



Selection of casting production parameters with the use of machine learning and data supplementation methods in order to obtain products with the assumed parameters

Dorota Wilk-Kołodziejczyk^{1,2} · Zenon Pirowski² · Adam Bitka^{1,2} · Kamil Wróbel¹ · Bartłomiej Śnieżyński¹ · Maciej Doroszewski¹ · Krzysztof Jaśkowiec^{1,2} · Marcin Małysha^{1,2}

Received: 31 March 2022 / Revised: 13 December 2022 / Accepted: 30 December 2022 / Published online: 2 February 2023
© The Author(s) 2023

Abstract

The main purpose of the research, presented in this publication, was to develop methodology for the construction of predictive models which allow the selection of material production parameters for the material-technological conversion process. The development of prototype modules based on information-decision system allows an initial assessment of the level of feasibility of undertaking this type of operation. Algorithms 1, 2, 3 presented in the article were used to complete the missing data. The result of the algorithm enabled the creation of a data table that specifies the operation of the predictive models indicated in chapter 3 of this article. Entire work is presented with regard to the background of the ADI cast iron production process to locate the requirement where to apply the developed methods in the field of predictive algorithms and data completion algorithms. On the basis of developed methods and predictive algorithms, trial castings were operated.

1 Introduction

Predicting the mechanical properties of foundry products using machine learning methods has existed as a long field of machine learning itself. Already in one of the first scientific papers on this topic [1] models in which more than one property is predicted experience problems with missing data for one of them. This problem, together with an example of a solution, is presented in the paper [2]. To fill in the missing values, one should find a subset of samples in which, for the dimension with missing data, there is a highly correlated other dimension, on the basis of which it is possible to determine the missing value, e.g. by linear regression. The paper concludes that the advantage of this approach is the possibility of obtaining new limit values for the data set. In case of no correlation between the attributes, the missing value can be computed by comparing selected records from the set containing the full data. One of the first works, which addressed the problem of predicting the

mechanical properties of ADI, was work [3]. This paper describes various models based on fuzzy-based logic, which were implemented to predict the ADI impact strength from the temperature and duration of the ausferritization process, with the assumption that other parameters were constant. The dataset, which the model was trained on, contained only 28 samples, of which 21 were selected as the training set and the remaining 7 samples, as the test set. The best result was obtained using the approach based on clustering, in which the coefficient of correlation between predicted and actual values was 0.9. The next paper [4] shows the prediction of ADI hardness using neural networks. It is the first of the works found in the literature addressing this problem. The use of this approach, model, based on Bayesian neural networks, was developed. These networks were built using one hidden layer, for which different numbers of neurons from 2 to 20 were tested and different regularization constants (one related to each input, one with biases and one for weights connected to the output). Various numbers of the best network configurations were also tested. The network learning process was carried out using a set of 1822 samples collected from the literature. The set was randomly divided into training and testing in a 50/50 proportion. The input data for all networks was the chemical composition of the heats, their heat treatment parameters and the presentation of the ausferritization time using the logarithm of the time value

✉ Dorota Wilk-Kołodziejczyk
dorota.wilk@kit.lukasiewicz.gov.pl; dwilk@agh.edu.pl

¹ AGH University of Science and Technology, Kraków, Poland

² Łukasiewicz Research Network–Kraków Institute of Technology, Kraków, Poland

in seconds. Such an approach was developed in the work on the determination of the content of retained austenite in ADI [5]. The network input dimensions have been normalized to the range $[-0.5; 0.5]$. Research has shown that the more neurons in the hidden layer, the lower noise level. Different regularization constants (the results of different configurations are shown with pluses) also had a minor impact. Based on the aforementioned graph, it can be determined that the error value of the best models oscillates between 1.5 and 2. The network input dimensions have been normalized to the range $[-0.5; 0.5]$. Research has shown that the more neurons in the hidden layer, the lower the noise level. Different regularization constants (the results of different configurations are shown with pluses) also had a minor impact. Based on the aforementioned graph, it can be determined that the error value of the best models oscillates between 1.5 and 2.

The construction of hardness predictive models has also been presented in [6–8], where the papers [7] and [8], by Savagounder, Patra and Bornand, were published in parallel, based on the data that was published presented in [6]. The aforementioned dataset contains the results obtained from 96 samples varied in terms of chemical composition and heat treatment parameters. It was prepared on the basis of ADI hardness tests following the procedure thoroughly described in this article. In the work of Hamid and Hamed PourAsiabi et al. [6], the predictive model was developed on the basis of a neural network, in which the input layer consists of the percentage of copper and molybdenum as well as the ausferritization temperature and time. The hidden layer consists of 5 neurons in which the activation function is the tangensoidal function. The data set was divided into a training set and a test set in the proportion of 70/26. The mean square error method was used to minimize the lattice error, and the Levenberg–Marquardt algorithm, based on the second-order derivative of the error function, was used to obtain optimal weights. The input dimensions were normalized to values in the range $[-1, 1]$. The results of the trained neural network tests were presented using the correlation coefficient, the value of which was 0.9912. Such a high result proves a well-trained network and the ability to correctly predict hardness.

2 Materials and methods

2.1 Experimental procedure

The article uses heuristic algorithms to optimize ADI production parameters. These are existing algorithms, but so far have not been used in the aspect presented in the article. These are the Hill Climbing Algorithm presented in [9], the Tabu Search Algorithm described in [10] and Metropolis Search presented in [3]. In the literature, you can find examples of the use of heuristic algorithms in the optimization

