SubClass: Classification of Multidimensional Noisy Data Using Subspace Clusters

Ira Assent[1], Ralph Krieger[1], Petra Welter[1], Jörg Herbers[2], and Thomas Seidl[1]

[1] Data Management and Exploration Group
RWTH Aachen University
Aachen, Germany
{assent,krieger,welter,seidl}@cs.rwth-aachen.de
[2] INFORM GmbH
Pascalstraße 23
Aachen, Germany
joerg.herbers@inform-ac.com

Abstract. Classification has been widely studied and successfully employed in various application domains. In multidimensional noisy settings, however, classification accuracy may be unsatisfactory. Locally irrelevant attributes often occlude class-relevant information. A global reduction to relevant attributes is often infeasible, as relevance of attributes is not necessarily a globally uniform property. In a current project with an airport scheduling software company, locally varying attributes in the data indicate whether flights will be on time, delayed or ahead of schedule. To detect locally relevant information, we propose combining classification with subspace clustering (*SubClass*). Subspace clustering aims at detecting clusters in arbitrary subspaces of the attributes. It has proved to work well in multidimensional and noisy domains. However, it does not utilize class label information and thus does not necessarily provide appropriate groupings for classification. We propose incorporating class label information into subspace search. As a result we obtain locally relevant attribute combinations for classification. We present the SubClass classifier that successfully exploits classifying subspace cluster information. Experiments on both synthetic and real world datasets demonstrate that classification accuracy is clearly improved for noisy multidimensional settings.

1 Introduction

Data produced in application domains like life sciences, meteorology, telecommunication, and multimedia entertainment is rapidly growing, increasing the demand for data mining techniques which help users generate knowledge from data. Many applications require incoming data to be classified according to models derived from labeled historic data. In a current project, we investigate flight delays for airport scheduling purposes. The significance of flight delays can e.g. be studied in reports of the Bureau of Transportation Statistics in the U.S. [7] and the Central Office for Delay Analysis of Eurocontrol [11]. Extensive flight data is recorded by flight information systems at all major airports. Using such databases, we classify flights as on time, delayed or ahead

T. Washio et al. (Eds.): PAKDD 2008, LNAI 5012, pp. 40–52, 2008.

of schedule. This classification is essential in refining robust scheduling methods for airport resources and ground staff (like the one presented in [6]).

For classification, numerous techniques exist. For our noisy database that contains nominal attributes, numerical classifiers are not applicable. Neural networks or support vector machines do not allow users to easily understand the decision model for flight classification [20,17]. Bayes classifiers, decision trees, and nearest neighbor classifiers provide explanatory information, yet assume globally uniform relevance of attributes [20,18,3]. It has been shown that each type of classifier has its merit; there is no inherent superiority of any classifier [10]. However, classification is difficult in the presence of noise. Moreover, patterns may not show across all data attributes for all classes to be learned. In multidimensional data only a subgroup of attributes may be relevant for classification. This relevance is not globally uniform, but differs from class to class and from instance to instance.

We have validated the assumption of local relevance of attributes for the flight classification project by training several types of classifiers. When using only attributes which are determined as relevant by standard statistical tests, classification accuracy actually drops. This suggests that globally irrelevant attributes are nonetheless locally relevant for individual patterns. We therefore target at grouping flights with similar characteristics and identifying structure on the attribute level. In the flight domain, several aspects support the locality of flight delay structures. As an example, passenger figures may only influence departure delays when the aircraft is parked at a remote stand, i.e. when bus transportation is required. At some times of the day, these effects may be superposed by other factors like runway congestion. Weather conditions and other influences not recorded in the data cause significant noise.

Recent classification approaches like [9] use local weighting in nearest neighbor classification to overcome this drawback. Combing relevant attributes hierarchically a subspace is constructed for nearest neighbor classification. However, locally adaptive nearest neighbor methods do not consider the correlation of different attribute sets. Association rules have been extended to classification [16]. Recent approaches adopt subspace clustering methods to identify relevant subspaces for rule based classification [21].

