (x_{ij})$, where $x_{ij} = x_i - x_j$.

d. **Kemeny distance**

Rather than defining an ad hoc distance measure, Kemeny formulated four axioms that should hold for any acceptable distance measure d($\boldsymbol{R}$, $\boldsymbol{R}'$) between rankings $\boldsymbol{R}$ and $\boldsymbol{R}'$. These axioms require that the acceptable distance measures should:

A1. Be mathematical metrics, that is, have the following properties:

(a) Symmetry: d($\boldsymbol{R}$, $\boldsymbol{R}'$) = d($\boldsymbol{R}'$, $\boldsymbol{R}$);
(b) Non-negativity and definiteness: d($\boldsymbol{R}$, $\boldsymbol{R}'$) $\geq$ 0 and d($\boldsymbol{R}$, $\boldsymbol{R}'$) = 0 if and only if $\boldsymbol{R} = \boldsymbol{R}'$;
(c) Strict triangle inequality: for any rankings $\boldsymbol{R}$, $\boldsymbol{R}'$ and $\boldsymbol{R}''$, d($\boldsymbol{R}$, $\boldsymbol{R}''$) $\leq$ d($\boldsymbol{R}$, $\boldsymbol{R}'$) + ($\boldsymbol{R}'$, $\boldsymbol{R}''$); moreover, equality holds if and only if $\boldsymbol{R}'$ is between $\boldsymbol{R}$ and $\boldsymbol{R}''$.

A2. If $\boldsymbol{R}'$ is obtained from $\boldsymbol{R}$ by a permutation of the set I and $\boldsymbol{S}'$ from S by the same permutation, then d($\boldsymbol{R}'$, $\boldsymbol{S}'$) = d($\boldsymbol{R}$, $\boldsymbol{S}$).

To formulate the next axiom, let us say that a subset $B \subset I$ is a *segment* of a tied ranking $\boldsymbol{R}$ if its complement $I - B \neq \varnothing$ and each element i $\in I - B$ either precedes all the elements of B or is situated after all the elements of B. The tied ranking $\boldsymbol{R}$ restricted to a segment B will be denoted by $\boldsymbol{R}_B$.

A3. If $\boldsymbol{R}$ and $\boldsymbol{R}'$ coincide on $I - B$ and B is a segment of both $\boldsymbol{R}$ and $\boldsymbol{R}'$, then d($\boldsymbol{R}$, $\boldsymbol{R}'$) = d($\boldsymbol{R}_B$, $\boldsymbol{R}_B{}'$).
A4. Unit of scale: The minimum positive distance is equal to 1.

Kemeny proved that the only distance satisfying all four axioms is the L_1-metric between the corresponding skew-symmetric Kendall matrices divided by 2 (Kemeny 1959), namely:

$$\mathrm{kd}(R, R') = \frac{1}{2}\sum_{i,j \in I} \left| k_{ij} - k'_{ij} \right| \tag{3.52}$$

We see from (3.52) that the pairs of elements (i, j) in I can be divided into three subsets:

(a) those contributing 1 to kd($\boldsymbol{R}$, $\boldsymbol{R}'$): pairs (i, j) such that i precedes j in either $\boldsymbol{R}$ or $\boldsymbol{R}'$ while j precedes i in the other;
(b) those contributing ½ to kd($\boldsymbol{R}$, $\boldsymbol{R}'$): pairs (i, j) such that i and j are indifferent in either $\boldsymbol{R}$ or $\boldsymbol{R}'$ whilst one precedes the other in the other ranking;
(c) those contributing 0 to kd($\boldsymbol{R}$, $\boldsymbol{R}'$): pairs (i, j), that are similarly related in both rankings—either i precedes j, or j precedes i, or i and j are indifferent.

e. The mismatch distance between binary relations and the corresponding binary matrices

Binary relations considered as subsets of the Cartesian product $I \times I$ may be compared using any of the many measures of dissimilarity between subsets that have been introduced over the years (Morlini and Zani 2012). One particularly simple measure is the number of pairs for which they differ, the so-called mismatch distance, i.e., the number of pairs in their symmetric difference:

$$d(\rho, \rho') = |(\rho - \rho') \cup (\rho' - \rho)| \tag{3.53}$$

The mismatch distance between unordered partitions was described in earlier publications by B. Mirkin in Russian from 1969 onwards [see, for example, (Mirkin and Cherny 1972)]; it is sometimes referred to as Mirkin's distance (Meilă 2007).

We note that $d(\rho, \rho')$ is a metric on the space of all binary relations on I and satisfies Axiom A1, including the strict triangle inequality, even for binary relations that do not correspond to tied rankings.

The mismatch distance can easily be translated into a distance between N×N matrices. Given a binary relation $\rho \in I \times I$, we define its binary matrix $\mathbf{r} = (r_{ij})$ by:

$$r_{ij} = \begin{cases} 1 & \text{if } (i,j) \in \rho \\ 0 & \text{if } (i,j) \notin \rho \end{cases}$$

Then the mismatch distance between $\boldsymbol{R}$ and $\boldsymbol{R}'$ is the mismatch (Hamming) distance between the corresponding binary relations ρ and ρ', and is thus given by:

$$\begin{aligned} d(R, R') &= d(\rho, \rho') = |(\rho - \rho') \cup (\rho' - \rho)| \\ &= \sum\nolimits_{i,j \in I} \left| r_{ij} - r'_{ij} \right| = \sum\nolimits_{i,j \in I} \left(r_{ij} - r'_{ij} \right)^2 \end{aligned} \tag{3.54}$$

The right-hand equality allows the original L_1-distance to be transformed into the square of the more conventional, Euclidean or L_2-distance because the absolute differences are either 1 or 0.

Obviously, the mismatch distance (3.54) is much simpler than the Kemeny distance (3.52) because the only possible non-zero contribution to $d(\boldsymbol{R}, \boldsymbol{R}')$ by an ordered pair (i, j) is 1, and this only occurs when j precedes i in one of the tied rankings but not in the other ranking. This happens when $r_{ij} = 0$ and $r_{ii}' = 1$ or, vice versa, $r_{ij}' = 0$ and $r_{ij} = 1$. It may therefore be somewhat of a surprise that these two distance measures are, in fact, equal, that is, the Kemeny distance (3.52) is equal to the mismatch distance (3.54).

To prove that, let us first analyse the contributions of pairs of elements $i, j \in I$ to the Kemeny distance between $\boldsymbol{R}$ and $\boldsymbol{R}'$ depending on their relative positions in the rankings $\boldsymbol{R}$ and $\boldsymbol{R}'$; the various cases are shown in Table 3.28. We note that the contribution of the pair (j, i) is exactly the same as that of the pair (i, j).

Table 3.28 The contribution of a pair (i, j) $\in I \times I$ to the Kemeny distance (3.52) between $\boldsymbol{R}$ and $\boldsymbol{R'}$

		$\boldsymbol{R}$		
	Cases	i precedes j	i and j are indifferent	j precedes i
R'	i precedes j	0	½	1
	i and j are indifferent	½	0	½
	j precedes i	1	½	0

Now we need to take into account a subtle difference between the concepts of ranking and preference relation. The Kemeny distance is between two rankings—it records disagreements in the relative positions between a pair of elements in the two rankings; the symmetry between i and j accounts for the factor ½ in the expression (3.52) for the Kemeny distance.

In contrast, the mismatch distance is between binary relations and counts the disagreements between the relations in respect of ordered pairs of elements. We, therefore, must distinguish between the ordered pair (i, j) and the inverse pair (j, i), relative to the corresponding relation, ρ or ρ′. The various cases of the contributions to the mismatch distance are shown in Tables 3.29 and 3.30, respectively.

Returning to the analysis of the interrelation between two elements i, j $\in I$, we need to combine Tables 3.29 and 3.30 by summing them, which produces Table 3.31.

Table 3.29 The contribution of the ordered pair (i, j) $\in I \times I$ to the mismatch distance (3.54) between $\boldsymbol{R}$ and $\boldsymbol{R'}$

		$\boldsymbol{R}$		
	Cases	i precedes j	i, j are indifferent	j precedes i
R'	i precedes j	0	0	1
	i and j are indifferent	0	0	1
	j precedes i	1	1	0

Table 3.30 The contribution of the ordered pair (j, i) $\in I \times I$ to the mismatch distance (3.54) between $\boldsymbol{R}$ and $\boldsymbol{R'}$

		$\boldsymbol{R}$		
	Cases	i precedes j	i, j are indifferent	j precedes i
R'	i precedes j	0	1	1
	i and j are indifferent	1	0	0
	j precedes i	1	0	0

Table 3.31 Summary contribution of the ordered pairs (i, j) and (j, i) to the mismatch distance (3.54)

		R		
	Cases	i precedes j	i, j are indifferent	j precedes i
R'	i precedes j	0	1	2
	i and j are indifferent	1	0	1
	j precedes i	2	1	0

If we double the values in Table 3.28 to account for both ordered pairs (i, j) and (j, i), we observe that the resulting entries are identical to those in Table 3.31, which completes the proof.

Consider a simple example where I consists of three elements, 1, 2, and 3 that are linearly ordered in $\boldsymbol{R}$ and all tied in $\boldsymbol{R}'$, so that $\boldsymbol{R}$ = ({1}, {2}, {3}) and $\boldsymbol{R}'$ = ({1, 2, 3}). Their respective Kendall matrices are

$$\mathbf{k} = \begin{matrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{matrix} \quad \text{and} \quad \mathbf{k}' = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix},$$

so that the Kemeny distance $\mathrm{kd}(R, R') = 6/2 = 3$.

On the other hand, their respective weak order matrices are

$$\mathbf{r} = \begin{matrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{matrix} \quad \text{and} \quad \mathbf{r}' = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix},$$

so that the mismatch distance $\mathrm{d}(R, R') = 3$ as well.

Q.3.30. Consider three rankings on a 7-element set I = {1,2,3,4,5,6,7}: R1 = (1–2–3, 4–5, 6–7), R2 = (2–3–6, 4, 1–5–7), and R3 = (1–2, 3–4, 5–6–7). Build their relation and Kendall matrices, and compute mismatch and Kemeny distances between all the three

f. **The mismatch distance expressed in terms of the contingency table**

Although the following results can be established directly, we now rely on Axiom A1(c), which states that $\mathrm{d}(\boldsymbol{R}, \boldsymbol{R'}) = \mathrm{d}(\boldsymbol{R}, \boldsymbol{R''}) + \mathrm{d}(\boldsymbol{R''}, \boldsymbol{R'})$ if and only if $\boldsymbol{R''}$ is between $\boldsymbol{R}$ and $\boldsymbol{R'}$, that is $\rho \cap \rho' \subseteq \rho'' \subseteq \rho \cup \rho'$ for the corresponding preference relations. By virtue of (3.5.4), we may use $\mathrm{d}(\boldsymbol{R}, \boldsymbol{R'})$ and $\mathrm{d}(\rho, \rho')$ interchangeably.

Let us first consider a tied ranking $\boldsymbol{R}$ and its reverse $\boldsymbol{R}^{-1}$. Obviously, $\rho \cap \rho^{-1} = E$, where E is the indifference relation of $\boldsymbol{R}$, which is an equivalence relation, stripped of all ranking information. Similarly, $\rho \cup \rho^{-1} = U$, the universal relation $U = I \times I$, which contains all possible ordered pairs of elements of I. Both E and U are, therefore, between ρ and ρ^{-1} for any weak order ρ.

Let N_s be the number of elements in part R_s of the tied ranking $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$. Then the mismatch distance between U and E is easily seen to be

$$d(E, U) = N^2 - \Sigma_s N_s^2, \tag{3.55}$$

since the first term on the right is the number of ones in the binary matrix of U and the second term is the number of ones in the binary matrix of E.

Curiously, the mismatch distance between U and $\boldsymbol{R}$ itself is exactly half the distance in (3.55). That is, the mismatch distance between a tied ranking $\boldsymbol{R}$ and the universal relation U is given by

$$d(R, U) = 1/2\left(N^2 - \Sigma_s N_s^2\right). \tag{3.56}$$

To prove that, we first notice that $d(\boldsymbol{R}, U) = d(\boldsymbol{R}^{-1}, U)$ and $d(\boldsymbol{R}, E) = d(\boldsymbol{R}^{-1}, E)$. Indeed, neither U nor E depend on the ranking information in $\boldsymbol{R}$, and, moreover, the number of pairs in ρ and ρ^{-1}, which is the number of ones in their respective matrices $\mathbf{r}$ and $\mathbf{r}^{-1}$, is the same. Since both U and E are between $\boldsymbol{R}$ and $\boldsymbol{R}^{-1}$, we have: $d(\boldsymbol{R}, \boldsymbol{R}^{-1}) = d(\boldsymbol{R}, U) + d(U, \boldsymbol{R}^{-1}) = 2d(\boldsymbol{R}, U)$ and $d(\boldsymbol{R}, \boldsymbol{R}^{-1}) = d(\boldsymbol{R}, E) + d(E, \boldsymbol{R}^{-1}) = 2d(\boldsymbol{R}, E)$.

This implies that $d(\boldsymbol{R}, U) = d(\boldsymbol{R}, E)$. So, since $\boldsymbol{R}$ is between E and U, $d(E, U) = d(E, \boldsymbol{R}) + d(\boldsymbol{R}, U) = 2d(\boldsymbol{R}, U)$. Equation (3.56) now follows from this and (3.55), which completes the proof.

We also have proved that the distance $d(\boldsymbol{R}, \boldsymbol{R}^{-1})$ is equal to $d(E, U)$ given by (3.55), whereas the distance $d(\boldsymbol{R}, E)$ is equal to $d(\boldsymbol{R}, U)$, given by (3.56).

Now we are in a position to prove a formula for the mismatch distance between a ranking $\boldsymbol{R}$ and its arbitrary refinement $\boldsymbol{R}'$. Like the previous results in this subsection, this does not depend on the ranking information.