of the production process. In the work [11], Pareto optimization was applied using genetic algorithms based on a data set. The subject of optimization covered the mechanical properties of micro-alloyed steel, such as tensile strength, elongation and yield point. For optimization purposes, models of mechanical properties based on evolutionary neural networks (EvoNN) were created. The work presents multi-criteria optimization based on the optimization of sets of criteria relating to the maximization of tensile strength and yield point, maximization of yield point and elongation, maximization of yield point and minimization of the ratio of yield point to tensile strength. The above three optimization tasks were carried out with the use of a multi-criteria genetic algorithm. Pareto solutions for each of the optimization tasks differed significantly from the solutions based on the data used to build the models. To test this very important observation, an alloy was prepared in close proximity to the composition obtained by this method, which, to the knowledge of the researchers, has not been tested by anyone so far. The samples from the prepared alloy were tested and their properties (within the limits of experimental errors) were quite consistent with the Pareto front. Researchers estimated that the creation of steel with better properties than any of the elements in the dataset would be impossible to achieve by performing the experiments alone, while this combination was achieved by multi-criteria optimization. Another work on the optimization of the production process is the work of the author, R. Radis et al. [12], in which researchers have taken up the problem of optimizing the geometry of the feeder used to pour a Pelton turbine bucket. The created objective function was developed based on the heat transfer requirements. In addition, 4 different requirements have been developed for the solutions returned by the optimization algorithms. 4 different values (H, R, r, l) were optimized using 4 different meta-heuristic algorithms: genetic, ant, simulated annealing and particle swarm optimization. The results obtained for each of the algorithms turned out to be very similar to each other. On their basis, the feeder was modelled for which numerical simulations were carried out to confirm the correctness of the solution. The required quality for a given order is specified in the EN-PN 1564: 2012 standard [13]. The diagrams presented in Fig. 1 show how the production process in the foundry looks like. The execution of production orders starts with a client sending in request for quotation by the client, which includes the following elements:

- technical drawing, and on it:
- workpiece weight,
- roughness,
- alloy grade,
- applied standards, deviations.
- number of pieces/elements,

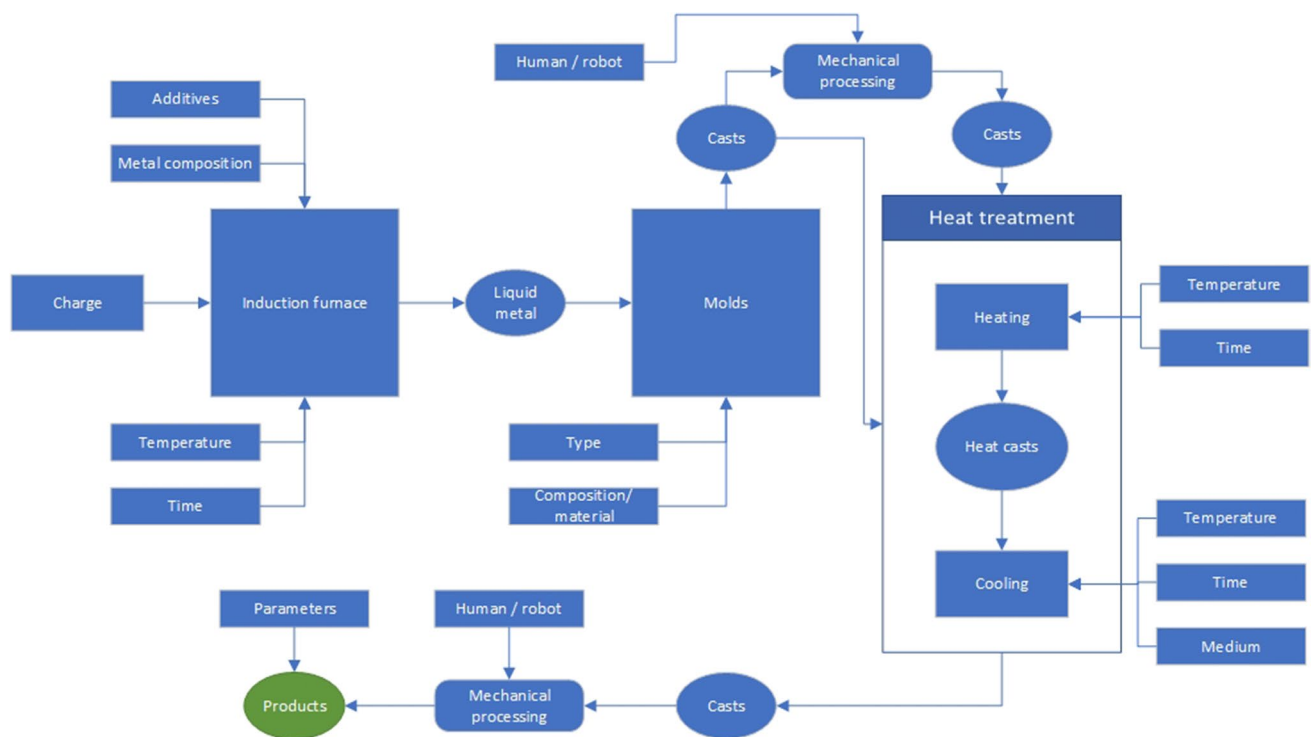


Fig. 1 Diagram of the ADI foundry production line

- expected completion time,
- conditions of receipt,
- the scope of machining,

The focus was on:

- weight of the detail,
- alloy grade,
- applicable standards,
- number of pieces of a detail.

The above elements were selected being directly related to production costs and constraints that should be addressed during decision process. Before starting production, an offer is prepared based on an inquiry, which stems from:

- analysis of the processing capacity,
- analysis of the constructor and / or technologist or resources at the disposal of the foundry, enabling the execution of the casting in accordance with the guidelines and the provided drawing,
- cost estimation.

As part of the planned solution, the following stages of the production process will be supported:

- planning process → collecting information about the detail to be made,
- charge preparation process → determination of the wt.% of elements in the cast alloy,
- austenitization, austempering → process temperatures and times.

2.2 Model

To reduce the number of trials related to the launch of the production of new castings, there is a need to create a predictive model of the planned mechanical properties on the basis of the selected extraction method and chemical composition. The task of this model will be to predict the mechanical properties of the planned casting based on:

- chemical composition of the casting, (elements defined as percentages have been selected)
- casting thickness (value in millimeters)
- heat treatment parameters:
 - austenitization temperature (degrees Celsius),
 - austenitization time (minutes),
 - austempering temperature (degrees Celsius),
 - austempering time (minutes)

The values predicted by the model will be: ´

- UTS - tensile strength [MPa], ´
- YS - yield strength [MPa], ´
- A - elongation [%], ´
- HB - Brinell hardness [HB], ´
- K - impact strength [J].

The model was designed on the basis of ADI data.