In this work, we propose a nearest neighbor classifier which directly uses the result of our new subspace clustering method. Note that our approach is different from semi-supervised learning where unlabeled data is used for training [22]. Our approach assumes class labels that are directly incorporated into subspace clustering. Clustering is helpful for understanding the overall structure of a data set. Its aim is automatic grouping of the data in absence of any known class labels in historic data [13]. Since class labels are not known in advance (“unsupervised learning”), they are not used to classify according to given groupings (“supervised learning”). Hence clustering is not appropriate for classification purposes by its very nature [13]. However, the structures detected by clustering may be helpful for detecting local relevance of attributes. For noisy and high-dimensional data, clustering is often infeasible as clusters are hidden by irrelevant attributes. Different attribute combinations might show different clustering structures, thus the aim of subspace clustering is to detect clusters in arbitrary projections (“subspaces”) of the attributes. As the number of subspaces is exponential in the number of

attributes, most approaches try to prune the subspace search space [1,8,4]. Subspace clustering has been shown to successfully detect locally relevant attribute combinations [15,5].

We propose combining both worlds, supervised learning and unsupervised learning by incorporating class label information into subspace search and clustering. Classification based on these classifying subspace clusters exploits both class and local correlation information. The flight classification problem is used to evaluate our model. Its applicability, however, goes beyond this scenario. In fact, there are many more application areas where classification has to handle noisy multidimensional data with locally relevant attributes.

This paper is structured as follows: we define interesting subspaces for subspace classification in Section 2.1. Classifying subspace clusters and the overall classification scheme are discussed in Sections 2.2 and 2.3, respectively. Algorithmic concepts are presented in Section 3. The proposed method is evaluated in the experiments in Section 4 on both synthetic and flight data, before concluding the paper.

2 Subspace Classification

Subspace clustering is a recent research area which tries to detect local structures in the presence of noise or high-dimensional data where meaningful clusters can no longer be detected in all attributes [1,8,15]. As searching all possible subspaces is usually intractable, subspace clustering algorithms try to focus on promising subspace regions. The challenge is a suitable notion of interestingness for subspaces to find all relevant clusters. Subspace clustering is a technique well-suited to identify relevant regions of historic data, however, it is not suited for classification "as is". Our classification approach is capable of exploiting local patterns in the data for classification. This requires detecting subspaces and subspace clusters that are also based on class structure. Our $SubClass$ model thus comprises three steps:

- Step 1: **interesting subspaces** for classifying clusters: Section 2.1
- Step 2: **classifying subspace clusters**: Section 2.2
- Step 3: a **classification** scheme: Section 2.3.

2.1 Step 1: Interesting Subspaces

Interesting subspaces for classifying clusters exhibit a clustering structure in their attributes as well as coherent class label information. Such a structure is reflected by homogeneity in the attribute values or class labels of that subspace. Homogeneity can be measured using Shannon Entropy [19], or entropy for short. From an information theoretic perspective, Shannon entropy is the minimum number of bits required for encoding information. More frequently occurring events are encoded with fewer bits than less frequent ones. The sum over logarithmic probabilities weighted by their probability, measures the amount of information, i.e. the heterogeneity of the data.

Definition 1. ***Shannon Entropy.*** *Given a random variable X and its possible events $v_1, .., v_m$ the Shannon Entropy $H(X)$ is defined as:*

$$H(X) = -\sum_{i=1}^{m} p(v_i) \cdot \log_2 p(v_i)$$

Transferring the entropy notion to the clustering or classification domain, an attribute can be seen as a random variable whose domain is the set of all possible events. In case of continuous domains, the entropy requires discretization of attributes. Entropy according to a set of attributes with respect to a set of class labels is then:

Definition 2. ***Attribute Entropy.*** *Given a set of attributes $X_1, \ldots, X_m$, their possible values $v_1, \ldots, v_m$, and class labels $C = \{c_1, \ldots, c_n\}$, attribute entropy is defined as:*

$$H(X_1, \ldots, X_m|C) = -\sum_{c_i \in C} \sum_{v_1 \in X_1} \cdots \sum_{v_m \in X_m} p(c_i) \cdot H(X_1, \ldots, X_m|C = c_i)$$

Attribute entropy is thus the sum over all conditional attribute entropy value combinations weighted by the class label probabilities. It is a measure for the clustering tendency for all class labels c_i of a subspace in terms of the attributes. To measure the clustering tendency in terms of individual class labels, we define class entropy according to conditional entropy $H(C|X)$ (as e.g. in [18]).