Specifically, the mismatch distance between a ranking $\boldsymbol{R} = (R_1, R_2, \ldots, R_p)$ and its arbitrary refinement $\boldsymbol{R}' = (R_1', R_2', \ldots, R_q')$, where $q > p$, is given by

$$d(R, R') = \left(\Sigma_s N_s^2 - \Sigma_t N_t'^2\right), \tag{3.57}$$

where N_s and N'_t are the numbers of elements in the parts R_s of R and R_t' of $\boldsymbol{R}'$, respectively.

Indeed, since $\boldsymbol{R}$ is between $\boldsymbol{R}'$ and $\boldsymbol{U}$, we have $d(\boldsymbol{R}', \boldsymbol{U}) = d(\boldsymbol{R}', \boldsymbol{R}) + d(\boldsymbol{R}, \boldsymbol{U})$, so $d(\boldsymbol{R}', \boldsymbol{R}) = d(\boldsymbol{R}', \boldsymbol{U}) - d(\boldsymbol{R}, \boldsymbol{U})$. Both distances $d(\boldsymbol{R}', \boldsymbol{U})$ and $d(\boldsymbol{R}, \boldsymbol{U})$ are determined by Eq. (3.56), adjusted for the corresponding parts of $\boldsymbol{R}'$ and $\boldsymbol{R}$, respectively. This immediately yields (3.57), completing the proof.

Consider now two ordered partitions, $\boldsymbol{R}$ and $\boldsymbol{R}'$, and their lexicographic products $\boldsymbol{R} * \boldsymbol{R}'$ and $\boldsymbol{R}' * \boldsymbol{R}$. We shall show that the entire ranking component contributing to the distance between $\boldsymbol{R}$ and $\boldsymbol{R}'$ is accounted for by the distance between $\boldsymbol{R} * \boldsymbol{R}'$ and $\boldsymbol{R}' * \boldsymbol{R}$. First, consider the intersection $\boldsymbol{R} \cap \boldsymbol{R}'$, as presented in Fig. 3.27.

Letting $N_{st} = |R_s \cap R'_t|$, for $s = 1, 2, \ldots, p$ and $t = 1, 2, \ldots, q$, denote the numbers of elements in the parts of the intersection, we can present these cardinalities as a contingency table, or cross-classification, between $\boldsymbol{R}$ and $\boldsymbol{R}'$.

The distance between $\boldsymbol{R} * \boldsymbol{R}'$ and $\boldsymbol{R}' * \boldsymbol{R}$ is equal to half of the total of the products of the cardinalities of those parts in the intersection $\boldsymbol{R} \cap \boldsymbol{R}'$ for which the orderings in $\boldsymbol{R}$ and $\boldsymbol{R}'$ are contradictory:

$$d(\boldsymbol{R} * \boldsymbol{R}', \boldsymbol{R}' * \boldsymbol{R}) = \frac{1}{2}\Sigma_{s>s'}\Sigma_{t>t'}N_{st}N_{s't'} \tag{3.58}$$

Considering the rankings $\boldsymbol{R}$ and $\boldsymbol{R}'$ as unordered partitions, denoted above by $\check{\boldsymbol{R}}$ and $\check{\boldsymbol{R}}'$, respectively, the mismatch distance between the corresponding equivalence relations, E and E', can be expressed as

$$d(E, E') = \Sigma_s N_s^2 + \Sigma_t N_t'^2 - \Sigma_{s,t} N_{st}^2 \tag{3.59}$$

where N_s, N'_t, and N_{st} are, as above, the numbers of elements in parts R_s of $\boldsymbol{R}$, R'_t of $\boldsymbol{R}'$ and $R_s \cap R'_t$ of $\boldsymbol{R} \cap \boldsymbol{R}'$, respectively. The mismatch distance between tied rankings $\boldsymbol{R}$ and $\boldsymbol{R}'$ can be decomposed into ranking and equivalence parts as follows:

$$d(\boldsymbol{R}, \boldsymbol{R}') = \frac{1}{2}d(E, E') + d(\boldsymbol{R} * \boldsymbol{R}', \boldsymbol{R}' * \boldsymbol{R}). \tag{3.60}$$

To prove that, consider the corresponding binary relations ρ, ρ', and $\rho \cap \rho'$. Since the intersection $\rho \cap \rho'$ is between ρ and ρ', $d(\rho, \rho') = d(\rho, \rho \cap \rho') + d(\rho \cap \rho', \rho')$. On the other hand, $\rho * \rho'$ is between $\rho \cap \rho'$ and ρ, and $\rho' * \rho$ is between $\rho \cap \rho'$ and ρ', so $d(\rho, \rho \cap \rho') = d(\rho, \rho * \rho') + d(\rho * \rho', \rho \cap \rho')$ and $d(\rho \cap \rho', \rho') = d(\rho \cap \rho', \rho' * \rho) + d(\rho' * \rho, \rho')$. But $\rho \cap \rho'$ is between $\rho * \rho'$ and $\rho' * \rho$, so $d(\rho * \rho', \rho' * \rho) = d(\rho * \rho', \rho \cap \rho') + d(\rho \cap \rho', \rho' * \rho)$. Substituting these in the equation $d(\rho, \rho') = d(\rho, \rho \cap \rho') + d(\rho \cap \rho', \rho')$, we obtain

$$d(\rho, \rho') = d(\rho, \rho * \rho') + d(\rho * \rho', \rho' * \rho) + d(\rho' * \rho, \rho').$$

Since $\rho * \rho'$ is a refinement of ρ, and $\rho' * \rho$ is a refinement of ρ', $d(\rho, \rho * \rho') = 1/2\left(\Sigma_s N_s^2 - \Sigma_{s,t} N_{st}^2\right)$ and $d(\rho' * \rho, \rho') = 1/2\left(\Sigma_t N_t'^2 - \Sigma_{s,t} N_{st}^2\right)$. This implies, by (3.59), that $d(\rho, \rho * \rho') + d(\rho' * \rho, \rho') = \frac{1}{2}d(E, E')$. Together with the equation above, this completes the proof.

This section can be looked at as an attempt to find some structure in the -1 entries in the Kendall matrices occurring in the formula for the Kemeny distance between tied rankings. These entries appear whenever a pair of elements, i and j, are inversely related. First, we showed that the Kemeny distance can be expressed in terms of the mismatch distance between the preference relations (weak orders) corresponding to the rankings, in which no negative entries appear. The mismatch distance can be properly defined in terms of the non-negative 0−1 matrices of weak orders, rather than the Kendall matrices of the rankings, containing entries 1, 0 and -1.

Q.3.31. What is the meaning of the mismatch distance between partitions?
A. This is the probability of two random objects to belong to a same part in one partition and to different parts in the other partition.
Q.3.32. Frequently, when dealing with partitions, pairs of objects are considered as subsets rather than elements of the cartesian square of the set of objects. Then the number of pairs in set S is not $|S|^2$, but rather what is called the binomial coefficient $|S|(|S|-1)/2$, equal to the number of two-element subsets of S. Reformulate the expression for the mismatch distance for this case.

3.8 Decision Trees

3.8.1 General

Decision tree is a structure used for learning and predicting quantitative or nominal target features. In the former case it is referred to as a regression tree, in the latter, classification tree. This structure can be considered a multivariate extension of contingency tables in such a way that only meaningful combinations of feature categories are involved.

As illustrated on Fig. 3.28, a decision tree recursively partitions the entity set into smaller clusters by splitting a parental cluster over a single feature. The root of a decision tree corresponds to the entire entity set. Each node corresponds to a subset of entities, cluster, and its children are the cluster's parts defined by values of a single predictor feature x. Note that the trees on Fig. 3.28 are binary: each interior node is split in two parts. This is a most convenient format, currently used in most popular programs. Only binary trees are considered in this section.

Decision trees are built from top to bottom in such a way that every split is made to maximize the homogeneity of the resulting subsets with respect to a desired target feature. The splitting stops either when the homogeneity is enough for a reliable prediction of the target feature values or when the set of entities is too small to consider its splits reliable. A function scoring the extent of homogeneity to decide of the stopping is, basically, a measure of correlation between the partition of the entity set being built and the target feature.

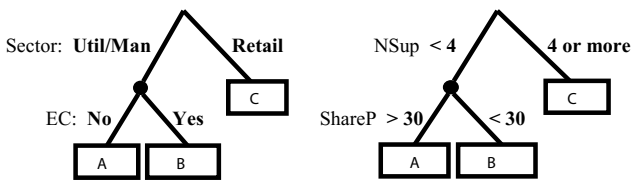

Fig. 3.28 Decision trees for three product based classes of Companies, A, B, and C, made using categorical features, on the left, and quantitative features, on the right

When the process of building a tree is completed, each terminal node is assigned with a value of the target that is determined to be characteristic for that node, and thus should be predicted at the conditions leading to the node. For example, both trees on Fig. 3.28 are precise—each terminal class corresponds to one and only one product, which is the target feature, so that each of the trees give a precise conceptual description of all products by conjunctions of the corresponding branch values. For example, product A can be described as that which is not in Utility sector, nor E commerce utilized in the production process (left-side tree) or as that in which less than 4 suppliers are involved and the share price is greater than 30. Both descriptions are fitting here since both give no errors at all.

Decision trees are very popular because they are simple to understand, use, and interpret. However, one should use them properly, because the decision rules produced with them can be overly simplistic and frequently imprecise. Their effectiveness much depends on the features and samples selected for the analysis. As always in learning correlation, a simpler tree is preferred to a complex one because of the over-fitting problem: a complex tree is more likely to reflect noise in the data rather than the true tendencies.

In the Sect. 3.6 we have described popular association indexes between partitions. Many of them are used as homogeneity scoring functions (to be) utilized in the process of classification tree building. This will be described next.

To build a binary decision tree, one needs the following information:

(a) a dataset of input features X,
(b) an output feature u over the set of objects,
(c) a scoring function $W(S,u)$ that scores admissible partitions S against the output feature,
(d) a rule for splitting a subset of objects, corresponding to the terminal node under consideration, in more homogeneous clusters,
(e) split-stopping criterion
(f) rule for pruning long or unreliable branches, and
(g) rule for the assignment of u-values to terminal nodes.

Let us comment on each of these items:

(a) The input features are, typically, quantitative or nominal. Quantitative features are handled rather easily by testing all possible splits of their ranges. More problematic are categorical features, especially those with many categories because the number of possible binary splits can be very large. However, this issue does not emerge at all if categorical features are preprocessed into the quantitative format of binary dummy variables corresponding to individual categories (which is advocated in this text too, see more detail in Sect. 2.3.2). Indeed, each of the dummy variables admits only one split—that separating the corresponding category from the rest, which reduces the number of possible splits to the number of categories—an approach advocated by Loh and Shih (1997), among many others. A number of such splits can be done in sequence to warrant that any combination of categories is admissible in this approach too.

Since this approach involves one feature at a time only, missing values are not of an issue here, because all the relevant information such as means and frequencies can be reasonably well estimated from those values that are available —this is a stark contrast with the other multivariate techniques.

(b) In principle, the decision tree format does not prevent from using multiple target features—just single-target criteria should be summed when there are several targets; this approach was successfully applied to sociology survey mixed scale data by P. Rostovtsev and B. Mirkin back in seventies (Mirkin 1985). However, all current internationally available programs involve only single target feature. Depending on the scale of the target feature, the learning task may differ, as well as the terminology. Specifically, if the target feature is quantitative, a decision tree is referred to as a regression tree, and if the target feature is categorical, a decision tree is referred to as a classification tree. Yet classification trees may differ with regards to the learning task: (a) learning a whole partition, if the target is nominal, or (b) learning just a category. This section focuses only on the task of learning a classification tree with a partitional target.

(c) Given a decision tree, its terminal nodes (leaves) form a partition S, which is considered then against the target feature u with a scoring function measuring the overall correlation $W(S,u)$. This suggests a context of the analysis of correlation between two features. If the target u is quantitative, then a tabular regression of u over S should be analyzed and scored. This approach involving the concept of the correlation ratio, as a natural scoring function, is described in the next section.

Unfortunately, in the data mining literature, this natural approach is not appreciated; thus, the correlation ratio is not popular. In contrast, at a categorical target, two most popular scoring functions, Gini index and Pearson chi-squared, fit perfectly in the framework of contingency tables and Quetelet indexes as described in Sect. 3.6.2. Moreover, it is mathematically proven in that section that these two can be considered as implementations of the same approach of maximizing the contribution to the data scatter of the target categories—the only difference being the way the dummy variables representing the categories are normalized: (i) no normalization to make it Gini index or (ii) normalization by Poissonian standard deviations so that less frequent categories get more important, to make it Pearson chi-squared. This sheds a fresh light on the criteria and suggests the user a way for choosing between the two indexes depending on user's preferences over the importance of being rare.

(d) Admissible partitions conventionally are obtained by splitting the entity subset corresponding to one of the current terminal nodes over one of the features. To make it less arbitrary, most modern programs do only binary splits. That means that any node may be split only in two parts: (i) that corresponding to a category and the rest, for a categorical feature or (ii) given an a, those "less than

a" and those "greater than a", for a quantitative feature. This text attends to this approach as well. All possible splits are tested and that split which leads to the largest value of the criterion is actually made, after which the process is reiterated.

(e) Stopping rule typically assumes a degree of homogeneity of sets of entities, that is, clusters, corresponding to terminal nodes and, of course, their sizes: too small clusters are not stable and should be excluded.

(f) Pruning: In some programs, the size of a cluster is unconstrained so that in the process of splitting nodes over features, some split parts may become very small and, thus, unreliable as terminal nodes. This makes it useful to prune the tree after it is computed, usually by merging the small subset nodes into greater agglomerations. This is typically done not according to the splitting criterion $W(S,u)$ but according to more local considerations such as testing whether proportions of the target categories in a cluster are similar to those used at the assignment of u values to terminal nodes or by removing nodes with small chi-squared values (see, for a review, Esposito et al. 1997).