The data needed for the model comes partly from experiments done as part of the work carried out at the former Foundry Research Institute in Kraków (currently the Łukasiewicz Research Network -Kraków Institute of Technology) and at the Bydgoszcz University of Science and Technology. Data was also collected from articles published in recognized scientific and trade journals. It is problematic that in the literature sources, the test results are mostly presented as tables showing how the given melt compositions and how their heat treatment influenced the mechanical properties. Some of the articles, which contain a very extensive cross-section of various alloys and machining parameters, present only data in graphs, which makes it unhelpful to aggregate data in one place. In some cases, not complete data has been provided.

It has been agreed that the following information will be collected for the data set:

- chemical composition of the melt (percentage of elements):
- carbon (C),
- silicon (Si),
- manganese (Mn),

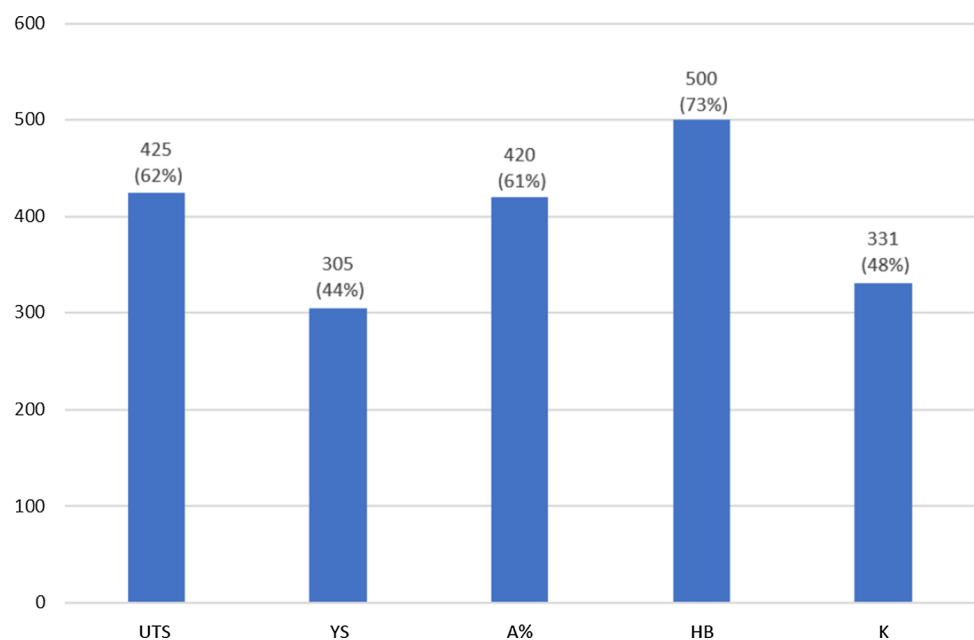
- magnesium (Mg),
- copper (Cu),
- nickel (Ni),
- molybdenum (Mo),
- sulfur (S),
- phosphorus (P),
- vanadium (V),
- chromium (Cr).
- carbon equivalent (CE) - calculated as [13]: $CE = \% C + 0.33 (\% Si + \% P)$,
- heat treatment parameters:
- austenitization temperature and time (aust_temp, aust_time),
- ausferitization temperature and time (ausf_temp, ausf_time).
- mechanical properties:
- UTS,
- YS,
- A,
- HB,
- K
- casting thickness (mm),
- mechanical properties of the alloy in as-cast condition (before heat treatment).

Statistics were developed describing the number of articles analyzed, records collected, and records rejected on the basis of a negative assessment of specialists.

Statistics:

- number of collected articles / research papers: 66,

Fig. 2 Statistics of mechanical properties



- number of records collected (without division into mechanical properties): 941 (including 252 records rejected),
- number of complete records (including all mechanical properties, without rejected ones): 137,
- number of records collected (without rejections and with division into mechanical properties): 1981,
- number of unique alloys: 94,
- number of missing values: 1464 (42% of all possible values for mechanical properties in the data set).

The statistics of the records collected, broken down by mechanical properties, are shown in Fig. 2. It shows that for two mechanical properties (yield strength and impact toughness), the number of collected data in relation to all records is less than 50%. These data were the rarest in the found works. The most common data turned out to be hardness.

The method for determining the missing values is presented in [2]. A significant problem related to the data published in the articles was also the fact that the hardness measurement method was presented in various scales, i.e. the Brinell hardness scale, the Rockwell hardness scale and the Vickers hardness scale. Due to the majority of data presented in the form of the Brinell scale, data in other scales were converted to this scale. Impact strength turned out to be an additional problem, as it is also represented by two different markings:

- K–impact strength measured on unnotched samples,
- KV–impact strength measured on samples with a V–notch.

Due to a lack of additional information on how these two values are related (no data describing the impact toughness for both notched and unnotched samples at the same time), it was assumed on the basis of the expert knowledge that $K = 11 \text{ kV}$ based on the tables in PN-EN 1564: 2012 standard. Due to the greater share of notch toughness records for unnotched specimens, the toughness records measured on notched specimens were converted to unnotched specimen values using the relationship indicated. The data found in the articles were also analyzed. Charts have been prepared for each of the dimensions for better analysis. In the case of the results for the chemical composition, there are outliers in every dimension except for copper and nickel. In most cases, these are small deviations from the average value in a given dimension, so it can be concluded that these are the ranges of values that are not well covered. However, there

are dimensions where these deviations are very large, which may indicate errors in the articles from which the data was used from. An additional disadvantage of removing outliers in the case of chemical composition would be throwing away a significant number of records from the set, as there are on average more than 9 records for each melt. In the case of diagrams for dimensions describing thermal treatment, the austenitizing temperature is the dimension that is striking. The value of this dimension fluctuates around 900 degrees Celsius, as it is the standard temperature for the austenitization process. Values shown as outliers have been deliberately left in the dataset to investigate how other values of this temperature will affect the mechanical properties. A similar situation occurs in the dimension describing the ausferritization temperature, there are 3 values that differ significantly from the rest of the data, but they were also preserved in the data set. The outliers can be seen in the dimensions of K (impact toughness) and in the dimension describing the thickness of the casting. Outliers for the impact dimension were taken from one study which accurately presented the data and there is no indication that they could be incorrect. For the thickness dimension, the outliers result from the fact that 25 mm thick cast ingots were tested in most of the works.