Definition 3. ***Class Entropy.*** *Given a set of attributes $X_1, \ldots, X_m$, their possible values $v_1, \ldots, v_m$, and class label C the conditional entropy of a segmentation along these attribute values is defined as:*

$$H(C|X_1, \ldots, X_m) = -\sum_{v_1 \in X_1} \cdots \sum_{v_m \in X_m} p(v_1, .., v_m) \cdot H(C|X_1 = v_1, \ldots, X_m = v_m)$$

Class entropy is thus the sum over all conditional class entropy value combinations for individual class labels C. It corresponds to investigating the data for individual classes instead of aggregated as for attribute entropy.

We are interested in subspaces that exhibit both a distinct class structure as well as a clear clustering structure. Since entropy measures homogeneity, we are interested in low entropy values that reflect a non-uniform distribution of class or attribute values.

However, comparing subspaces using entropy is clearly biased with respect to the number of attributes. Subspaces with more attributes typically have lower entropy values. This is due to the fact that with increasing attribute number, objects tend to be less similar: each attribute contributes potential dissimilarity [5]. Thus, we have to normalize entropy with respect to the number of attributes. Normalization to a range of [0,1] can be achieved by taking the maximum possible entropy value for a given number of attributes into account. Maximum entropy means all values are equally likely, i.e. a uniform distribution. $H_{uniform}(X_1, .., X_m|C)$ for $d = |X_1 \times \cdots \times X_m|$ possible attribute combinations is determined as: $H_{uniform}(X_1, .., X_m|C) = -d \cdot \frac{1}{d} \cdot \log_2 \frac{1}{d} = -\log_2 \frac{1}{d} = \log_2 d$, since in uniform distribution, each attribute value occurs $1/d$ times. For larger numbers of attributes, the theoretical upper bound of $\log_2 d$ cannot be reached, as the actual number of instances is smaller than the number of possible attribute value combinations d.

To account for this, we the number of instances $|I|$ is used in this case:

$$H_N(X_1, .., X_m|C) = \frac{H(X_1, .., X_m|C)}{\min\{log_2|I|, \log_2 d\}}$$

In a similar spirit, we use the overall class distribution to normalize class entropy:

$$H_N(C|X_1, .., X_m) = \frac{H(C|X_1, .., X_m)}{H(C)}$$

Since those subspaces are interesting that cover both aspects, we define interestingness as a convex combination of attribute and class entropy, provided that each of the two is within reasonable bounds:

Definition 4. ***Subspace Interestingness.*** *Given attributes $X_1, \ldots, X_m$, a class attribute C, and a weighting factor $0 \leq w \leq 1$, a subspace is interesting with respect to thresholds β, λ iff:*

$$w \cdot H_N(X_1, .., X_m|C) + (1-w) \cdot H_N(C|X_1, .., X_m) \leq \beta$$
$$\wedge\, H_N(X_1, .., X_m|C) \leq \lambda \;\wedge\; H_N(C|X_1, .., X_m) \leq \lambda$$

Thus, a subspace is interesting for subspace classification if it shows low normalized class and attribute entropy as an indication of class and cluster structure. w allows assigning different weights to these two aspects for different applications, while λ is set to fairly relaxed threshold values to ensure that both aspects fulfill minimum entropy requirements.

2.2 Step 2: Classifying Subspace Clusters

Having defined interesting subspaces, the next step is detecting *classifying subspace clusters*. On discretized data, clusters can be defined as frequent attribute value combinations. To incorporate class information, these groupings should be homogeneous with respect to class label. We defined the absolute frequency

$$AbsFreq(v_1, \ldots, v_m) = |\{o, o|_S = (v_1, \ldots, v_m)|$$

as the number of objects o which exhibit the attribute values $(v_1, \ldots, v_m)$ in subspace S (projection $o|_S$ contains those attribute values v_i from o where $X_i \in S$).

To ensure that non-trivial clusters are mined, we normalize frequency with respect to the expected frequency of uniformly distributed subspaces. The expected frequency

$$ExpFreq(v_1, \ldots, v_m) = AbsFreq(v_1, \ldots, v_m) * d/|I|$$

is the number of cluster objects in comparison to the number of instances $|I|$ per attribute combination under uniform distribution. Classifying subspace clusters exceed minimum frequency for both absolute and relative (expected) frequency. Note that minimum absolute frequency simply ensures that a cluster exceeds a minimum size even for very small expected frequency values:

Definition 5. ***Classifying Subspace Cluster.*** *Given a subspace S of attributes $X_1, \ldots, X_m$, a classifying subspace cluster SC with respect to attribute values $v_1, \ldots, v_m$, minimum frequency thresholds ϕ_1, ϕ_2, and maximum entropy γ is defined as follows:*

- $H_N(C|X_1 = v_1, \ldots, X_m = v_m) \leq \gamma$
- $AbsFreq(v_1, \ldots, v_m) \geq \phi_1$
- $ExpFreq(v_1, \ldots, v_m) \geq \phi_2$

Classifying subspace clusters have low normalized class entropy, as well as high frequency in terms of attribute values. Thus, they are homogeneous in terms of class and show local attribute correlations.