(g) Assigning a terminal node with a u category conventionally is done by just averaging its values over the node entities if u is quantitative or according to the maximum probability of an u category. Then the quality of quantitative prediction is accessed, as usual, by computing the differences between observed and predicted values of u, and their variance of course. In the nominal target case, this leads to an obvious estimate of the probability of the error: unity minus the maximum probability; these then are averaged over the terminal nodes of the decision tree. To make the error's estimate more robust, cross-validation techniques are used. Consider, say, a tenfold cross validation. The entity set is randomly divided into ten equal-sized subsets. Each of them is used as a testing ground for a decision tree built over the rest: these errors are averaged and given as the error's estimate to the tree built over the entire entity set. These techniques are beyond the scope of the current text.

It should be mentioned that the assignment of a category to a terminal cluster in the tree can be of an issue in some situations: (i) if no obvious winning category occurs in the cluster, (ii) if the category of interest is quite rare, that is, when u's distribution is highly skewed. In this latter case using Quetelet coefficients relating the node proportions with those in the entire set may help by revealing some great improvements in the proportions, thus leading to interesting tendencies discovered.

3.8.2 *Three Approaches to Scoring Correlation for Decision Trees*

The process of building a classification tree is, basically, a process of splitting clusters into smaller parts driven by a measure of correlation between the partition

S being built and the target feature *u*. Since our focus here is the case of nominal *u*'s only, the target feature is represented by a partition *T* which is known to us on the training set.

How to define a function $w(S,T)$ to score correlation between the target partition *T* and partition *S* being built? Three possible approaches are:

1. A popular idea is to use a measure of uncertainty, or impurity, of a partition and score the goodness of split S by the reduction of uncertainty achieved when the split is made. If it is Gini index, or nominal variance, which is taken as the measure of uncertainty, the reduction of uncertainty is the popular impurity function utilized in a popular decision tree building program CART (Breiman et al. 1984). If it is entropy, which is taken as the measure of uncertainty, the reduction of uncertainty is the popular Information gain function utilized in another popular decision tree building program C3.5 (Quinlan 1993).
2. Another idea would be to use a popular correlation measure defined over the contingency table between partitions S and T such as Pearson chi-squared. Indeed Pearson chi-squared is used for building decision trees in one more popular program, SPSS (Green and Salkind 2003), as a criterion of statistical independence criterion, though, rather than a measure of association. Yet because Pearson chi-squared is equal to the summary relative Quetelet index (see Sect. 3.6.2), it is a measure association, and it is in this capacity that Pearson chi-squared is used in this text. Moreover, both the impurity function and Information gain mentioned above also are correlation measures defined over the contingency table as shown in the formulation part of this section. Indeed, the Information gain is just the mutual information between *S* and *T*, a symmetric function, and the impurity function, the summary absolute Quetelet index.
3. One more idea comes from the discipline of analysis of variance in statistics: the correlation can be measured by the proportion of the target feature variance taken into account by the partition *S*. How come? The variance is a property of a quantitative feature, and we are talking of a target partition here. The trick is that each class of the target partition is represented by the corresponding dummy feature, which is equal to 1 at entities belonging to the class and 0 at the rest. Each of them can be treated as quantitative, as explained in Sect. 3.6, so that the summary explained proportion would make a measure of correlation between *S* and *T*. What is nice in this approach, that it is uniform across different types of feature scales: both categorical and quantitative features can be treated the same, which is not the case with other approaches. Although this approach has been advocated by the author for a couple of decades already (see, for example, Mirkin 1996, 2012), no computational program has come out of that so far. There is a good news though: both the impurity function and Pearson chi-squared can be expressed as the summary explained proportion of the target variance, under different normalizations of the dummy variables course (see Sect. 3.6). To get the impurity function (Gini index), no normalization is needed at all, and Pearson chi-squared emerges if each of the dummies is normalized by

the square root of its frequency. That means that Pearson chi-squared is underlied by the idea that more frequent classes are less contributing. This might suggest the user to choose Pearson chi-squared if they attend to this idea, or, in contrast, the impurity function if they think that the frequencies of target categories are irrelevant to their case.

There have been developed a number of myths about classification tree building programs and correlation scoring functions involved in them. The following comments are purported to shed light on some of them.

Comment 1. Difference between CART and CHAID.

There is an opinion lurking in some comments on the web that of two popular programs, CART (Breiman et al. 1984) and CHAID (Green and Salkind 2003), the former is more oriented at prediction whereas the latter, at description. The reason for this perhaps can be traced to the fact that CART involves the impurity function that is defined as the reduction in uncertainty whereas CHAID involves Pearson chi-squared as a measure of the deviation from statistical independence. Yet this opinion is completely undermined by the fact that the measures have very similar predictive powers shaped as the summary Quetelet indexes, the only difference being that one of them involves the relative Quetelet indexes, and the other absolute ones (see Statements 3.5.3.1(b) and 3.5.3.2(b)).

Comment 2. Difference between Pearson chi-squared index and impurity function.

The difference between impurity function and Pearson chi-squared amounts to just different scaling options for the dummy variables representing classes of the target partition T (see items (c) in Statements 3.5.3.1 and 3.5.3.2). The smaller T classes get rescaled to larger values, thus contributing more, when using Pearson chi-squared.

Comment 3. Zeros in contingency tables.

Pearson chi-squared introduced to measure the deviation of a bivariate distribution from the statistical independence, appears also to signify a purely geometric concept, the contribution to the data scatter (see (a) and (c) in Statement 3.6.2.2 on p. 231). This leads to a different advice regarding the zeros in a contingency table. According to classical statistics, the presence of zeros in a contingency table contradicts the hypothesis of statistical independence so that the data are to be trimmed to avoid zeros. However, in the context of data scatter decompositions, the chi-squared is just a contribution with no statistical independence involved so that the presence of zeros is of no issue in this context: thus, no data trimming is needed.

Consider an entity set I with a pre-specified partition $T = \{T_l\}$—which can be set according to categories l of a nominal feature—that is to be learnt by producing a classification tree. At each step of the tree building process, a subset $J \subseteq I$ is to be

split into a partition $S = \{S_k\}$ in such a way that S is as close as possible to $T(J)$ which is the overlap of T and J. The question is: how the similarity between S and $T(J)$ is to be measured? When $S = T(J)$, there is no confusion between the two. Otherwise, it is the contingency table between S and $T(J)$, $P = (p_{kl})$ where p_{kl} is the proportion of J- entities in $S_k \cap T_l$, that expresses the confusion, which is why it is frequently referred to as a confusion table in this context.

One idea for assessing the extent of similarity is to use a correlation measure over the contingency table such as the averaged Quetelet coefficients, Q and A, or chi-squared X^2, as discussed in Sect. 3.6.2.

Seemingly another idea is to score the extent of reduction of uncertainty over $T(J)$ obtained when S becomes available. This idea works like this: take a measure of uncertainty of a feature, in this case partition $T(J)$, $\upsilon(T(J))$, and evaluate it at each of S-classes, $\upsilon(T(S_k))$, $k = 1,\ldots, K$. Then the average uncertainty on these classes will be $\sum_{k=1}^{K} p_{k+}\upsilon(T(S_k))$, where p_{k+} are proportions of entities in classes S_k, so that the reduction of uncertainty is equal to

$$\upsilon(T(J)/S) = \upsilon(T(J)) - \sum_{k=1}^{K} p_{k+}\upsilon(T(S_k)) \tag{3.61}$$

Of course, a function like (3.61) can be considered a measure of correlation over the contingency table P as well. One more nice feature of this approach is that it can be extended from nominal features to quantitative ones—just with an uncertainty index over quantitative T-features,

Two very popular measures defined according to (3.61) are the so-called impurity function and information gain. The impurity function builds on Gini coefficient as a measure of variance (see Sect. 3.6). Let us recall that Gini index for partition T is $G(T) = 1 - \sum_{l=1}^{L} p_l^2$ where p_l is the proportion of entities in T_l. If J is partitioned in clusters S_k, $k = 1,\ldots, K$, partitions T and S form a contingency table of relative frequencies $P = (p_{kl})$. Then the reduction (3.61) of the value of Gini coefficient due to partition S is equal to $\Delta(T(J), S) = G(T(J)) - \sum_k p_k G(T(S_k))$. This index $\Delta(T(J),S)$ is referred to as impurity of S over partition T. The greater the impurity the better the split S. As established in Statement 3.6.2.1 in Sect. 3.6.2.1, $\Delta(T, S) = A(T, S)$ where $A(T, S)$ is the summary absolute Quetelet index defined by Eq. (3.35) in Q.3.21, p. 225.

Three functions discussed above, Gini index, Pearson chi-squared, and Information gain can be coded as presented in columns of the box below where input p is a contingency table. Due to a holistic nature of MatLab computation, it is possible to organize the computation without looping through the matrix elements. The subroutines gini, chi and ing in the box can be considered pseudocodes of the functions for coding in any other language as well.

```
function a=gini(p)
  tot=sum(sum(p));
% total
  pr=p/tot;
  rp=sum(pr');
% row sums
  cp=sum(pr);
%column sums
  ps=pr.*pr;
  rps=sum(ps');
  ir=find(rp>0);
  tr=rps(ir)./rp(ir)
  a1=sum(tr);
  a2=sum(cp.*cp);
  a=a1-a2;
  return
```

```
function a=chi(p)
tot=sum(sum(p));
% total
  pr=p/tot;
  rp=sum(pr');
  cp=sum(pr);
  ir=find(rp>0);
 % nonzero rows
  ic=find(cp>0);
 %nonzero columns
  ps=pr.*pr;
  ip=rp'*cp;
  psi=ps(ir,ic);
  ipi=ip(ir,ic);
  tp= psi./ipi;
  a1=sum(sum(tp));
  a=a1-1;
  return
```

```
function a=ing(p)
 p=p+1;
% to avoid zeros
 tot=sum(sum(p));
 pr=p/tot;
 rp=sum(pr');
 cp=sum(pr);
 pl=log2(pr);
 pp=pr.*pl;
 rpp=sum(pp');
 a1=sum(rpp.*rp);
 tp=cp.*log2(cp);
 a2=sum(tp);
 a=a1-a2;
  return
```

Q.3.33. What is the formula of summary contribution B of partition S to the set of dummy features representing partition T when they have been normalized by dividing by their Bernoullian standard deviations $b_l = \sqrt{p_{+l}(1-p_{+l})}$?

Q.3.34. Consider a partition $S = \{S_k\}$ ($k = 1, 2, \ldots, K$) on J and a set of categorical features $v \in V$, each with a set of categories $L(v)$. The category utility function (Fisher 1987) scores partition S against the feature set according to formula:

$$u(S) = \frac{1}{K}\sum_{k=1}^{K} p_k \left[\sum_{v \in V} \sum_{l \in L(v)} p(v = l/S_k)^2 - \sum_{v \in V} \sum_{l \in L(v)} p(v = l)^2 \right] \tag{3.62}$$

The term in the square brackets is the increase in the expected number of attribute values that can be predicted given a class, S_k, over the expected number of attribute values that could be predicted without using the class. The assumed prediction strategy follows a probability-matching approach. According to this approach, entities arrive one-by-one in a random order, and the category l is predicted for them with the frequency reflecting its probability, $P(l/k)$ if the class S_k is known, or $p_k = N_k/N$ if information of the class S_k is not provided. Factors p_k weigh classes S_k according to their sizes, and the division by K takes into account the differences in the numbers of clusters: the smaller the better. Prove that the category utility function $u(S)$ is the sum of impurity functions $\Delta(l,S)$ over all features $l \in L$ related to the number of clusters, that is, $u(S) = \sum_{l \in L} \Delta(l, S)/K$.

3.8.3 Tabular Regression for Regression Trees and the Correlation Ratio

This section concerns yet another invention by Karl Pearson, the concept of piecewise constant regression and its version, tabular regression.

Consider x a categorical feature on the same entity set as a quantitative feature y, such as Occupation and Age at Students data set, or when building a regression tree, x being a partition to be built, and y, the target feature. The within-category distributions of y can be used to investigate the correlation between x and y.

The correlation between x and y is higher when the within-category spreads are tighter because the tighter the spread within an x-category, the more precise is prediction of the value(s) of y at it. Figure 3.29 illustrates an ideal case of a perfect correlation—all within-category y-values are the same leading to an exact prediction of Age when Occupation is known.

Figure 3.30 presents another extreme, when knowledge of an Occupation category does not lead to a better prediction of Age than when the Occupation is unknown.

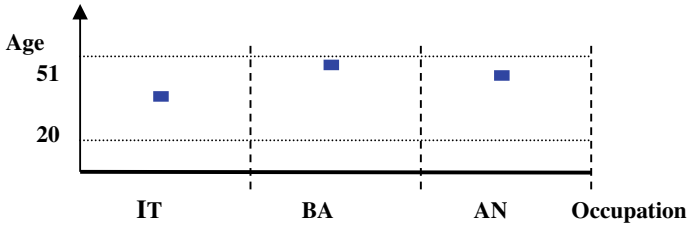

Fig. 3.29 In a situation of ideal correlation, with zero within-category variances, knowledge of the Occupation category would provide an exact prediction of the age within it

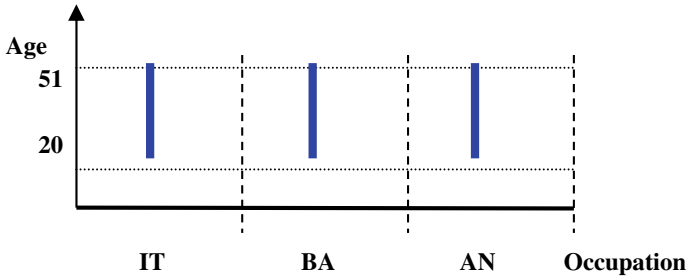

Fig. 3.30 Wide within-category distributions: the case of full variance within categories in which the knowledge of Occupation would give no information of age

A simple statistical model extending that for the mean will be referred to as tabular regression. The tabular regression of quantitative y over categorical x is a table comprising three columns corresponding to:

(1) Category of x
(2) Within category mean of y
(3) Within category standard deviation of y.