2.3 Analysis of correlations occurring in a data set

To better analyze the data, using the pandas library a correlation matrix was prepared. It was generated between the dimensions describing the chemical composition and heat treatment parameters and the mechanical properties. The values of the correlation matrix were mapped to absolute values, and such a matrix with absolute values was used as an argument of the heatmap function from the seaborn library, which returns a heatmap based on the value in the matrix passed. The correlation matrix (Fig. 3) has not been presented for all dimensions due to their large number and the willingness to investigate the correlation between the dimensions describing the relationships between the production process and mechanical properties and mechanical properties. The values visible in the matrix are the absolute values to see the dimensions most correlated with each other. The matrix will be used to analyze the correlation in the data set.

The conducted analysis indicates the possibility of supplementing the missing data using other dimensions for which values exist. The correlation matrix from Fig. 2 shows:

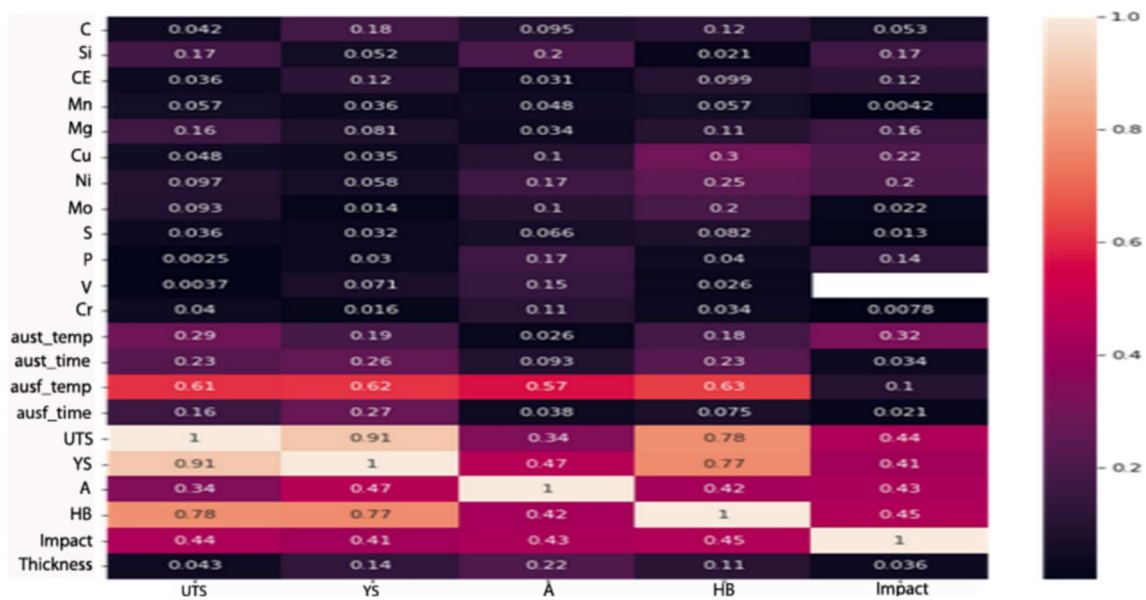


Fig. 3 Matrix of correlation of all dimensions to dimensions describing mechanical properties

- Strong correlation between the dimensions of UTS and YS (0.91),
- The dimensions of UTS and YS are also highly correlated with the HB dimension (0.78 and 0.77),
- The dimension describing the ausferritization temperature is to some extent correlated with the dimensions UTS, YS, A, HB (approx. 0.6),
- The K dimension describing the impact toughness looks less significantly correlated with the other dimensions describing the physical properties.
- the size of a subset of samples should be specified, e.g. 10 samples,
- there may be more than one correlated dimension, the number of such dimensions that will be taken into account should be specified, e.g. 2 dimensions,
- due to the large dispersion of data in the set, the subset of samples should be selected in such a way that the data in the uncorrelated dimensions and dimensions not related to the mechanical properties of cast iron are their closest neighbors,
- the use of linear regression will be adequate for subsets with a very high correlation coefficient, i.e. above 0.8, for subsets with a smaller coefficient, another method of determining the value should be used (e.g. decision trees),
- the higher the correlation coefficient in a given subset and the samples from the subset are as close as possible to the sample for which the missing value will be completed, the more likely it will be that the determined value will be correct.
- Additional assumptions about the refilling method:
- due to the different number of missing data and different degrees of correlation between the dimensions with missing data, the user should specify the order in which the dimensions will be completed (e.g. in the first 3 places there should be the most numerous dimensions that are strongly correlated with each other - UTS, HB, YS),

The results of the application of correlation have been confirmed by the expert knowledge in this area. The conclusions drawn during the correlation analysis clearly show that the dimensions UTS, YS, A, HB, ausf_temp are significantly correlated with each other and it can be assumed that it is possible to supplement missing data in these dimensions using other existing data. Due to the low degree of correlation of the K dimension with other dimensions, it will be omitted for the period of supplementing the above-mentioned dimensions.

2.4 Completing the missing data

The problem of "missing data" has been discussed in [2]. Based on this diagram, assumptions were made for the own method of completing the missing data:

- it is assumed that supplementing the missing values may be performed with the use of samples for which the values were completed in the earlier stages of supplementation,
- it will be advantageous first to complete the samples for which the correlation coefficient in the found subsets is higher than in the remaining ones—for this purpose it is assumed that at a given stage of data completion only those samples for which the correlation coefficient for the subset will be higher or equal to the threshold will apply at a given stage of supplementation,
- due to the possibility of using supplemented data when supplementing others, it is assumed that the completion process for a specific correlation coefficient threshold will be carried out as long as there are samples with subsets that meet a given threshold (this approach will allow the use of samples supplemented in the previous steps that will meet a given threshold),
- due to the expected different correlation coefficients in the subsets, it is assumed that samples with high correlation coefficients among the subsets will be completed first, and then those with smaller ones,
- due to the fact that for some data the subsets found may have very small correlation values, it is assumed that the maximum number of steps of data completion is set, after which the completion process will be interrupted.
- On the basis of the above, a method of supplementing the missing values in the data set describing the mechanical properties of ADI was prepared. In order to complete the missing data, the user should determine:
 - order of completed dimensions,
 - the initial threshold of the correlation coefficient (mentioned in the assumptions),
 - number of samples in the subset,
 - the boundary of the correlation coefficient to which the linear regression will be applied,
 - the number of dimensions from which the missing value will be predicted, a machine learning algorithm used outside of linear regression.