2.3 Step 3: Classification

Classification of a given object o is based on the class label distribution of similar classifying subspace clusters. For nominal values as they occur in our flight data, an object o is typically contained in several subspace clusters and similarity is reduced to containment. Let $CSC(o) = \{SC_i | v_k = o_k \forall v_k \in SC_i\}$ denote the set of all classifying subspace clusters containing object o. Simply assigning the majority class label from this set $CSC(o)$ would be biased with respect to very large and redundant subspace clusters, where redundancy means similar clusters in slightly varying projections [5]. We therefore propose an iterative procedure that takes the *information gain* into account to build the decision set $DS_k(o)$.

Just as in the subspace clustering step we measure class homogeneity using the conditional class entropy. Starting with an empty decision set and apriori knowledge about class distribution $H(C)$ we select up to k subspace clusters with maximal information gain on the class label as long as more than ϕ_1 objects are contained in the decision space, i.e. the projection to the union of dimensions of the subspace clusters in the decision set.

Definition 6. ***Classification.*** *Given a dataset D, parameter k, an object $o = (o_1, \ldots, o_d)$ is classified to the majority class label of decision set DS_k. DS_k is iteratively constructed from $DS_0 = \emptyset$ by selecting the subspace cluster $SC_j \in CSC(o)$ which maximizes the information gain about the class label:*

$$DS_j = DS_{j-1} \cup SC_j, SC_j = \left\{ \underset{SC_i \in CSC(o)}{\mathbf{argmax}} \{H(C|DS_{j-1}) - H(C|DS_{j-1} \cup SC_i)\} \right\}$$

under the constraints that the decision space contains at least ϕ_1 objects:

$$|\{v \in D, v|_{DS_k} = o|_{DS_k}\}| \geq \phi_1$$

and that the information gain is positive

$$H(C|DS_{j-1}) - H(C|DS_{j-1} \cup SC_i) > 0$$

Hence, the decision set of an object o is created by choosing those k subspace clusters containing o that provide most information on the class label, as long as more than a

minimum number of objects are in the decision space. o is then classified according to the majority in the decision set DS_k. The decision set is then the set of locally relevant attributes that were used to classify object o. The attributes in the decision set are helpful for users wishing to understand the information that led to classification.

3 Algorithmic Concept

Our algorithmic concept focuses on step 1 that is the computationally most complex. A simple brute-force search would require evaluating all 2^N subspaces which is not acceptable for high dimensionality N. We thus propose lossless pruning of subspaces based on two entropy monotonicities.

Theorem 1. ***Upward Monotony of the Class Entropy.*** *Given a set of m attributes, subspace $S = \{X_1, .., X_m\}$, $e \in \mathbb{R}^+$ and $T \subseteq S$, the class entropy in subspace T is less than or at most equal to the class entropy of its superspace S:*

$$H(C|T) < e \quad \Rightarrow \quad H(C|S) < e$$

Proof. The theorem follows immediately from $H(X|X_i, X_j) \leq H(X|X_i)$ [12].

This theorem states that the class entropy decreases monotonically with growing number of attributes. Conversely, attribute entropy increases monotonically with the number of attributes.

Theorem 2. ***Downward Monotony of the Attribute Entropy.*** *Given a set of m attributes, subspace $S = \{X_1, .., X_m\}$, $e \in \mathbb{R}^+$ and $T \subseteq S$, the attribute entropy in subspace T is greater than or at most equal to the class entropy of its superspace S:*

$$H(S|C) < e \quad \Rightarrow \quad H(T|C) < e$$

Proof. The theorem follows immediately from $H(X_i, X_j|C) \geq H(X_i|C)$ [12].