The number of rows in the tabular regression thus corresponds to the number of x-categories; there should be a marginal row as well, with the mean and standard deviation of y on the entire entity set.

Worked Example 3.14. Tabular Regression of Age (Quantitative Target) Over Occupation (Categorical Predictor) in Students Data
Let us draw a tabular regression of Age over Occupation in Table 3.32. The table suggests that if we know the Occupation category, say IT, then we can safely predict the Age as being 28.2 within the margin of plus/minus 5.6 years. With no knowledge of the Occupation category, we could only say that the Age is on average 33.7 plus/minus 8.5, a somewhat less precise estimate.

The table can be visualized in a manner similar to those in box-plots (see Fig. 3.31).

Table 3.32 Tabular regression of age over occupation in Students data

Occupation	Age mean	Age Std
IT	28.2	5.6
BA	39.3	7.3
AN	33.7	8.7
Total	33.7	8.5

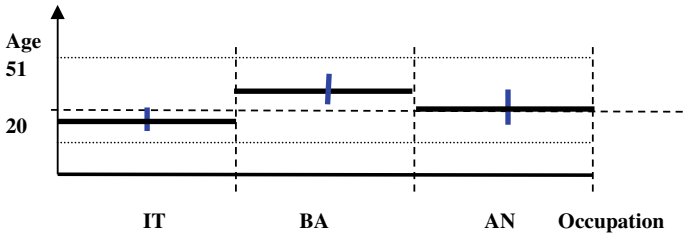

Fig. 3.31 Tabular regression visualized with the within-category averages and standard deviations represented by the position of solid horizontal lines and vertical line sizes, respectively. The dashed line's position represents the overall average (grand mean)

There is an integral characteristic of the tabular regression, the correlation ratio, which is akin to the determinacy coefficient at linear regression. This coefficient scores the extent at which the within group variance is smaller on average than the variance of the feature on the set before the split—a determinacy coefficient for the tabular regression.

Given a quantitative feature y, with no further information, its average, $\bar{y} = \sum_{i \in I} y_i / |I|$, would represent a proper summarization of the data. If, however, a set of categories of another variable, x, is additionally present, a more detailed summarization can be provided: the within category averages. Let S_k denote the set of entities falling in k category of x, then the within-category averages are $\bar{y}_k = \sum_{i \in S_k} y_i / |S_k|$.

This can be considered the least-squares solution to the model of tabular regression which extends the data recovery model for the average in Sect. 2.2.2 as follows. Find a set of c_k values such that the summary square error $L = \sum_{i \in I} e_i^2$ is minimized, where $e_i = y_i - c_k$ according to equations

$$y_i = c_k + e_i \text{ for all } i \in S_k \tag{3.63}$$

The equations underlie the tabular regression and are referred to sometimes as the piece-wise constant regression. It is not difficult to prove that the optimal c_k in (3.61) is the within category average $\bar{y}_k$, which implies that the minimum value of L is equal to $L_m = \sum_{k=1}^{K} \sum_{i \in S_k} (y_i - \bar{y}_k)^2$. By dividing and multiplying the interior sum by the number of elements in S_k, $|S_k|$, we can see that in fact $L_m = N\sigma_w^2$ where σ_w^2 is the average within category variance defined as

$$\sigma_w^2 = \sum_{K} p_k s_k^2 \tag{3.64}$$

where $p_k = |S_k|/N$ is the proportion of category k and σ_k^2 the variance of y within S_k.

To further analyze this, consider equation

$$(y_i - \bar{y}_k)^2 = y_i^2 + \bar{y}_k^2 - 2y_i\bar{y}_k$$

and sum it over all $i \in S_k$. This would lead to the summary right-hand item being similar to that in the middle, thus producing $\sum_{i \in S_k} (y_i - \bar{y}_k)^2 = \sum_{i \in S_k} y_i^2 - |S_k|\bar{y}_k^2$. Summing these equations over k and moving the right-hand item to the other side of the equation, would lead to the following decomposition:

$$\sum_{i \in I} y_i^2 = \sum_{k=1}^{K} |S_k|\bar{y}_k^2 + \sum_{k=1}^{K} \sum_{i \in S_k} (y_i - \bar{y}_k)^2 \tag{3.65}$$

Note that the right-hand item in (3.65) is the summary least-squares criterion of model in (3.63) L_m. This allows us to interpret the Eq. (3.65) as a decomposition of the scatter of variable y, the item on the left, in two parts on the right: the explained part, in the middle, and the unexplained part L_m.

The explained part sums contributions of individual categories k, $|S_k|\bar{y}_k^2$. The value of the contribution is proportional to both the category frequency and its squared value—the greater the better.

Another expression of decomposition (3.65) can be obtained under the assumption that variable y is centered, so that its mean is 0, by relating it to N:

$$\sigma^2 = \sum_{k=1}^{K} p_k \bar{y}_k^2 + \sum_{k=1}^{K} p_k \sigma_k^2 \tag{3.66}$$

where σ^2 is the variance of y, the item on the right the minimum value L_m/N from (3.64), and the item in the middle, the weighted summary squared distance between the grand mean $\bar{y} = 0$ and within-category means $\bar{y}_k$.

Equation (3.66) is very popular in statistics as the decomposition of the variance into the within-group variance, the item on the right, and the between-group variance, the item in the middle, as the base of a popular method for comparison of within-category means which is referred to as ANOVA (ANalysis Of VAriance). In the context of the tabular regression model (3.65) viewed as a data recovery model, the original decomposition (3.65) of the quantitative feature scatter into part explained by the nominal feature and part remaining unexplained is more appropriate. Viewed in this light, decomposition (3.66) shows that the category k contribution to the total variance of y is proportional to its frequency multiplied by the squared difference between within-category mean $\bar{y}_k$ and grand mean $\bar{y} = 0$.

The correlation ratio shows the relative drop in the variance of y when y is predicted according to model (3.63) or, in other words, the relative proportion of the explained part of the variance. The correlation ratio is usually denoted by η^2 and defined by the following formula:

$$\eta^2 = 1 - \sigma_w^2/\sigma^2 \tag{3.67}$$

The notation, η^2, is well accepted in the literature, although some authors prefer using the term, correlation ratio, for the squared root of η^2, but most authors leave the term as is.

The definition implies the following properties:

- The range of η^2 is between 0 and 1.
- Correlation ratio $\eta^2 = 1$ when all within-category variances σ_k^2 are zero (that is, when y is constant within each group S_k).
- Correlation ratio η^2 is about 0 when all σ_k^2 are of the order of σ_k^2.

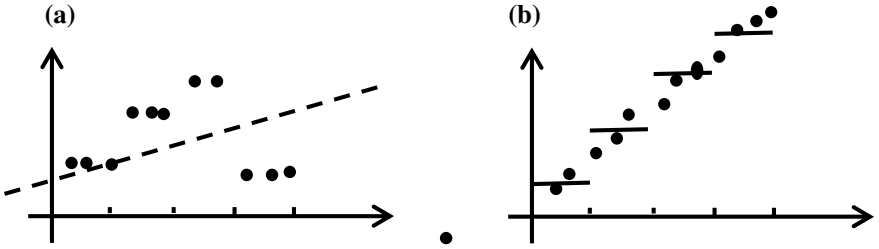

Fig. 3.32 Different patterns of linear and piecewise constant association between features corresponding to x-axis ad y-axis: almost perfect piecewise constant match against a highly non-linear pattern in (**a**), and almost linear arrangement against a highly non-constant pattern in (**b**)

Case-Study 3.7. Is There Any Relation Between Correlation Coefficient and Correlation Ratio?

Consider two quantitative features x and y. Divide the range of x in four equal-sized bins to produce a categorical variable xc. Is there any relation between the correlation coefficient between x and y and the correlation ratio coefficient between xc and y?

Some claim that η^2 should be always greater than ρ because the correlation ratio captures any type of functional relation whereas the correlation coefficient relates to linear functions only.

In general, no relation between η^2 and ρ can be claimed. The former can be greater than the latter in some cases, and smaller in some others, as presented in Fig. 3.32a at which $\eta^2 \gg \rho$ and 3.32b at which $\eta^2 \ll \rho$.

Q.3.35. Consider the variance to be an uncertainty measure for a quantitative feature y. Define the uncertainty reduction measure according to formula (3.16), with T changed for y of course, and prove that it is equal to the numerator of the correlation measure—the part of variance of y explained by its tabular regression over S.

A. The summary contribution of S to the data scatter is equal to $B = \sum_{k=1}^{K} c_k^2 |S_k| = \sum_{i \in I} y_i^2 - \sum_{k=1}^{K} \sigma_k^2 |S_k|$ where σ_k^2 is the within-cluster variance of y (see (3.13) in Sect. 3.2). Then $B = N\left(\sigma^2 - \sum_{k=1}^{K} p_k \sigma_k^2\right)$ where σ^2 is the variance of the standardized feature y (note that the mean of y is 0!) and p_k the proportion of entities in cluster S_k. The last equation clearly shows that the explained part of v is $B = N\sigma^2\eta^2$. If y has been z-score standardized so that $\sigma^2 = 1$, B equals the correlation ratio.

3.8.4 Building Classification Trees

Building of a classification tree is a recursive process: starting from the entire data set, partition a cluster into a number of parts according to one of the features. To make the partitions less arbitrary, only binary splits are involved in most of the

update programs. That means that any node may be split only in two parts: (i) that corresponding to a category and the rest, for a categorical feature, or (ii) given a threshold a, those "less than or equal to a" and those "greater than a", for a quantitative feature. This approach naturally comes when the data are preprocessed by "enveloping" categories into the corresponding "quantitative" dummy features, that assign a unity to every object falling into the category, and a zero to all the rest. Indeed, at $a = 0$, such a dummy feature would split the set in two parts—that for the corresponding category and the rest. Given a cluster, the choice of feature and threshold a for doing the split is driven by a correlation scoring function, be it Information gain, Pearson chi-squared, Gini index or anything else.

A cluster is not to be split anymore if it is smaller than a user defined threshold TS (TS = 10 is set further on) or is homogeneous enough. We use two different homogeneity tests: (a) large enough proportion of a target category in the cluster, say, above 80%, and (b) small enough value of the scoring function which is set to be 0.03 for Gini index, 0.08 for Pearson chi-squared, and 0.15 for Information gain. These levels of magnitude reflect the functions' ranges: Gini index is very close to 0 hardly reaching 0.5 at all, Pearson chi-squared, related to N, changes between 0 and 1 because it cannot be greater than the number of split parts minus 1, and Information gain can have larger values when the number of target categories is 3 or more. This sets the stopping conditions.

Worked Example 3.15 Classification Tree for the Iris Dataset

At Iris dataset with its three taxa, Iris setosa and Iris versicolor and Iris virginica, taken as target categories, all the three scoring functions—Impurity (Gini) function, Pearson chi-squared and Information gain—lead to the same classification tree, presented on Fig. 3.33.

The tree of Fig. 3.33 was found with program clatree.m. It comprises three leaf clusters: A, consisting of all 50 Iris setosa specimens; B, containing 54 entities of which 49 are of Iris versicolor and 5 of Iris virginica; C, containing 46 entities of which 45 are of Iris virginica and 1 of Iris versicolor. Altogether, this misplaces 6 entities leading to the accuracy of 96%. Of course, the accuracy would somewhat diminish if a cross-classification scheme is applied (see Loh and Shih 1997, who draw a slightly different tree for Iris dataset).

Let us take a look at the action of each variable at each of the two splits in Table 3.33. Each time features w3 and w4 appear to be most contributing, so that at

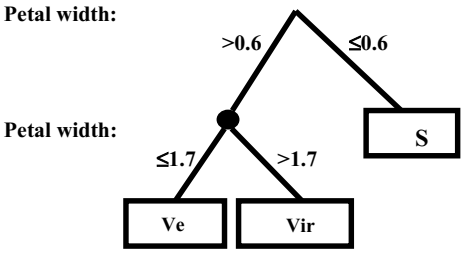

Fig. 3.33 Classification tree for the three-taxon partition at Iris dataset found by using each of the Gini, Pearson chi-squared and Information gain scoring functions

Table 3.33 Values of Gini index at the best split of each feature on the Iris dataset clusters in Fig. 3.33

	First split		Second split	
Feature	Value	Gini	Value	Gini
w1	5.4	0.228	6.1	0.107
w2	3.3	0.127	3.4	0.036
w3	1.9	0.333	3.7	0.374
w4	0.6	0.333	1.7	0.390

Table 3.34 Confusion tables between a split and target partition on the Iris dataset

Target partition classes	Iris setosa	Iris versicolor	Iris virginica	Total
Full set				
w4 $\leq$ 1.7	50	49	5	104
w4>1.7	0	1	45	46
Total	50	50	50	150
First cluster removed				
w4 $\leq$ 1.7	0	49	5	54
w4 > 1.7	0	1	45	46
Total	0	50	50	100

the first split, at which w3 and w4 give the same impurity value, w4 made it through just because it is the last maximum, which is remembered by the program.

The tree involves just one feature, w4: Petal width, used for splitting twice, first at w4 = 0.6 and then at w4 = 1.7. The Pearson chi-squared value (related to N of course) is 1 at the first split and 0.78 at the second. The Impurity function grows by 0.33 at the first split and 0.39 at the second. The fact that the second value is greater than the first one may seem to be somewhat controversial. Indeed, the first split is supposed to be the best, so that it is the first value that ought to be maximum. Nevertheless, this opinion is wrong: if the first split was at w4 = 1.7 that would generate just 0.28 of impurity value, less than the optimal 0.33 at w4 = 0.6. Why? Because the first taxon has not been extracted yet and grossly contributes to a higher confusion (see the top part in Table 3.34).