The method of completion consists in finding a subset of samples that are the closest neighbors of the sample to be replenished for dimensions identified as uncorrelated and not related to the physical properties of cast iron. Determination

of the missing value consists in predicting this value using a model built from the data in the subset found. First, the samples for which the average correlation coefficient in a defined subset will exceed a given threshold are completed. The threshold, together with the lack of change in the number of missing values, is decreased by a fixed value, e.g. 0.05. Samples up to a fixed value of the correlation coefficient in a subset (e.g. 0.8) are completed with linear regression, then another machine learning algorithm is used, e.g. a decision tree. For a given threshold of the correlation coefficient, a number of steps are taken to supplement the missing values by using the data that has been completed in the previous steps. Lowering the threshold is when the number of missing data does not decrease. The finalization of completion occurs when all missing values have been supplemented or the number of steps exceeds the predetermined maximum number of steps.

The mathematical model and flow chart of the algorithm is presented in three parts, the first of which (Algorithm 1) presents the main steps of the algorithm. Input data for Algorithm 1:

- X - data set,
- order - order of completed dimensions,
- corr_cutoff_bound - initial correlation threshold,
- req_neighs - number of samples in the subset,
- linear_reg_to - the boundary of the correlation coefficient to which the linear regression will be applied,
- amount_of_feats_to_predict - number of dimensions based on which the missing value is predicted,
- max_steps - the maximum number of steps,
- next_predictor_provider - an object used to return the machine learning algorithm used after crossing the limit of using linear regression.

Result: a set of data supplemented in the dimensions specified by a given order.

Auxiliary functions and complex lines of code used in Algorithm 1:

- line 2 - get_missing_values_count function that takes a set of samples and returns the total number of missing values in the UTS, YS, HB, A, K,
- line 10 is responsible for calling the fill_missing procedure described in algorithm 2

Algorithm 1: Completion of the data set

Input: order, corr_cutoff_bound, req_neighs, linear_reg_to, amount_of_feats_to_predict, max_steps, next_predictor_provider

Output: completed set of samples

Data: collection of samples X

```

1  corr_cutoff ← corr_cutoff_bound
2  missing_values_count ← get_missing_values_count(X)
3  repeat
4      filled ← 0
5      for feature in order do
6          if all values for feature are filled then
7              filled ← filled + 1
8              continue
9          for row in X do
10             // fill_missing procedure is described in
11             algorithm 2
12             fill_missing(row, feature, amount_of_feats_to_predict, req_neighs, corr_cutoff,
13             inear_reg_to, next_predictor_provider)
14             step ← step + 1
15             actually_missing ← get_missing_values_count(X)
16             if missing_values_count == actually_missing then
17                 corr_cutoff ← corr_cutoff - 0.05
18                 missing_values_count ← actually_missing
19 until filled == size(order) or step == max_steps
20 return X

```

Description of the input data of the fill_missing procedure (algorithm 2):

- row—row from the set of X samples, in which the missing value will be completed
 - feature—name of the currently completed dimension (e.g. UTS),
 - amount_of_feats_to_predict,
 - req_neighs, • corr_cutoff - the current threshold of the value of the correlation coefficient,
 - linear_reg_to,
 - next_predictor_provider,
- Description of the helper functions and more complex lines of code used in Algorithm 2:
- line 4—get_features_with_null function accepting the currently considered row and returning a set of all dimensions from this row that lack values,
 - line 7—the most_corr_feats function taking values of correlation coefficients for the dimension under consideration (feature), constant amounts_of_feats_to_predict and a set of dimensions with missing values and returning the n most correlated dimensions to the dimension under consideration (feature), where n is equal to amounts_of_feats_to_predict,
 - line 11—the remove_features_with_nulls function that takes a sample set of X and a set of not_correlated dimensions and returns a set of not_correlated dimensions with dimensions with missing values removed,
 - line 13 and 14—the min_max_scaler function returning the scaler object and when calling the scale function on this object, it returns a set of samples X 'scaled to the value [-1, 1],
 - line 15—overwriting the feature component of the row from the set of row samples with the value returned by the predict_value function, which was presented in algorithm 3.

Algorithm 2: Procedure of supplementing the value missing in a given sample

```

1  Procedure fill_missing(row, feature, amount_of_feats_to_predict, req_neighs,
   corr_cutoff, inear_reg_to, next_predictor_provider) is
   Data: collection of samples X
2  if row[feature] is not null then
3      continue
4  null_features ← get_features_with_nulls(row)
5  null_features.remove(feature)
6  corr_matrix ← get_correlation_matrix(X)
7  correlated ← most_corr_feats(corr_matrix[feature], amount_of_feats_to_predict,
   null_features)
8  correlated.remove(feature)
9  not_correlated ← X.features - correlated
10 not_correlated.remove(feature)
11 not_correlated ← remove_features_with_nulls(X, not_correlated)
12 X' ← X[not_correlated]
13 scaler ← min_max_scaler(X')
14 scaler ← scaler.scale(X)
   // predict_value() function is defined in algorithm 3
15 row[feature] ← predict_value(X', not_correlated, scaler, amount_of_feats_to_predict,
   req_neighs, corr_cutoff, inear_reg_to, next_predictor_provider)

```

Description of the input data of the function 'predict_value (algorithm 3):

- *X'* - a set of scaled samples with uncorrelated dimensions,
 - *not_correlated*—a set of uncorrelated dimensions,
 - *correlated*—a set of correlated dimensions,
 - *scaler*—an object that enables scaling values to the range $[-1, 1]$,
 - *req_neighs*
 - *amount_of_feats_to_predict*
 - *corr_cutoff* - the current threshold of the value of the correlation coefficient,
 - *next_predictor_provider*,
 - *linear_reg_to*,
 - Description of the helper functions and more complex lines of code used in Algorithm 3:
- line 4—*find_nearest_neighs* function matching the set *X'*, the scaled vector of values from the textitrow line for the *not_correlated* dimension set and the number of neighbors to find,
 - line 5—the *remove_neighs_with_nulls* function that takes a set of nns closest neighbors and a set of correlated dimensions and returns only those neighbors for which there are values in all correlated dimensions,
 - line 9—*features_above_corr_cutoff* function taking the vector of correlation coefficients for the feature dimension from the correlation matrix *corr_matrix*, constant *amount_of_feats_to_predict* and constant *corr_cutoff* and returning the *n* most correlated dimensions that meet the condition of a correlation coefficient greater than or equal to *corr_cut_feats_ amount_of redict_ (n)*.