We exploit monotonicity by pruning

- all those subspaces T whose superspaces $S \supset T$ fail the class entropy threshold. This is correct since the normalization factor $H(C)$ is independent of the subspace.
- Prune all those superspaces T whose subspaces $S \subset T$ fail the attribute entropy threshold if $log_2|I| \geq log_2|S|$. This is correct since the normalization factor is independent of the subspace if $\min\{log_2|I|, log_2|S|\} = log_2|I|$.

Our proposed algorithm alternately determines lower dimensional and higher dimensional *one-sided homogeneous* subspaces, i.e. subspaces that are homogeneous w.r.t. to class or attribute entropy, respectively. In each step new candidates are created from the set of one-sided homogeneous subspaces mined in the last step.

Figure 1 illustrates pruning in a subspace lattice of four attributes. The solid line is the boundary for pruning according to attribute entropy and the dashed line according to class entropy. Each subspace below the attribute boundary and above the class boundary is homogeneous with respect to the entropy considered. The subspaces between

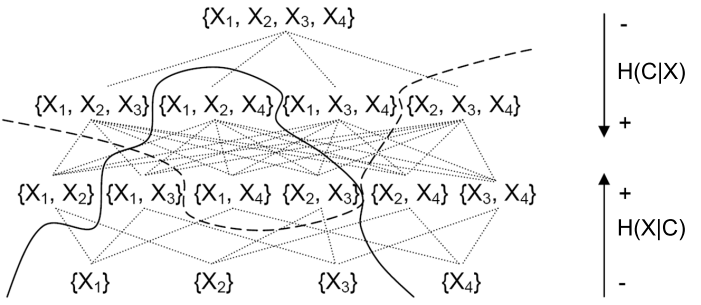


Fig. 1. Lattice of Subspaces and their projections used for up- and downward pruning

both boundaries are interesting subspace candidates, whose combined entropy has to be computed in the next step.

For the bottom up case, the apriori property, originally from association rule mining, can be used to create new candidates [2,8,15]. Following the apriori approach, we join two attribute homogeneous subspaces of size m with identical prefixes (e.g. in lexicographic ordering) to create a candidate subspace of size $m + 1$. After this, each new candidate is checked for entropy validity, i.e. if all of its possible subspace of cardinality m are contained in the set of attribute homogeneous candidate subspaces.

We suggest a similar method for top down candidate generation using class monotonicity. From the set of class homogeneous subspaces of dimensionality m, we generate all subspace candidates of dimensionality $m - 1$. We develop a method that ensures that each subspace candidate is only generated once. Based on the lexicographic order, our method uniquely generates a subspace of dimensionality $m-1$ from its smallest superspace. Note that this guarantees that all candidates but no superfluous candidates are generated (see example below). After this, just as with apriori, we check whether all superspaces containing the newly generated candidates are class homogeneous subspaces. Otherwise the new generated subspace is removed from the candidate set.

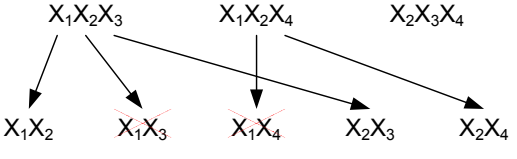


Fig. 2. Example top down generation

Example. Assume four attributes $X_1, \ldots, X_4$ from the previous step subspaces $X_1X_2X_3$, $X_1X_2X_4$, and $X_2X_3X_4$ that satisfy the class entropy criterion. In order to generate candidates, we iterate over these subspaces in lexicographic order. The first three-dimensional subspace $X_1X_2X_3$ generates the two-dimensional subspaces X_1X_2 (drop X_3), X_1X_3 (drop X_2), X_2X_3 (drop X_1). Next, $X_1X_2X_4$ generates X_1X_4 and

X_2X_4. X_1X_2 is not generated, because dropping X_4 is not possible, as it is preceded by X_3 which is not contained in this subspace. The last three-dimensional subspace $X_2X_3X_4$ does not generate any two-dimensional subspace since the leading X_1 is not contained; its subsets X_2X_3 and X_2X_4 have been generated by other three- dimensional subspaces. After candidate generation, we check their respective supersets. For example, for X_1X_2, its supersets $X_1X_2X_3$ and $X_1X_2X_4$ exist. For X_1X_3, its superset $X_1X_2X_3$ exists, but $X_1X_3X_4$ does not, so it is removed from further consideration following monotony pruning. Likewise, X_1X_4 is removed as $X_1X_3X_4$ is missing, but X_2X_3 and X_2X_4 are kept.