Project 3.3. Prediction of the Learning Outcome at Student Data Using Decision Trees

Consider the Student dataset and ask whether students' learning successes can be predicted from other features available (Occupation, Age, Number of children)? By looking at Table 1.5, one hardly can expect that marks can be predicted in this way. Therefore, let us divide students in three groups: I—not so good performers (average mark is less than 50), II—good performers (average mark between 50 and 70 inclusive), and III—excellent performers (average mark higher than 70). To do

this, we compute the average mark over the three subjects (SE, OOP, and CI) and create a partition of students *T* as described; the distribution of *T* appears to be I-25, II-58, III-17.

We have a 100×5 matrix X to explore the correlation between X and T, the three columns, 1,2,3, being dummy variables for Occupation categories (IT, BA, AN), column 4 for Age, and column 5 for Number of children. The two conventional stopping criteria, the cluster's size and prevalence of a target class, are not sufficient at this data, because after one or two splits, the program just chips away small fragments of clusters without much improving them. This corresponds to the situations at which the scoring function does not show much improvements either. Therefore, we utilize one more criterion—the minimum value of the scoring function, a threshold below which there is no splitting. Since the three scoring functions we use have different ranges, the thresholds must be different too. At this study, the threshold is set at 0.03 for the Gini index, 0.08 for the Pearson chi-squared, and 0.15 for the Information gain. The minimum cluster size is taken at 10, and the prevalence of a target class is set at 80%.

The classification tree found with Gini index is presented on Fig. 3.34. The distributions of target categories in clusters in Fig. 3.34 are presented in Table 3.35. Bold font highlights four terminal clusters as well as high or low proportions of target classes in clusters. High proportions here are those greater than 70% and low proportions are those smaller than 5%.

Tree on Fig. 3.34 is driven by two features: AN Occupation, that structures the set rather well—one split part, those of AN occupation, get more than 70% of category I, and none of category III, and the other of category II. All further divisions are over feature Age; the 12 students in cluster 8 are rather specific—these are of AN occupation aged between 22 and 28, so that 75% of them are in category I, an improvement over parental cluster 6. Cluster 4 of younger not-AN students seems an attempt at drawing a cluster to predict category III—it has a highest jump in its proportion, to 36.4% from 17% in the entire set (cluster 1). The 25 older people among not-AN students are overwhelmingly, 92%, in category II. More splits would have followed if we had decreased the minimum acceptable value of Gini index, say from 0.03 to 0.01.

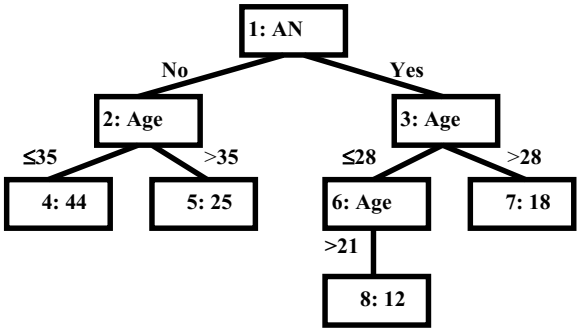

Fig. 3.34 Classification tree on Students data, targeting partition T of students in three categories, found using Gini index. The legend *Number: A* presents, at a split cluster labeled by Number, *A* as the split variable or, at an unsplit cluster, *A* as the size (the number of students in it)

Table 3.35 Distributions of target classes in clusters of tree on Fig. 3.34, %

Target categories	Clusters in tree on Figure 3.34							
	1	2	3	**4**	**5**	6	**7**	**8**
I	25.0	3.9	**73.2**	**3.3**	**3.0**	69.2	**77.8**	**75.0**
II	58.0	**73.5**	25.8	61.4	**93.0**	30.8	23.2	25.0
III	17.0	23.6	**0**	36.4	**3.0**	**0**	**0**	**0**
Gini index at split	0.168	0.046	0.035			0.048		
Cluster size	100	69	31	44	25	13	18	12

How well this tree would fare at prediction? To address this question properly, one should either conduct a cross-validation test or set aside a random testing set before using the rest for building a tree, after which see the levels of errors on the testing set.

Yet for the illustrative purposes, let us calculate the prediction error by using tree on Fig. 3.34. This is done by using the terminal clusters 4, 5, 7, 8 comprising 44, 25, 18, 12 elements, respectively. They total to 99, not 100, because of chipping off an element from cluster 6 to make it into cluster 7. That means: for students in AN category aged 21 or less, no prediction of their learning success level will be made; the classifier takes what is referred to as reject option (comprising approximately 1% of future cases if our sample is representative). According to the data in Table 3.35, the optimal prediction rule would predict then category II at Cluster 4 (with error 100 − 61.4 = 38.6%), category II again, at cluster 5 (with error 100 − 92 = 8%), and category I at clusters 7 and 8 (with errors 23.2 and 25.0%, respectively). The average error is the sum of the individual cluster errors weighted by their relative sizes, (38.6 * 44 + 8 * 25 + 23.2 * 18 + 25 * 12)/99 = 26.2%.

What happens, if we use the parental cluster 6 instead of the chipped cluster 8? First thing—no reject option is involved then. Second, the error somewhat increases as should be expected: (38.6 * 44 + 8 * 25 + 23.2 * 18 + 30.8 * 13)/100 = 27.0%.

Figure 3.35 presents trees found by using the Pearson chi-squared (a) and Information gain (b). In contrast to Gini index, decreasing the increment threshold

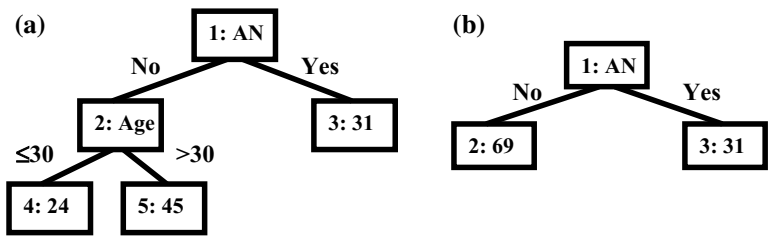

Fig. 3.35 Classification trees on Student dataset targeting partition T of students in the three categories, defined above, found using Pearson chi-squared (**a**) and Information gain (**b**). The legends are of format "Number: A" where "A", at a split cluster, is the split variable or, at an unsplit cluster, the cluster's size

does not much help at Information gain: chipping here and there rather than splits will be added. The change of splitting Age value to 30 at cluster 2 on tree (a) does lead to some improvements: the 45 older students are 83.2% in category II. Yet among the 24 younger students, 45.8% belong to category III (leaving 53.2% in category II and 0 in category I).

With this example, one can see that the 90–100% precision is not that easy to achieve. That is, a terminal node may have rather modest proportions of target categories, like cluster 5 on Fig. 3.35a: about 54% of II category and 46% of III category. Conventional thinking would label the node as an II category predictor because the share of II is greater than half.

However, one should note that, in fact, the proportion 54% is smaller than that, 58%, in the entire set, which means that in fact these conditions, Not_AN and younger age, less than 31, wash out some of II category. It is a case when the style of Quetelet's thinking may produce a better description. This thinking goes beyond proportions in the terminal node and requires comparing the category shares at the node with that in the whole sample. In contrast to a reduction of II category, this cluster boasts a dramatic increase of III category—from 17% in the entire set to 46% in the cluster, 29%. This difference would be picked up by using the absolute Quetelet coefficient which is equal to Gini index. Even more dramatic would be the relative increase, (45.8 − 17)/17 = 170%. It is this increase that has been picked up by Pearson chi-squared scoring function, because it is driven by the relative Quetelet coefficient.

Q.3.36. Drawing a lift chart in marketing research. Consider a marketing campaign advertising a product. There is a 1000 strong sample from the set of targeted customers whose purchasing behavior is known because of prior campaigns. The sample is composed of clusters of a classification tree with different response (that is, purchasing) rates (see Table 3.36). To plan an effective campaign, marketing researchers use what is called a lift chart—a visual representation of the response rate.

The x-axis of a lift chart shows the percentiles of the sample, say, from 10 to 100%. On y-axis, the so-called lifts are presented. Given a group of customers, the lift is defined as the ratio of the group's response rate to the baseline response rate, which is the response rate for the entire sample. On the lift chart, the percentiles of the sample are taken in the descending order of the lift. Both baseline and percentile lifts are presented on the chart. Build a lift chart for the sample. **A.** First, we calculate the baseline rate which is the average of the response rates in Table 3.36 weighted by the cluster proportions: $r = 0.1 * 30 + 0.4 * 10 + 0.25 * 4 + 0.25 * 0 = 8\%$.

Table 3.36 Proportions of four clusters in a sample of 1000 customers and their purchasing behavior (response rate)

Cluster share, %	10	40	25	25
Response rate, %	30	10	4	0

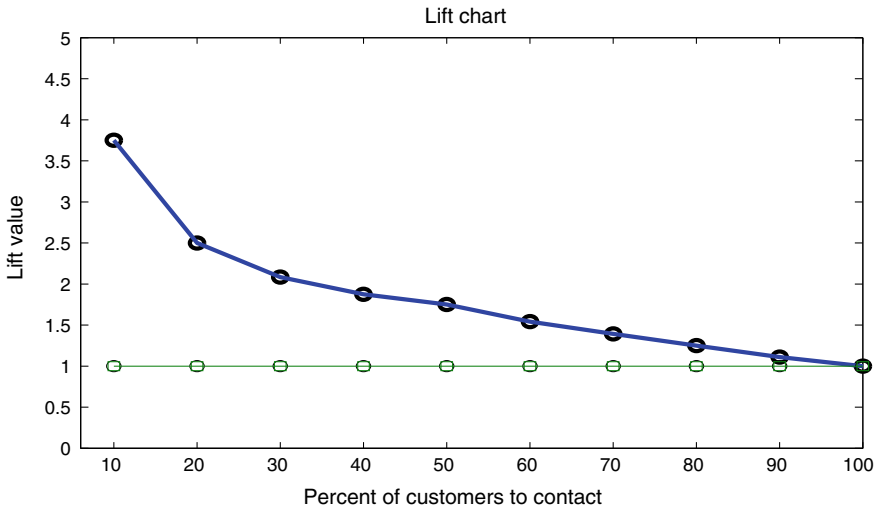

Fig. 3.36 Lift chart for data in Table 3.36

Now we take the most responsive 10% of the customers and calculate their lift value: 30/8 = 3.75. Next, we take the most responsive 20% of the sample, that is the first cluster plus a hundred customers from the second cluster and see their response rate—there should be 30 customers from the first cluster plus 10 from the second who have purchased the product, which gives 40/200 = 20% response rate leading to the lift value of 20/8 = 3.5. Next percentile, 30% of the sample is composed of the first cluster plus 200 customers from the second cluster leading to 50/300 = 17.7% response rate and lift 3.3. In this way, the chart presented on Fig. 3.36 is computed.

3.8.5 Building Classification Trees: Computation

Consider an entity set I with a nominal target feature represented by partition T of I as well as a set of quantitative input features X (some or all of X-features may be binary dummy variables corresponding to categories). At each step of the process of building a classification tree a cluster $J \subseteq I$ is to be split according to a

feature x_v from X in two clusters, S_1 and S_2 so that $S_1 = \{i|i \in J \text{ and } x_{iv} \leq a\}$ and $S_2 = \{i|i \in J \text{ and } x_{iv} > a\}$ where a is a value of x_v. The choice of x_v and a is guided by a scoring function $W(S,T)$ defined over the contingency table P cross-classifying T by S.

That implies that a cluster, as an element of the hierarchical structure being built, should maintain at least the following data:

(i) its entity set,
(ii) its parental cluster,
(iii) feature x_v over which it has been split,
(iv) splitting value a,
(v) the inequality, $\leq$ or $>$, in the cluster defining predicate.

The process starts at the universal cluster consisting of the entire set I. The process stops if either of two conditions holds:

(a) $|J| < n$, where n is a pre-specified threshold on the minimum number of entities in a cluster, and
(b) if the frequency of a T-cluster is greater than a pre-specified threshold α. To make testing of (b) easier, each cluster should bear one more feature

(vi) the distribution of T in it. One more useful piece of data supplied with a cluster would be
(vii) a signal of whether it may or may not be split again.

The recursive nature of the process, as well as the presence of a set of data to accompany each cluster, would make it a fitting subject of an object oriented code. Yet since the object oriented part of MatLab is not quite native in it, a procedural construction will be described in this section. This construction involves two parts, provided that computing scoring function $W(T,S)$ over contingency table P, has been implemented: (A) finding the best split over a feature, and (B) building a hierarchy of the best splits.

A pseudocode, or MatLab, function, msplit.m, takes in a column-feature x, partition of the set of its indices, t, as a cell array of t-classes, and a string with the name of a scoring method. It produces partition s, the feature splitting value y, and the value of scoring function ma. The stages of computation are annotated within the code.

```
function [g,ma,y]=msplit(x,t,method)

n=length(x);
%-------preparing the set of split value candidates
xv=union(x,x);%set of x values sorted
ll=length(xv);
rl=length(t);
if ll==1 %feature x is constant
    g{1}=[1:n];
    ma=0;
    y=max(x);
else
    for k=1:(ll-1) %loop over splitting values
        f{1}=find(x<=xv(k)); %first split set
        f{2}=setdiff([1:n],f{1}); % the rest
        for ik=1:2; for il=1:rl
              p(ik,il)=length(intersect(f{ik},t{il}));
            end
        end %  contingency table p
        switch method
            case 'gini'
                res=gini(p);
            case 'chi'
                res=chi(p);


           case 'ing'
               res=ing(p);
           otherwise
               disp('The method is wrong ');
               pause(10);
       end
   %----------looking for the best split
       if res>ma
           ma=res;
           g=f;
           y=xv(k);
       end
   end
end
```

The computation is organized in code clatree.m printed in the appendix. Here are just a few comments on its structure. Consider a set of ss clusters stored in a cell structure indexed from 1 to ss; in the beginning, the structure stores just the universal cluster I and its features at ss = 1. Of these clusters, those in the end, starting from index tt $\geq$ ss are eligible for splitting. The newly split clusters are indexed by index bb starting from bb = ss + 1. (Note that with this system of indexing, there is no need to assign clusters with a label informing that they should not be split anymore: the clusters to split can only be fresh ones!) After split parts are put in the structure, the indices are updated.