Algorithm 3: Value prediction function to complete the sample

```

1  Function predict_value(X, not_correlated, correlated, scaler, req_neighs,
   amount_of_feats_to_predict, corr_cutoff, linear_reg_to, next_predictor_provider) is
   Result: Predicted value
2  neighs ← req_neighs
3  repeat
4     nns ←
       find_nearest_neighs(X, scaler.scale(row[not_correlated]), neighs)
5     nss ← remove_neighs_with_nulls(nss, correlated)
6     neighs ← neighs + (req_neighs − size(nns))
7  until size(nns) ≥ req_neighs
8  corr_matrix ← get_correlation_matrix(nns)
9  correlated ← features_above_corr_cutoff(corr_matrix[feature], amount_of_feats_to_predict,
   corr_cutoff)

10 if size(correlated) == 0 then
11   return null
12 if corr_cutoff ≥ linear_reg_to then
13   predictor ← LinearRegression()
14 else
15   predictor ← next_predictor_provider.provide()
16   predictor.train(nns[correlated], nns[feature])
17   predicted_value ← predictor.predict(row[correlated])
18   if predicted_value > 0 then
19     return predicted_value
20   else
21     return null

```

The implementation of the described algorithm has been supplemented with the collection of information on the method of completing the missing data:

- dimensions used to build a subset of the so-called sample neighbors,
- dimensions selected as correlated in the subset on the basis of which the model for prediction is built,
- correlation coefficients of the above-mentioned dimensions to the dimension in which the missing value was present,
- minimum and maximum value of the dimension with the missing data in the subset of neighbors,
- indexes of samples from a subset of neighbors that have been selected to build the model,

the number of the step where the missing value was predicted.

3 Results and discussion

3.1 Study of the data completion algorithm

The tests were carried out on parameters with the following values:

- amount_of_feats_to_predict - [1, 2],
 - corr_cutoff_bound - [0.85, 0.9, 0.95],
 - linear_reg_to - [0.7, 0.85],
 - next_predictor_provider - ['lin', 'dec_tree'] (where 'lin' means linear regression,
- 'Dec_tree' means a decision tree),
- order - [['HB', 'UTS', 'YS', 'A', 'K'], ['YS', 'HB', 'UTS', 'K', 'A']],
 - req_neighs - [5, 10, 20].

In the case of the `max_steps` parameter, it was decided that it would not be tested with different values. For each of the runs, it was set to the value of 15. This was to limit the time needed to run all parameter configurations. From the combination of all possible parameter values, 144 unique sets were obtained.

3.2 Method of testing the quality of supplemented data

To check how a given configuration of parameters will behave when completing the data set, validation sets were randomly selected for each of the dimensions of mechanical properties with the size of 5% of the available values in a given dimension. The extraction was carried out carefully to leave each sample with at least one value in the dimensions of the mechanical properties (UTS, YS, A, HB, K). To clarify the method of selecting data, a table has been prepared with sample data for dimensions of mechanical properties. The values are selected in the order of the columns.

Summary of validation set sizes:

- UTS–21 values,
- YS–15 values,
- A–21 values,
- HB–18 values,
- K–11 values.

After extracting these values, the dataset passed to the algorithm contains 1550 missing values. The quality of the obtained restorations was measured using the following metrics:

- R^2 - coefficient of determination,
- MAPE - mean percentage absolute error,
- number of missing values in the entire data set,
- percentage of values missing from validation sets.

The choice of the R^2 and MAPE metrics was dictated by the willingness to present the results as an average value for each of the supplemented dimensions of mechanical properties.

3.3 Scheme for presenting the results

The results were presented in the form of a ranking of the best configurations in terms of the metric under consideration. Each of the rankings contains 15 different configurations of the parameters of the completion algorithm. To limit the width of the column for the 'order' parameter, value ['HB', 'UTS', 'YS', 'A', 'K'] has been replaced with a value '1', and value ['YS', 'HB', 'UTS', 'K', 'A'] has been replaced with

a value '2'. Additionally, due to long parameter names, mapping has been introduced.

3.4 Building a training and test set

There is a need to build a model that will represent the solution space of the described problem. However, it was not possible to build a common model for all mechanical properties due to undertaking this part of the implementation before completing the data completion process, which had not gone according to plan. Independent models were built for each of the mechanical properties using different datasets that shared common parts due to the presence of records with all properties. Machine learning algorithms are successfully used to build such models. In the case under consideration, regression models that model relationships between two or more variables were used.

Due to the small number of samples (689) in relation to the number of dimensions (14), it is not possible to draw a random set of samples for the test set without losing important information from the training set. To prevent this, five-fold cross-validation was used with the same proportion of samples from each class. Due to the fact that the output dimensions are numbers, 5 classes (1, 2, 3, 4, 5) have been created, which divide the values into 5 sets of similar size, by determining the range of values included in a given set. The ranges of values for each class for each of the mechanical properties were as follows:

UST: 1 – [UST_{min} , 960), 2 – [960, 1045), 3 – [1045, 1127), 4 – [1127, 1264), 5 – [1264, UST_{max}], YS: 1 – [$Rp_{0.2min}$, 696), 2 – [696, 813), 3 – [813, 900), 4 – [900, 1100), 5 – [1100, YS_{max}], HB: 1 – [HB_{min} , 286), 2 – [286, 325), 4 – [370, 415), 5 – [415, HB_{max}], A: 1 – [A_{min} , 2.7), 2 – [2.7, 4.5), 3 – [4.5, 6.5), 4 – [6.5, 8.4), 5 – [8.4, A], K: 1 – [K_{min} , 52), 2 – [52, 75), 3 – [75, 90), 4 – [90, 110), 5 – [110, K_{max}].