As we use two entropies, one with downward, one with upward pruning, subspaces may need to be considered twice. Minimizing computations is thus a trade-off. Figure 3 illustrates these effects. A missing candidate in S_{Down} (e.g. X_1X_2) means that this candidate has an attribute entropy above β. According to the attribute monotony, superspaces (e.g. $X_1X_2X_3$) have an attribute entropy above β and thus the combined entropy is also greater than β. Even though the subspace could be pruned according to combined entropy, it is still required for valid class entropy candidate generation. There is thus a trade off between avoiding computations and reducing the search space

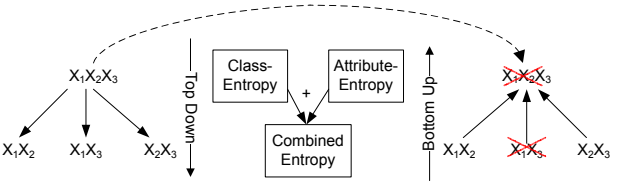


Fig. 3. Pruning of subspace $X_1X_2X_3$

by pruning high entropy subspaces. A good heuristic is to evaluate the entropy of those subspaces for which larger subspaces already had a high entropy. Randomly picking subspaces for additional evaluation also performs quite well in practice.

If the bottom up approach has not pruned the investigated subspace, the top down approach computes the entropy of the subspace. If the weighted normalized entropy is below β the subspaces is added to the result set and marked as one-sided homogeneous. The algorithm finally computes the combined entropy of all subspaces for which both subspaces are marked one-sided homogeneous in the result sets.

Once subspaces have been evaluated for **step 1**, the most complex algorithmic task has been solved. Having reduced the potentially exponential number of subspaces to the interesting ones, the actual clustering (**step 2**) is performed for each of these subspaces. This is done by computing the frequency and class entropy for all attribute value combinations in these subspaces. The resulting classifying subspace clusters then provide the model that is used for the actual classification (**step 3**). For incoming objects, compute the most similar classifying subspace clusters according to relative Hamming distance. If tied, compute reverse class entropy. The decision is then based on their class label distribution.

4 Experiments

Experiments were run on both synthetic and real world data. Synthetic data is used to show the correctness of our approach. Local patterns are hidden in a data set of 7.000 objects and eight attributes. As background noise, each attribute of the synthetic data set is uniformly distributed over ten values. On top of this, 16 different local patterns (subspace clusters) with different dimensionalities and different numbers of objects are hidden in the data set. Each local pattern contains two or three class labels among which one class label is dominating. We randomly picked 7.000 objects for training and 1.000 objects for testing.

The flight data contains historic data from a large European airport. For a three-month period, we trained the classifier on arrivals of two consecutive months and tested on the following month. Outliers with delays outside [-60, 120] minutes have been eliminated. In total, 11.072 flights have been used for training and 5.720 flights for testing. Each flight has a total of 13 attributes, including e.g. the airline, flight number, aircraft type, routing, and the scheduled arrival time within the day. The class labels are "ahead of schedule", "on time" and "delayed". Finally we use two well-known real world data sets from the UCI KDD archive (Glass and Iris [14]), as a general benchmark.

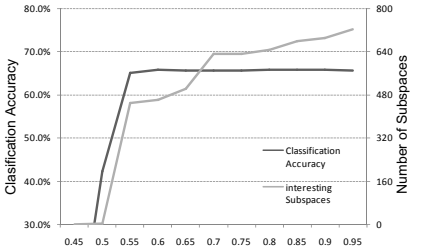
As mentioned before, preliminary experiments on the flight data indicate that no global relevance of attributes exist. Moreover, the data is inherently noisy, and important influences like weather conditions are not collected from scheduling. For realistic testing as in practical application, classifiers can only draw from existing attributes. Missing or not collected parameters are not available for training or testing neither in our experiments nor during the actual scheduling process.

We have conducted prior experiments to evaluate the effect of ϕ and γ for minimum frequency and maximum entropy thresholds, respectively. For each data set we used a cross validation to chose ϕ_1 (absolute frequency), ϕ_2 (relative frequency) and γ. For λ we have chosen 0.9. This value corresponds to a rather relaxed setting as we only want to remove completely inhomogeneous subspaces from consideration. To restrict the search space β can be set to a low value.