There can be a number of stopping criteria that are to be set in the very beginning of the program: it stops when no clusters eligible for splitting remain. In the current version of program clatree.m, three types of stopping criteria are employed. First is the number of entities, TS: a cluster with a smaller number of entities cannot make it into the tree and of course cannot be split further. Second, the dominant proportion of the target classes, ee: a cluster is not split anymore if this has been reached. And the third stopping criterion is tin, a threshold on the scoring function value: if it is less then tin at a split, the cluster is not split.

3.8.6 *Random Forest Classifier*

Although the decision tree structure is very good for interpretation, it is not as good, in general, for prediction; I guess, because of its instability. This is why one of the most prominent authors in the field proposed to diversify the tree structure by using a large number of decision trees generated by the bootstrap (Breiman 2001). To be more exact, let us specify a number P of bootstrap trials and generate, randomly with resampling, P series of the length N of indices from 1 to N, where N is the number of objects. Then we specify a number $m < V$ where V is the number of features and, for each of the P bootstrap series, generate a data table with objects corresponding to the indices in the series and k features randomly taken from the original V features. Then a decision tree is drawn based on each of the P data tables. In a refined version, the k-element feature subset is randomly drawn at each consecutive splitting step. This set of decision trees is what is called a random forest. Given an entity at which all the input variables are defined, one can decide over its class, in a classification problem, as follows. Use every tree in the forest, identify the location of the entity in the tree, and predict the target class accordingly. Count the number of votes, that is, the trees, for each of the target classes and predict that one with the largest number of votes. If the task at hand is not of classification but rather of prediction of the quantitative feature values, that is, of regression, the predicted outcome is the average of the values predicted by each of the regression trees in the forest.

This procedure works quite well, so that the random forest voting has become one of the most popular tools used by the practitioners.

However, the random forest concept loses its interpretability while gaining in the accuracy. This is why one needs a measure of feature importance according to a random forest, to compensate the loss albeit partly.

Consider a most popular feature importance measure, the valence (Breiman 2001; Louppe et al. 2013), which can be defined for any measure of impurity of a subset S with regard to a target feature u, $\gamma(S)$. (we skip the output partition in this notation). Limiting ourselves to the classification problem, that can be any measure considered in Sects. 3.6.1 and 3.6.2, as for example, Gini index. Then the impurity loss at partitioning a node s of the tree being built in its left part, s_L, and its right part, s_R, will be

$$\Delta\gamma(s) = \gamma(s) - p_L\gamma(s_L) - p_R\gamma(s_R) \tag{3.68}$$

where p_L and p_R are the proportions of S-entities in the left and right split parts, s_L and s_R, respectfully.

The importance of a variable w for predicting u is scored by summing the weighted impurity losses $p(s)\Delta\gamma(s)$, where $p(s)$ is the proportion of objects at the node s, for all nodes s at which w is used as the splitting criterion, averaged over all the trees in the forest:

$$\nabla_v(w) = \frac{1}{P}\sum_{t=1}^{P}\sum_{s_t \leftarrow w} p(s_t)\Delta\gamma(s_t) \tag{3.69}$$

Here the symbol $s_t \leftarrow w$ denotes the fact that the t-th tree node s_t involves splitting of the variable w, so that the second summation involves all the nodes s_t formed by splitting the variable w (t = 1,2,…, P).

Assume that all the features, both the target u and input V, are categorical and the decision trees in the ensemble are "fully developed and balanced" (see Louppe et al. 2013) so that all the divisions possible have been made—for the binary features that would mean that all the branches involve all the V features so that the tree has 2^V leaves. Then the importance weights can be estimated according to the following formulas. Given a subset W of k features, define

$$\gamma(u|W) = \sum_{w0} P(W = w0)\gamma(u|W = w0)$$

where P is the probability, and summation runs over all possible combinations $w0$ of the features in W. Given a feature v and subset of features W such that $v \notin W$, define $G(u|W + v) = \gamma(u|W) - \gamma(u|W + v)$. Then the following formula holds (Louppe et al. 2013):

$$\nabla_\gamma(w) = \sum_{k=0}^{V-1} \frac{1}{(V-k)C_V^k} \sum_{W \in P_k^{-v}} G(u|W + v) \tag{3.70}$$

Table 3.37 The table of digit-to-edge features e1–e7 according to the Fig. 1.2 in Sect. 1.2.3

Numeral	e1	e2	e3	e4	e5	e6	e7
0	1	1	1	0	1	1	1
1	0	0	1	0	0	1	0
2	1	0	1	1	1	0	1
3	1	0	1	1	0	1	1
4	0	1	1	1	0	1	0
5	1	1	0	1	0	1	1
6	1	1	0	1	1	1	1
7	1	0	1	0	0	1	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1
(3.70)	0.412	0.581	0.531	0.542	0.656	0.225	0.372
K = 1	0.414	0.583	0.532	0.543	0.658	0.221	0.368
K = 4	0.309	0.757	0.489	0.445	0.810	0.122	0.387

The bottom rows contain the feature importance weights computed according to f-la (3.70), as well as those estimated over a sample of 10,000 fully randomized trees, generated at k = 1 and k = 4, against the target feature Numeral, see on the left, by Louppe et al 2013

where P_k^{-v} is the set of all k-element subsets of features such that v is not among them, and C_V^k is the binomial coefficient.

Case-Study 3.8. Importance of Digit Features
Consider Table 3.37 of the ten numeral digits characterized by the seven binary features corresponding to the presence or absence of an edge in the numeral drawing over the seven-line rectangle in Fig. 1.2 in Sect. 1.2.3. The numerals themselves occupy the left-most column in the table.

The bottom rows contain the feature importance weights computed according to f-la (3.70), as well as those estimated over a sample of 10,000 fully randomized trees, generated at k = 1 and k = 4 according to the random forest algorithm, against the target feature Numeral. As one can see, the sampling estimates at k = 1 follow the theoretical estimates rather closely.

3.9 Naïve Bayes Approach

3.9.1 Bayes Decision Rule

Consider a situation in which there is only one target, a binary feature labeling two states of the world corresponding to "positive" and "negative" classes of entities. According to Thomas Bayes (c. 1701–1761), all relevant knowledge of the world should be shaped by the decision maker in the form of probability distributions. Then, whatever new data may be observed, they may lead to changing the

probabilities—hence the difference between prior probabilities and posterior, data-updated, probabilities. Specifically, assume that, $P(1) = p_1$ and $P(2) = p_2$ are prior probabilities of the two states so that p_1 and p_2 are positive and sum to unity. Assume furthermore that there are two probability density functions, $f_1(x_1, x_2, ..., x_p)$ and $f_2(x_1, x_2, ..., x_p)$, defining the generation of observed entity points $x = (x_1, x_2, ..., x_p)$ for each of the classes. That gives us, for any point $x = (x_1, x_2, ..., x_p)$ to occur, two probabilities, $P(1\&x) = p_1 f_1(x)$ and $P(2\&x) = p_2 f_2(x)$, of x being generated at either class. Therefore, the total probability of x to occur is $f(x) = p_1 f_1(x) + p_2 f_2(x)$. If an $x = (x_1, x_2, ..., x_p)$ is actually observed, it leads to a change in probabilities of the classes, from the prior probabilities $P(1) = p_1$ and $P(2) = p_2$ to posterior probabilities $P(1/x)$ and $P(2/x)$, respectively. These can be computed as conditional probabilities

$$P(1|x) = p_1 f_1(x)/f(x) \text{ and } P(2|x) = p_2 f_2(x)/f(x). \tag{3.71}$$

The decision of which class the entity x belongs to depends on what value, $P(1/x)$ or $P(2/x)$ is greater. The class is considered to be the positive if $P(1/x) > P(2/x)$ or, equivalently,

$$f_1(x)/f_2(x) > p_2/p_1 \tag{3.72}$$

or, the negative, if the reverse inequality holds. This rule is referred to as Bayes decision rule. Another expression of the Bayes rule can be drawn by using the difference $B(x) = P(1/x)\text{-}P(2/x)$ rather than the ratio: x is taken to belong to the positive class if $B(x) > 0$, and the negative class if $B(x) < 0$. Equation $B(x) = 0$ defines the so-called separating surface between the two classes.

The proportion of errors admitted by the Bayes rule is $1 - P(1/x)$ when 1 is predicted and $1 - P(2/x)$ when 2 is predicted. These are the minimum error rates achievable when both within-class distributions $f_1(x)$ and $f_2(x)$ and priors p_1 and p_2 are known.

Unfortunately, the distributions $f_1(x)$ and $f_2(x)$ are typically not known. Then some simplifying assumptions are to be made so that the distributions could be estimated from the observed data. Among most popular assumptions are:

(i) Gaussian probability and
(ii) Local independence.

Let us consider them in turn:

(i) Gaussian probability

The class probability distributions $f_1(x)$ and $f_2(x)$ are assumed to be Gaussian, so that each can be expressed as

$$f_k(x) = \left[\exp -(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k)/2\right]/\left[(2\pi)^p |\Sigma_k|\right]^{1/2} \tag{3.73}$$

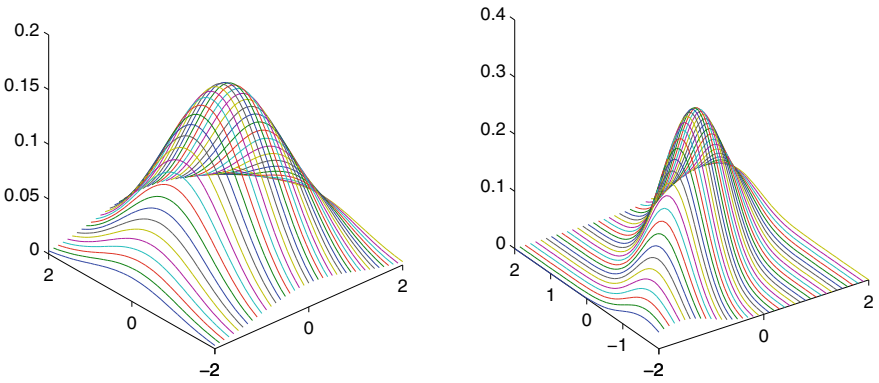

Fig. 3.37 Gaussian bivariate density functions over the origin as the expectation point—with zero correlation on the left and 0.8 correlation on the right

where μ_k is the central point, Σ_k the $p \times p$ covariance matrix and $|\Sigma_k|$ its determinant ($k = 1, 2$).

The Gaussian distribution is tremendously popular. There are at least two reasons for that. First, it is treatable theoretically and, in fact, may frequently lead to the least squares criterion within the probabilistic approach. Second, some real-world stochastic processes, especially in physics, can be thought of as having the Gaussian distribution. This is justified theoretically with the so-called probability limit theorems. These theorems state that the sum of a multitude of independent probabilistic distributions converges to a Gaussian distribution. Typical shapes of a 2D Gaussian density function are illustrated on Fig. 3.37: that with zero correlation on the left and 0.8 correlation on the right.

In the case at which the within-class covariance matrices are equal to each other, the Bayes decision function $B(x)$ is linear so that the separating surface $B(x) = 0$ is a hyperplane.

(ii) Local independence (Naïve Bayes)

The assumption of local independence states that all variables are independent within each class so that the within-cluster distribution is a product of one-dimensional distributions:

$$f_k(x_1, x_2, \ldots, x_p) = f_{k1}(x_1) f_{k2}(x_2) \ldots f_{kp}(x_p) \tag{3.74}$$

This postulate much simplifies the matters because usually it is not difficult to produce rather reliable estimates of the one-dimensional density functions $f_{kv}(x_v)$ from the training data. Especially simple such a task is when features x_1, x_2, ..., x_p are binary themselves. In this case the Bayes rule is referred to as a naïve Bayes rule because in most cases the assumption of independence (3.74) is obviously wrong in practical situations. Take, for example, the cases of text categorization or genomic

analyses—constituents of a text or a protein, serving as the features, are necessarily interrelated according to the syntactic and semantic structures, in the former, and biochemical reactions, in the latter. Yet the decision rules based on the wrong assumptions and distributions appear surprisingly good (see discussion in Manning et al. 2008).

Combining the assumptions of local independence and Gaussian distributions in the case of binary variables, one can arrive at equations expressing the conditional probabilities through exponents of linear functions of the variables (as described in Mitchell 2010) so that:

$$\begin{aligned} P(1/x) &= \frac{1}{1+\exp(c_0+c_1x_1+\ldots+c_px_p)}, \\ P(2/x) &= \frac{\exp(c_0+c_1x_1+\ldots+c_px_p)}{1+\exp(c_0+c_1x_1+\ldots+c_px_p)} \end{aligned} \tag{3.75}$$

Equations (3.75) express what is referred to as the logistic regression. Logistic regression is a popular decision rule that can be applied to any data on its own right as a model for the conditional probability, and not necessarily derived from the restrictive independence and normality assumptions.

3.9.2 *Naïve Bayes Classifier*

Consider a learning problem related to data in Table 3.38 further on: there is a set of entities, which are newspaper articles, divided into a number of categories—there are three categories in Table 3.38 according to the three subjects: Feminism, Entertainment and Household. Each article is characterized by its set of keywords presented in the corresponding line. The entries are either 0—no occurrence of the keyword, or 1—one occurrence, or 2—two or more occurrences of the keyword.