The model training process was carried out as follows:

- training of each model was carried out using cross-validation,
- for each of the algorithms, different values of parameters controlling their operation have been selected.
- parameter tuning was performed by training models for each permutation of algorithm parameters,
- metrics used to evaluate the quality of the trained models:
 - MAE (mean absolute error),
 - R^2 (determination coefficient).
- the model, after the evaluation of the metrics, is trained on the test set,
- one best model for each of the metrics was used for comparative research.

The tests were carried out on parameters with the following values.

- amount_of_feats_to_predict—[1, 2],
- corr_cutoff_bound—[0.85, 0.9, 0.95],
- linear_reg_to—[0.7, 0.85],
- next_predictor_provider—['lin', 'dec_tree'] (where 'lin' denotes linear regression, 'dec_tree' stands for a decision tree),
- order—[['HB', 'UTS', 'YS', 'A', 'K'], ['YS', 'HB', 'UTS', 'K', 'A']],
- req_neighs—[5, 10, 20].

In the case of the max_steps parameter it was decided that it would not be tested with different values. For each of the runs, it was set to the value of 15. This was to limit the time needed to run all parameter configurations. From the combination of all possible parameter values, 144 unique sets were obtained.

3.4.1 Model training with a machine learning algorithm

The Random Forest algorithm belongs to the 'ensemble learning' type methods, i.e. those that generate many simple models and aggregate their results. In the case of this algorithm, these simple models are decision trees. This algorithm was proposed by Breiman [14]. It consists in adding an additional layer of randomness to the Bagging algorithm [15]. In addition to constructing each tree with a different bootstrap data sample, random forests change the way classification or regression trees are constructed. In decision trees, each node is split using the best split of all the variables. In a random forest, each node is split using the best of the subset of the predictors randomly selected on that node. This somewhat counterintuitive strategy turns out to work very well when compared to many other classifiers, including discriminant analysis, support vector machines, and neural networks, and is resistant to overfitting [14]. The Gradient Boosting algorithm is another example of an 'ensemble learning' method. It is one of the best techniques for building predictive models. Gradient Boosting is a generalization of the AdaBoost algorithm [9]. It was created and described by Friedman [16] in 1999. Currently, the latest version of Gradient Boosting is XGBoost (Extreme Gradient Boosting), which was proposed in 2016 by Tiangi Chen [17]. This is again a method that uses many simple models that are decision trees and the final result depends on all of them. XGBoost uses an incremental strategy because it is a simpler and less time-consuming task than training all trees at once. An innovation in relation to Friedman's algorithm is the introduction of regularization. It was used as a penalty for having too

many leaves in the decision tree. This is how the complexity of the model is controlled. The Multilayer Perceptron [18] is a fully connected neural network of the feed-forward type, which is, in fact, a double-decker version of a single-layer perceptron. Unlike single-layer perceptrons, it is used to solve nonlinear problems. The activation functions defined in the layers enable networks based on multi-layer perceptrons to derive a non-linear model. Each multilayer perceptron consists of at least 3 layers. The first is the input layer that takes a vector of input values. The next layer is a hidden layer through which information passes from the input layer to the next hidden layer or to the output layer. The last layer, called the output layer, gives us the values predicted by the network. Training of such a network is performed using the error backpropagation and stochastic gradient slope algorithm 7.

The Random Forest algorithm, in the case of the implementation from the scikitlearn library, has many parameters that affect its operation.

Model training with the Gradient Boosting algorithm was carried out in the same way as the model training with the Random Forest algorithm.

Algorithm parameters used during the tests:

- learning_rate,
- subsample - a coefficient determining what proportion of training samples will be randomly selected for training before the trees grow,
- max_depth - maximum tree depth,
- eval_metric - metric used to evaluate the model,
- booster - the type of booster used.

The combination of all possible parameters creates 216 unique sets. As in the previous algorithm, the random_state parameter was set with one value for each training process.

To explore the parameter space, the GridSearchCV tool was used again.

The Keras interface from the TensorFlow library was used to create and train models using the Multilayer Perceptron algorithm. It allows for a clear and simple definition of the neural network architecture by creating a sequential model to which we add subsequent layers of the network. Such a model is finally compiled using the indicated stochastic slope along the gradient method and the loss function.

The created neural network models consist of three network layers:

- input layer with 14 neurons,
- a hidden layer for which the following parameters and their values were investigated:
- activation function: hyperbolic tangent,

- L2 regularization for weights, biases and layer output,
- output layer with one neuron.

Models are compiled using the Adam optimizer with the 'learning_rate' parameter set to 0.1 and the loss function defined as a mean square error.

The learning process was carried out using the Early Stopping method, which allows to prevent overfitting of the network by checking whether the value of the loss function for validation samples changes during the learning process.

The model was developed on the basis of research published in [5]. The Ensemble averaging algorithm consists in creating many models with different parameters and combining them into one model in such a way that the input and output of such a model do not differ from a single model. The model output is combined and the final model output is the average value of the outputs of the models included in the kit. The models in the network set were trained with different regularization constants and different numbers of neurons in the hidden layer. To limit the number of different regularization constants, a study was conducted to see which constants would perform best for models with 10 and 25 neurons in the hidden layer. The tested constants for weights, biases and layer outputs were the following values: [1e-1, 1e-2, 1e-3, 1e-4, 1e-5]. The study was conducted using a fivefold cross-validation along with collecting R2 metric values and the mean absolute error.

Parameters with which the models were trained:

- activation function: hyperbolic tangent,
- regularization constants,
- number of neurons in the hidden layer: [3 ... 30],
- optimizer: Adam with the learning_rate parameter equal to 0.01,
- optimized function: mean square error,
- maximum number of epochs: 3000,
- load size: 10,
- number of epochs with no change in the value of the loss function after which the learning process will be completed: 100.

The models for construction were selected in two ways: • The model configuration ranking was created, in which the place was determined by the mean value of the R2 metric or the average absolute error of all divisions of the data set from the cross-validation. For each division of the set, a model was built consisting of the n best models according to the created ranking, and the final quality of the model was calculated as the average value of the metrics of each model.