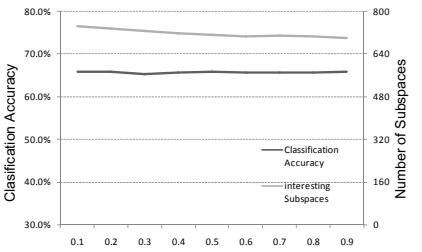
In our first experiments we develop a heuristic to set up reasonable parameters for the threshold β of the interestingness and the weight w of the class and attribute entropy, respectively.

Figure 4(a) illustrates varying β from 0.45 to 0.95 on the synthetic data, measuring classification accuracy and the number of classifying subspaces. The weight w for interestingness was set to 0.5. As expected, the number of classifying subspaces (CSS) decreases when lowering the threshold β. At the same time, the classification accuracy does not change substantially or even increases slightly when less subspaces are used. This effect may be related to the effect of overfitting. Using too many subspaces patterns are not sufficiently generalized, and noise is not removed. To set up the threshold β, slowly increasing β until the number of classifying subspace clusters shows a rapid rise, allows adjusting β to a point between generalization and overfitting. For both our data sets, a value around 0.65 obtains produces good results.

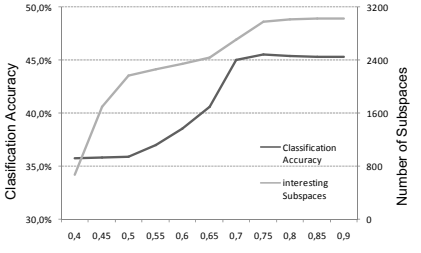
The effect of slightly increasing classification accuracy when reducing the number of subspaces can also be observed on the flight delay data (see Figure 4(c)). This confirms that the flight data contains local patterns for classification.



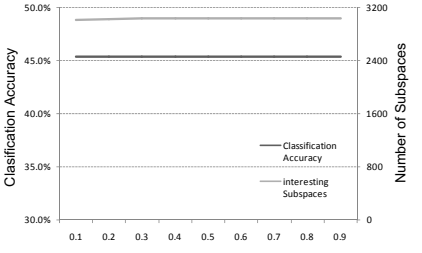
(a) Varying β on synthetic data



(b) Varying w on synthetic data



(c) Varying β on flight delay data



(d) Varying w on flight delay data

Fig. 4. Parameter evaluation using synthetic and real world data set

Varying parameter w yields the results depicted in the left part of Figure 4(b) and 4(d). The number of classifying subspaces decreases when giving more weight to attribute entropy. At the same time, classification accuracy does not change significantly. This robustness is due to the ensuing subspace clustering phase. As classification accuracy does not change this confirms that our classifying subspace cluster definition selects the relevant patterns. Setting $w = 0.5$ gives equivalent weight to the class and attribute entropy and hence is a good choice for pruning subspaces. We summarize our heuristics used to setup the parameters for our *SubClass* algorithm in Figure 5.

Next, we evaluate classification accuracy by comparing *SubClass* with other well-established classifiers that are applicable on nominal attributes: the k-NN classifier with Manhattan distance, the C4.5 decision tree that also uses a class and attribute entropy model [18], and a Naive Bayes classifier, a probabilistic classifier that assumes independence of attributes. Parameter settings use the best values from the preceding experiments.

Figure 6 illustrates the classification accuracy using four different data sets. In the noisy synthetic data set, our *SubClass* approach outperforms other classifiers. The large degree of noise and the varying class label distribution within the subspace clusters make this a challenging task. From the real world experiment on the flight data, depicted in Figure 6, we see that the situation is even more complex. Still, our *SubClass* method performs better than its competitors. This result supports our analysis that locally relevant information for classification exists that should be used for model building. Experts

Subspace Search Parameter

	Parameter	Value
β	Treshold for combined subspace interestingness	0.8-0.9
λ	Treshold for subspace interestingness	0.85-0.95
w	Weight for combined subspace interestingness	≥ 0.5

Subspace Clustering Parameter

	Parameter	Value
γ	Threshold for class information	0.6-0.7
ϕ_1	Treshold for absolute frequency	$0.01\text{-}0.005 \cdot \lvert I \rvert$
ϕ_2	Treshold for expected frequency	$3\text{-}5 \cdot \lvert C \rvert$