The problem is to form a rule according to which any article, including those outside of the collection in Table 3.38, can be assigned to one of these categories using its profile—the data on occurrences of the keywords in the corresponding line of Table 3.38.

Consider the Naïve Bayes decision rule. It assigns each category k with its conditional probability $P(k/x)$ depending on the profile x of an article in question according to equations in (3.71):

$$P(k/x) = p_k f_k(x)/f(x)$$

where $f(x) = \sum_l p_l f_l(x)$. According to the Bayes rule, the category k, at which $P(k/x)$ is maximum, is selected. Obviously, the denominator does not depend on k and can be removed: that category k is selected, at which $p_k f_k(x)$ is maximum.

Table 3.38 An illustrative database of 12 newspaper articles along with 10 keywords

Article	Keyword									
	Drink	Equal	Fuel	Play	Popular	Price	Relief	Talent	Tax	Woman
F1	1	2	0	1	2	0	0	0	0	2
F2	0	0	0	1	0	1	0	2	0	2
F3	0	2	0	0	0	0	0	1	0	2
F4	2	1	0	0	0	2	0	2	0	1
E1	2	0	1	2	2	0	0	1	0	0
E2	0	1	0	3	2	1	2	0	0	0
E3	1	0	2	0	1	1	0	3	1	1
E4	0	1	0	1	1	0	1	1	0	0
H1	0	0	2	0	1	2	0	0	2	0
H2	1	0	2	2	0	2	2	0	0	0
H3	0	0	1	1	2	1	1	0	2	0
H4	0	0	1	0	0	2	2	0	2	0

The articles are labeled according to their main subjects—F for feminism, E for entertainment, and H for household

According to the Naïve Bayes approach, $f_k(x)$ is assumed to be the product of the probabilities of occurrences of the keywords in category k. How can one estimate such a probability? This is not that simple as it sounds.

For example, what is the probability of term "drink" in H category according to Table 3.38? Probably, it can be taken as ¼—since the term is present in only one of four members of H. But what's about term "play" in H—it occurs thrice but in two documents only; thus its probability cannot be taken ¾; yet ²⁄₄ does not seem right either. A popular convention accepts the "bag-of-words" model for the categories. According to this model, all occurrences of all terms in a category are summed, to produce 31 for category H in Table 3.38. Then each term's probability in category k would be its summary occurrence in k divided by the bag's total. This would lead to a fairly small probability of the "drink" in H, just 1/31. This bias is not that important, however, because what matters indeed in the Naïve Bayes rule is the feature relative contributions, not the absolute ones.

And the relative contributions are all right with "drink", "fuel" and "play" contributing 1/31, 6/31 and 3/31, respectively, to H. Moreover, taking the total account of all keyword occurrences in a category serves well for balancing the differences between categories according to their sizes.

There is one more issue to take care of: zero entries in the training data. Term "equal" does not appear at all in H, leading thus to its zero probability in the category. This means that any article with an occurrence of "equal" would not be classed into H category, however heavy evidence from other keywords may be. One could make a point of course that term "equal" has not been observed in H just because the sample of four articles in Table 3.38 is too small, which is a strong argument indeed. To make up for these, another, a "uniform prior" assumption is

widely accepted. According to this assumption each term is present once at any category before the count is started. For the case of Table 3.38, this adds 1 to each numerator and 10 to each denominator, which means that the probability of "drink", "equal", "fuel" and "play" in category H will be (1 + 1)/(31 + 10) = 2/41, (0 + 1)/(31 + 10) = 1/41, (6 + 1)/(31 + 10) = 7/41 and (3 + 1)/(31 + 10) = 4/41, respectively.

To summarize, the "bag-of-words" model represents a category as a bag containing all occurrences of all keywords in the documents of the category plus one occurrence of each keyword, to be added to every count in the data table.

Table 3.39 contains the prior probabilities of categories, that are taken to be just proportions of categories in the collection, 4 of each in the collection of 12, as well as within-category probabilities of terms (the presence of binary features) computed as described above. Logarithms of these are given too.

Now we can apply Naïve Bayes classifier to any entity presented in the format of Table 3.38 including those in Table 3.38 itself (the training set). Because the probabilities in Table 3.39 are expressed in thousands, we may use sums of their logarithms rather than the probability products; this seems an intuitively appealing operation. Indeed, after such a transformation the score of a category is just the inner product of the row representing the tested entity and the feature scores corresponding to the category. Table 3.40 presents the logarithm scores of article E1 for each of the categories.

Q.3.37. Apply Naïve Bayes classifier in Table 3.39 to article X = (2 2 0 0 0 0 2 2 0 0) which involves items "drink", "equal", "relief" and "talent" frequently.
A. The category scores are: s(F/X) = 35.2, s(E/X) = 35.6, and S(H/X) = 29.4 pointing to Entertainment or, somewhat less likely, Feminism.
Q.3.38. Compute Naïve Bayes category scores for all entities in Table 3.38 and prove that the classifier correctly attributes them to their categories.
A. See Table 3.41

It should be mentioned that the Naïve Bayes computations here, as applied to the text categorization problem, follow the so-called multinomial model in which only terms present in the entities are considered—as many times as they occur. Another popular model is the so-called Bernoulli model, in which terms are assumed to be generated independently as binomial variables. The Bernoulli model based computations differ from these on two counts: first, the features are binary indeed so that only binary information, yes or no, of term occurrence is taken, and, second, for each term the event of its absence, along with its probability, is counted too (for more detail, see Manning et al. 2008; Mitchell 2010).

Table 3.39 Prior probabilities for Naïve Bayes rule for the data in Table 3.39 according to the bag-of-word conventions

Category	Prior probability its logarithm	Total count	Term counts Term probabilities (in thousands) Logarithms of the probabilities									
F	333	37	4	6	1	3	3	4	1	6	1	8
			108	162	27	81	81	108	27	162	27	216
	5.809		3.6	5.1	3.3	3.4	3.4	3.7	3.3	5.1	3.3	5.4
E	333	42	4	3	4	7	7	3	4	6	2	2
			95	71	95	167	167	71	95	143	48	48
	5.809		3.6	3.3	3.6	5.1	5.1	3.3	3.6	5	3.9	3.9
H	333	41	2	1	7	4	4	8	6	1	7	1
			49	24	171	98	98	195	146	24	171	24
	5.809		3.9	3.2	5.1	3.6	3.6	5.3	5	3.2	5.1	3.2

There are three lines for each of the categories representing, from top to bottom, the term counts from Table 3.38, their probabilities multiplied by 1000 and rounded to an integer, and the natural logarithms of the probabilities

Table 3.40 Computation of category scores for entity E1 (first line) from Table 3.38 according to the logarithms of within-class feature probabilities; the maximum score is highlighted by bold font

Entity E1		2	0	1	2	2	0	0	1	0	0	
Category	$Log(p_k)$	Feature weights (probability logarithms) Inner product										Score
F	5.809	3.6	5.1	3.3	3.4	3.4	3.7	3.3	5.1	3.3	5.4	35.2
		2 * 3.6	0	1 * 3.3	2 * 3.4	2 * 3.4	0	0	1 * 5.1	0	0	
E	5.809	3.6	3.3	3.6	5.1	5.1	3.3	3.6	5.0	3.9	3.9	**39.2**
		2 * 3.6	0	1 * 3.6	2 * 5.1	2 * 5.1	0	0	1 * 5.0	0	0	
H	5.809	3.9	3.2	5.1	3.6	3.6	5.3	5.0	3.2	5.1	3.2	33.5
		2 * 3.9	0	1 * 5.1	2 * 3.6	2 * 3.6	0	0	1 * 3.2	0	0	

There are two lines for each of the categories: that on top replicates the logarithms from Table 3.39 and that on the bottom computes the inner product

Table 3.41 Naïve Bayes category scores for the items in Table 3.38 with maxima highlighted using bold font

Articles	Category scores		
	F	E	H
F1	**37.7006**	35.0696	29.3069
F2	**28.9097**	25.9362	21.5322
F3	**23.9197**	20.1271	13.8723
F4	**38.276**	33.6072	30
E1	33.2349	**37.9964**	33.3322
E2	37.244	**43.1315**	40.2435
E3	43.1957	**43.5672**	40.8398
E4	21.1663	**23.9203**	19.4367
H1	25.8505	29.394	**33.5895**
H2	33.929	40.4527	**43.749**
H3	29.9582	35.3573	**38.3227**
H4	23.7518	28.8344	**33.8408**

3.10 Metrics of Accuracy

Consider a generic problem of learning a binary target feature, so that all entities belong to either class 1 or class 2. A decision rule, applied to an entity, generates a "prediction" which of these two classes the entity belongs to. The classifier may return some decisions correct and some erroneous. Let us pick one of the classes as that of our interest, say 1, then there can be two types of errors: false positives (FP)—the classifier says that an entity belongs to class 1 while it does not, and false negatives (FN)—the classifier says that an entity does not belong to class 1 while it does.

Let it be, for example, a lung screening device for testing against a lung cancer. Whilst established in a hospital cancer ward, on a selected sample of 200 patients sent by local surgeries for investigation, it may produce results that are presented in Table 3.42. Its rows correspond to the diagnosis by the screening device and the columns to the results of further, more elaborate and definitive, tests. This is a cross-classification contingency table, and it is frequently referred to as a confusion table.

There are 94 true positives TP and 98 true negatives TN in the table so that the total accuracy of the device can be rated as (94 + 98)/200 = 0.96 = 96%.

Table 3.42 Confusion table of patients' lung screening test results

		True lung cancer		Total
		Yes	No	
Device's diagnosis	Yes	94	7	101
	Not	1	98	99
Total		95	105	200

Table 3.43 Contingency table of volunteers' lung screening test results

		True lung cancer		Total
		Yes	No	
Device's diagnosis	Yes	2	2	4
	Not	1	195	196
Total		3	197	200

Respectively, the numbers of false positives FP = 7, and false negatives FN = 1 sum up to 8 leading to 4% error rate. Yet there are significant differences between these two showing that the device is in fact better than the totals show. Indeed, the 7 FP are not that important, because patients with the suspected cancer will be investigated further in depth anyway so that their No-status will be restored, with the cost of further testing. In contrast, 1 FN may go out of the medical system and get their cancer untreated with the potential loss of life because of the error. This is an example of different costs associated with FP and FN errors. The device made just one serious error out of 95 true cancer cases. The TP rate, the proportion of correctly identified true cases, frequently referred to as recall or sensitivity, 94/95 = 98.9%, is impressive indeed. On the other hand, the precision, that is, the proportion of the 94 TP cases related to all cancer predicted cases, 101, is somewhat smaller, just 93% to reflect that FP rate is 7%. The difference between precision and sensitivity is somewhat averaged in the value of accuracy rate, 96% in this case, so that the accuracy rate works reasonably well here as a single characteristic of the quality of the testing device.

Yet in a situation in which there is a great disparity in the sizes of Yes and No classes, the accuracy rate fails to reflect the results properly. Consider, for example, results of the same device at a random sample of 200 individuals who have not been sent for the screening by doctors but rather volunteered to be screened from public at large (Table 3.43).

The accuracy rate at Table 3.43 is even greater than that at Table 3.42, (2 + 195)/200 = 98.5%. Yet both sensitivity, 2/3 = 66.7%, and precision, 2/4 = 50%, are quite mediocre. The high accuracy rate is caused by the very high specificity, the proportion of correctly identified No cases, 195/197 = 98.9%, and by the fact that there are very few Yes cases.

As to a single measure adequately reflecting sensitivity and precision, the one most popular is their harmonic mean, the F-measure, which is equal to F = 2/(1/(2/3) + 1/(2/4)) = 2/(3/2 + 4/2) = 4/7 = 57.1%.

Case-Study 3.9. Prevalence and Quetelet Coefficients

If one looks at the record of the screening device according to Table 3.43, out of 4 cancer cases diagnosed, 2 are correct, and compares that with the prevalence of the cancer at the sample, 3 cases of 200—the difference is impressive indeed. This difference is exactly what is caught up in the concept of Quetelet coefficient $q(l/k)$ (see Sect. 3.6.1) at row $k = 1$ and column $l = 1$. This takes the relative difference between the conditional probability P(1/1) = 2/4 and the average probability

P(l = 1) = 3/200 which is referred to sometimes as the prevalence: q(1/1) = (2/4-3/200)/(3/200) = 2 * 200/(3 * 4) − 1 = 33.33 = 3333%, quite a change. This high value probably explains the difference in sensitivity and specificity between Tables 3.43 and 3.42.

Indeed, a similar Quetelet coefficient at Table 3.42 is q(1/1) = 94 * 200/(101 * 95) − 1 = 0.96 = 96%, a less than a 100% increase, which may convey the idea that Table 3.42 is much more balanced than Table 3.43. The accuracy measure works well at balanced tables and it does not at those that are not.

In general, the situation can be described by a confusion, or contingency, table between two sets of categories related to the class being predicted (1 or not) and the true class (1 or not), see Table 3.44 further on. Of course, if one changes the class of interest, the errors will remain errors, but their labels will change: false positives regarding class 1 are false negatives when the focus is on class 2, and vice versa.

Among popular indexes scoring the error or accuracy rates are the following:

FP rate = FP/(FP + TN)—the proportion of false positives among those not in 1; 1-FP rate is referred to sometimes as specificity—it shows the proportion of correct predictions among other, not class 1, entities.
TP rate = TP/(TP + FN)—the proportion of true positives in class 1; in information retrieval, this frequently is referred to as recall or sensitivity.
Precision = TP/(TP + FP)—the proportion of true positives in the predicted class 1.