Fig. 5. Parameters used by *SubClass*

	Flight Data	Synthetic Data	Iris	Glass
SubClass	**45.4%**	**65.9%**	**96.27%**	70.9%
C4.5	43.9%	58.0%	95.94%	66.8%
K-NN	42.4%	54.3%	93.91%	**71.1%**
Naive Bayes	42.8%	64.1%	95.27%	46.7%

Fig. 6. Classification accuracy on four data sets

from flight scheduling confirm that additional information on further parameters, e.g. weather conditions, is likely to boost classification. This information is inexistent in the current scheduling data that is collected routinely. *SubClass* exploits all the information available, especially locally relevant attribute and value combinations, for the best classification in this noisy scenario. Finally we evaluated the performance of *SubClass* on Glass and Iris [14]. The results indicate that even in settings containing no or little noise *SubClass* performs well.

5 Conclusion

Classification in noisy data with locally varying attribute relevance, as for our project in scheduling at airports, requires an approach that detects local patterns. Our *SubClass* method automatically detects classifying subspace clusters by incorporating class structure into the subspace search and the subspace clustering process. The general concept requires a definition of interesting subspaces for classification, of classifying subspace clusters and a classification scheme. Based on class and attribute value entropy, our *SubClass* ensures that clusters contain class-relevant information. Working both bottom-up and top-down on the lattice of subspaces, *SubClass* prunes irrelevant subspaces from the mining process. Our experiments on synthetic and real world data demonstrate that local structures are successfully detected and employed for classification, even in extremely noisy data.

References

1. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications. In: Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 94–105 (1998)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of International Conference on Very Large Databases (VLDB), pp. 487–499 (1994)
3. Aha, D., Kibler, D., Albert, M.: Instance-based learning algorithms. Machine Learning 6(1), 37–66 (1991)
4. Assent, I., Krieger, R., Glavic, B., Seidl, T.: Spatial multidimensional sequence clustering. In: Proceedings of International Workshop on Spatial and Spatio-Temporal Data Mining (SSTDM), conjunction with IEEE International Conference on Data Mining (ICDM) (2006)
5. Assent, I., Krieger, R., Müller, E., Seidl, T.: DUSC: Dimensionality unbiased subspace clustering. In: Proceedings of IEEE International Conference on Data Mining (ICDM) (2007)
6. Bolat, A.: Procedures for providing robust gate assignments for arriving aircrafts. European Journal of Operational Research 120, 63–80 (2000)
7. Bureau of Transportation Statistics. Airline on-time performance data, `http://www.transtats.bts.gov`
8. Cheng, C., Fu, A., Zhang, Y.: Entropy-based subspace clustering for mining numerical data. In: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 84–93 (1999)
9. Domeniconi, C., Peng, J., Gunopulos, D.: Locally adaptive metric nearest-neighbor classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(9), 1281–1285 (2002)
10. Duda, R., Hart, P., Stork, D.: Pattern Classification, 2nd edn. Wiley, Chichester (2000)
11. Eurocontrol Central Office for Delay Analysis. Delays to air transport in europe, `http://www.eurocontrol.int/eCoda`
12. Gray, R.: Entropy and Information Theory. Springer, Heidelberg (1990)
13. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco (2001)
14. Hettich, S., Bay, S.: The uci kdd archive. University of California, Department of Information and Computer Science, Irvine, CA (1999), `http://kdd.ics.uci.edu`
15. Kailing, K., Kriegel, H.-P., Kröger, P.: Density-connected subspace clustering for high-dimensional data. In: Proceedings of IEEE International Conference on Data Mining (ICDM), pp. 246–257 (2004)
16. Li, W., Han, J., Pei, J.: CMAR: accurate and efficient classification based on multipleclass-association rules. In: Proceedings of IEEE International Conference on Data Mining (ICDM), pp. 369–376 (2001)
17. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Schoelkopf, Burges, Smola (eds.) Advances in Kernel Methods, MIT Press, Cambridge (1998)
18. Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1992)
19. Shannon, C., Weaver, W.: The Mathematical Theory of Communication. University of Illinois Press, Urbana, Illinois (1949)
20. Silva, L., de Sa, J.M., Alexandre, L.: Neural network classification using shannon's entropy. In: Proceedings of European Symposium on Artificial Neural Networks (ESANN) (2005)
21. Washio, T., Nakanishi, K., Motoda, H.: Deriving Class Association Rules Based on Levelwise Subspace Clustering. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 692–700. Springer, Heidelberg (2005)
22. Zhu, X.: Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison (2005)