These reflect each of the possible errors separately. There are indexes that try to combine all the errors, too. Among them the most popular are:

Accuracy = (TP + TN)/N—the total proportion of accurate predictions. Obviously, 1—Accuracy is the total proportion of errors.
F-measure = 2/(1/Precision + 1/Recall)—the harmonic average of Recall and Precision.

The latter measure is getting more popularity than the former because the Accuracy counts both types of errors equally, which may be at odds with the common sense in those frequent situations at which errors of one type are "more expensive" than the others. Consider, for example, the case of medical diagnostics in Tables 3.42, 3.43 and 3.44: a tumor wrongly diagnosed as malignant would cost much less than the other way around when a deadly tumor is diagnosed as benign. F-measure, to some extent, is more conservative because it, first, combines rates rather than counts, and, second, utilizes the harmonic mean which tends to be close to the minimum of the two, as can be seen from the statements in the next questions, Q.3.39 and Q.3.40.

Q.3.39. Consider two positive reals, a and b, and assume, say that $a < b$. Prove that the harmonic mean, $h = 2/(1/a + 1/b)$ stays within the interval between a and $2a$ however large the difference $b - a$ is.
A. Take b be $b = ka$ at some $k > 1$. Then $h = 2/(1/a + 1/(ka)) = 2ka/(1+ k)$. The coefficient at a, $2k/(1+ k)$, is less than 2, which proves the statement.

Table 3.44 A statistical representation of the match between the true class and predicted class. The entries are counts of the numbers of co-occurrences

		True class		
		1	Not	Total
Predicted Class	1	True Positives	False Positives	TP + FP
	Not	False Negatives	True Negatives	FN + TN
Total		TP + FN	FP + TN	N

Q.3.40. Consider two positive real values, a and b, and prove that their mean, $m = (a + b)/2$, and harmonic mean, $h = 2/(1/a + 1/b)$, satisfy equation $mh = ab$.
A. Take the product $mh = [(a + b)/2][2/(1/a + 1/b)]$ and perform elementary algebraic operations.

More elaborate representation of errors of the two types can be achieved with the so-called receiver operating characteristics (ROC) graph analysis (see, for example, Fawcett 2006). ROC graphs are especially suitable in the cases of classifiers that have a continuous output such as Bayes classifiers. ROC graph is a 2D Cartesian plane plotting TP rate against FP rate so that the latter is shown on x-axis and the former, on y-axis (see Fig. 3.38).

To be specific, let us take a Bayes classifier's rule in (3.66) and change the ratio p_2/p_1 for an arbitrary threshold $d > 0$. Take now $d = d_1$ for a specific d_1, so that the rule now predicts class 1 if $f1(x)/f2(x) > d_1$. Count the proportions of true and false positives, $tp1$ and $fp1$, at this threshold and put the point ($fp1$, $tp1$) onto a ROC graph. Then change d to d_2 and count the rates, $tp2$ and $fp2$, at this threshold. If, say, $d_2 > d_1$, then the TP rate can only decrease, because the number of positive

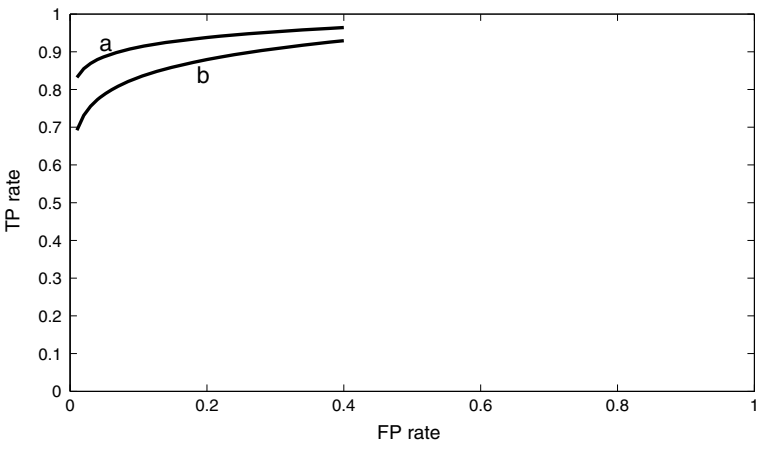

Fig. 3.38 ROC curves for two classifiers; that of a is superior to that of b

predictions can only decrease. The FP rate, in a regular case, should increase at $d_2 > d_1$ so that point ($fp2$, $ft2$) would go to the right and above the former point on the ROC plot. In this way, by step-by-step changing the threshold d, one can obtain a ROC curve such as curves "a" and "b" on the plot of Fig. 3.38. Such a curve can be utilized as a characteristic of the classifier under consideration that can be used, for instance, for selection of suitable levels of TP and FP rates. In the case shown on Fig. 3.38, one can safely claim that classifier "a" is superior to that of "b", because at each FP rate level, TP rate of "a" is greater than that of "b".

There is a quantitative characteristic of the quality of a ROC, called the area under the curve (AUC), which is indeed the area between the ROC and the FP-rate-axis: the greater the AUC, the better the classifier. The AUC expresses the probability that for a random pair of objects in which one is a "yes" and the second a "no" object, the classifier correctly predicts their belongingness to the "yes" and "no" classes. Experiments show that the AUC of 0.9 or greater can be considered an excellent one.

3.11 Summary

The goal of this chapter is to present a significant variety of techniques for learning correlation from data. Most popular concepts—regression, correlation, chi-squared, discrimination, Bayes classifiers, decision trees, neural networks, support vector machine, and correspondence analysis—are presented. Some of these are accompanied with concepts that are interesting on their own such as the bag-of-words model or kernel. The description, though, is rather fragmentary, except perhaps the classification trees for which a number of theoretical results is invoked to show their firm relations to bivariate analysis: first, summary Quetelet indexes in contingency tables and, second, normalization options for dummy variables representing target categories.

Overall, the chapter contents reflect basics of the art of learning correlations from data. Perhaps the subject is by far too complex and major advances are a matter of the future rather than the past. One such advance, though, should be mentioned here—deep learning based on neural networks with many layers (see, for example, Schmidhuber 2015).

The Chapter outlines several important characteristics of summarization and correlation between two features, and displays some of the properties of those. They are:

- linear regression and correlation coefficient for two quantitative variables;
- tabular regression and correlation ratio for the mixed scale case; and
- contingency table, Quetelet index, statistical independence, and Pearson's chi-squared for two nominal variables.

They all are applicable in the case of multidimensional data as well.

Some of the characteristics described here are rather unconventional. For example, the concepts of tabular regression and correlation ratio are not terribly popular in data mining. The Quetelet indexes are recognized by neither community, the more so the idea that Pearson chi-squared is a summary correlation measure, not necessarily a criterion of statistical independence.

Some examples of non-linear regression and nature-inspired approaches for fitting that are outlined. Computational bootstrap based validation is considered.

References

J.-P. Benzecri, *Correspondence Analysis Handbook* (CRC Press, 1992). ISBN 10 0824784375
M. Berthold, D. Hand, *Intelligent Data Analysis* (Springer, Berlin, 2003)
L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, *Classification and Regression Trees* (Wadswarth, Belmont, Ca, 1984)
A.C. Davison, D.V. Hinkley, *Bootstrap Methods and Their Application*, 7th edn. (Cambridge University Press, Cambridge, 2005)
H.B. Demuth, M.H. Beale, O. De Jess, M.T. Hagan, *Neural network design* (Martin Hagan, 2014)
R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification* (Wiley-Interscience, 2001). ISBN 0-471-05669-3
S.B. Green, N.J. Salkind, *Using SPSS for the Windows and Macintosh: Analyzing and Understanding Data* (Prentice Hall, 2003)
M. Greenacre, *Correspondence Analysis in Practice* (CRC Press, 2017)
P.D. Grünwald, *The Minimum Description Length Principle* (MIT Press, 2007)
J.F. Hair, W.C. Black, B.J. Babin, R.E. Anderson, *Multivariate Data Analysis*, 7th edn. (Prentice Hall, 2010). ISBN-10: 0-13-813263-1
J. Han, M. Kamber, J. Pei, *Data Mining: Concepts and Techniques*, 3rd edn. (Elsevier, 2011). ISBN: 978-9380931913
S.S. Haykin, *Neural Networks*, 2nd edn. (Prentice Hall, 1999). ISBN: 0132733501
A.Z. Israëls, *Eigenvalue Techniques for Qualitative Data* (Leiden, DSWO Press, 1987)
J. Kemeny, L. Snell, *Mathematical Models in Social Sciences* (New-York, Blaisdell, 1962)
M.G. Kendall, A. Stewart, *Advanced Statistics: Inference and Relationship*, 2nd edn. (Griffin, London, 1967)
L. Lebart, A. Morineau, M. Piron, *Statistique Exploratoire Multidimensionelle* (Dunod, Paris, 1995). ISBN 2-10-002886-3
C.D. Manning, P. Raghavan, H. Schütze, *Introduction to Information Retrival* (Cambridge University Press, Cambridge, 2008)
B. Mirkin, *Group Choice* (Halsted Press, Washington, DC, 1979)
B. Mirkin, *Grouping in Socio-Economic Research* (Finansy i Statistika Publishers, Moscow, Russia, 1985)
B. Mirkin, *Mathematical Classification and Clustering* (Kluwer, AP, Dordrecht, 1996)
B. Mirkin, *Clustering: A Data Recovery Approach* (Chapman & Hall/CRC, 2012). ISBN 978-1-4398-3841-9
F. Murtagh, *Correspondence Analysis and Data Coding with Java and R* (Chapman & Hall/CRC, Boca Raton, FL, 2005)
T.M. Mitchell *Machine Learning* (McGraw Hill, 2010)
S. Nishisato, *Elements of Dual Scaling: An introduction to practical data analysis* (Psychology Press, 2014)

B. Polyak, *Introduction to Optimization* (Optimization Software, Los Angeles, 1987). ISBN 0911575146
J.R. Quinlan, *C4. 5: Programs for Machine Learning* (Morgan Kaufmann, San Mateo, 1993)
B. Schölkopf, A.J. Smola, *Learning with Kernels* (The MIT Press, 2005)
V. Vapnik, *Estimation of Dependences Based on Empirical Data*, 2nd edn. (Springer Science + Business Media Inc., 2006)
A. Webb, *Statistical Pattern Recognition* (Wiley, 2002). ISBN-0-470-84514-7

Articles

L. Breiman, Random forests. Mach. Learn. **45**(1), 5–32 (2001)
J. Bring, How to standardize regression coefficients. Am. Stat. **48**(3), 209–213 (1994)
J. Carpenter, J. Bithell, Bootstrap confidence intervals: when, which, what? A practical guide for medical statisticians. Stat. Med. **19**, 1141–1163 (2000)
H.E. Daniels, The relation between measures of correlation in the universe of sample permutations. Biometrika, **33**(2), 129–135 (1944)
F. Esposito, D. Malerba, G. Semeraro, A comparative analysis of methods for pruning decision trees. IEEE Trans. Pattern Anal. Mach. Intell. **19**(5), 476–491 (1997)
T. Fawcett, An introduction to ROC analysis. Pattern Recogn. Lett. **27**, 861–873 (2006)
D.H. Fisher, Knowledge acquisition via incremental conceptual clustering. Mach. Learn. **2**, 139–173 (1987)
P.J.F. Groenen, G. Nalbantov, J.C. Bioch, SVM-Maj: a majorization approach to linear support vector machines with different hinge errors. Adv. Data Anal. Classif. **2**(1), 17–43 (2008)
J.G. Kemeny, Mathematics without numbers. Daedalus **88**(4), 577–591 (1959)
L. Lebart, B.G. Mirkin, Correspondence analysis and classification, in *Multivariate Analysis: Future Directions*, vol. 2 ed. by C. Cuadras, C.R. Rao (North Holland, 1993), pp. 341–357
Y. LeCun, Y. Bengio, G. Hinton, Deep learning. Nature **521**(7553), 436 (2015)
W.Y. Loh, Y.S. Shih, Split selection methods for classification trees. Stat. Sin. 815–840 (1997)
E. Lombardo, F. Beh, P. Kroonenberg, Modelling trends in ordered correspondence analysis using orthogonal polynomials. Psychometrika **81**(325–349), 2016 (2016)
G. Louppe, L. Wehenkel, A. Sutera, P. Geurts, Understanding variable importances in forests of randomized trees. in *Advances in Neural Information Processing Systems* (*NIPS*), (2013), pp. 431–439
M. Meilă, Comparing clusterings—an information based distance, J. Multivar. Anal. **98**(5), 873–895 (2007)
B. Mirkin, L. Cherny, Some properties of the partition space, in K. Bagrinovsky, E. Berland (Eds.), Math. Anal. Econ. Models III, Institute of Economics of the Siberian Branch of the USSR's Academy of the Sciences, Novosibirsk, 126–147 (1972)
B. Mirkin, Eleven ways to look at the chi-squared coefficient for contingency tables. Am. Stat. **55** (2), 111–120 (2001)
B. Mirkin, T.I. Fenner Tied rankings, ordered partitions, and weak orders: Distance and consensus. J. Classif. **36**(2), (2019)
J.N. Morgan, J.A. Sonquist, Problems in the analysis of survey data, and a proposal. J. Am. Stat. Assoc. **58**, 415–435 (1963)
I. Morlini, S. Zani, A new class of weighted similarity indices using polytomous variables. J. Classif. **29**(2), 199–226 (2012)
K. Pearson, On a criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen in random sampling. Phil. Mag. **50**, 157–175 (1900)
J. Schmidhuber, Deep learning in neural networks: an overview. Neural Netw **61**, 85–117 (2015)

K. Steele, H.O. Stefánsson, Decision Theory, The Stanford Encyclopedia of Philosophy (Winter 2015 edn.), Edward N. Zalta (Ed.), http://plato.stanford.edu/archives/win2015/entries/decision-theory/

N.G. Waller, J.A. Jones, Correlation weights in multiple regression. Psychometrika **75**(1), 58–69 (2010)