- For each of the splits, a model was created, where it consisted of the best n models in relation to the R2 metric or

the average absolute error for a given division of the data set. The final model quality was calculated as the average value of each model's metrics from each split. The first strategy will be labeled as "avg" (from the average value of the metric) and the second as "split" (from the cross-validated split). A different number of models included in the kit were tested. A range of 2 to 30 models was tested for each of the mechanical properties.

The evaluation of the quality of the models was carried out for each algorithm separately. For each of the algorithms, the best 5 configurations of parameters were presented, the evaluation of which was carried out using cross-validation and metrics of the mean absolute error (MAE) and the coefficient of determination (R2). Additionally, the worst configurations are presented for comparison and drawing conclusions. It should also be noted here that the models were also trained on non-supplemented data.

Parameters tested:

- n_estimators - [10, 50, 100],
- max_depth - [15, 20, 25, 30],
- max_features - ['auto', 8, 9, 10, 11, 12] ('auto' - number of input dimensions),
- criterion - ['mse', 'mae'] ('mse' - mean square error, 'mae' - mean absolute error),
- min_samples_split - [2-4].

The combination of all possible values of the above parameters creates 432 unique sets, which together with the fivefold cross-validation result in 2880 training processes. Each of the models was trained with the same "random_state" parameter that controls the seed of the random number generator while the algorithm is running. The search of the parameter space was performed with the GridSearchCV tool from the scikit-learn library. The best models for each of the metrics considered are trained over the entire dataset.

Gradient Boosting

The parameter space was built with the following possibilities:

- learning_rate - values: [0.01, 0.1, 0.3],
- subsample - coefficient determining how much of the training samples will be randomly selected before the trees grow, values: [0.3, 0.5, 1.0],
- max_depth - maximum tree depth, values: [6, 10, 20, 30],
- eval_metric - a metric to evaluate the model, value: ['rmse', 'mae'],
- booster - value: ['gbtree', 'gblinear', 'dart'].

The combination of all possible parameters creates 216 unique sets. As in the previous algorithm, the random_state parameter was set with one value for each training process.

To explore the parameter space, the GridSearchCV tool was used again. The time needed to train all models was 10 s. First evaluation criterion was the average absolute error (MAE) metric, and the second was the coefficient of determination (R^2). Configurations with the 'learning_rate' parameter equal to 0.01 were not taken into account during the research due to the very poor quality of the results, due to an insufficient number of learning steps.

Multilayer Perceptron

Parameters tested:

- L2 regularization constants for weights, biases and layer outputs - [1e-1, 1e-2, 1e-3],
- the number of neurons in the layer: [20, 30, 40],
- batch size: [32, 64],
- number of epochs: [500, 1000],
- number of epochs without changes in the value of the loss function after which the learning process will be completed: [50, 100, 200].

The combination of all possible parameters creates 972 unique sets. The entire space of possible parameters was cross-validated in the same way as for the two previous algorithms. The test set determined by the cross-validation served as the validation set for the Early Stopping method. First evaluation criterion was the metric of the mean absolute error (MAE), and the second was the coefficient of determination (R^2).

Ensemble Averaging

To limit the number of possible configurations of regularization constants and at the same time to ensure that the results obtained with their participation are the best ones, preliminary studies have been carried out. The subject of these studies was to find 4 configurations of regularization constants for each of the mechanical properties with the following characteristics:

- the best configuration in terms of R^2 metric for 10 neurons in the hidden layer,
- the best configuration in terms of R^2 metric for 25 neurons in the hidden layer,
- the best configuration from the point of view of the MAE metric for 10 neurons in the hidden layer,
- the best configuration in terms of MAE metric for 25 neurons in the hidden layer.

The best configurations were searched for from the values [1e-1, 1e-2, 1e-3, 1e-4, 1e-5] for the regularization constants for weight, biases and output. Given 10 and 25 neurons in the hidden layer and 5 mechanical properties, this gives a total of 1,250 different configurations to study. In the next step, more models were trained. Throughout the process, 2800 different models were trained taking into account

Table 1 Table of the best models for all mechanical properties broken down into the best selection strategies and metrics

Properties	Strategy	Metric	Number of models	R^2	MAE
UTS	Avg	R^2	6	0.794	63.594
		MAE	10	0.791	63.678
	Split	R^2	4	0.810	61.533
Rp02	Avg	R^2	8	0.808	65.402
		MAE	5	0.802	65.749
	Split	R^2	2	0.836	62.612
A	Avg	R^2	4	0.832	59.652
		MAE	4	0.832	59.652
	Split	R^2	4	0.710	1.320
HB	Avg	R^2	5	0.701	1.300
		MAE	5	0.701	1.300
	Split	R^2	3	0.837	18.988
K	Avg	R^2	3	0.835	18.974
		MAE	5	0.849	18.062
	Split	R^2	3	0.855	18.089
K	Avg	R^2	8	0.683	14.732
		MAE	10	0.686	14.387
	Split	R^2	5	0.712	13.808
		MAE	4	0.713	13.612

Bold values indicate the best result

5 different mechanical properties and fivefold cross-validation and different values of regularization constants and the number of neurons in the hidden layer. In the next step, using two developed model selection strategies—'avg' and 'split', neural network models were built, which included from 2 to 30 best models. For each of the strategies, the models were selected in terms of the best results for the coefficient of determination and mean absolute error metrics. Models were built using models trained from one subset of cross-validated samples. When analyzing the graphs of the dependence of the number of models on their quality, it can be noticed that increasing the number of models improves the quality of the results only up to a certain point. Then a tendency to decline in quality as the number of models increases can be noticed. This proves the bad influence of too many model parameters on the quality of the prediction. There is also a noticeable advantage of the strategy of selecting models from the best models trained on a given subset chosen from sets determined during cross-validation ("split" strategy). The strategy of determining the best configurations, based on the best quality scores presented as the average of all the cross-validation splits, gives slightly worse results. Using this strategy is less deleterious to quality when increasing the number of models is visible. The exact best results are shown in Table 1. For each of the mechanical properties,

4 are presented in terms of each strategy and metric used to select the best models. The best results for each of the mechanical properties were achieved using the 'split' strategy. The number of entering models did not exceed 5 for the R^2 metric. For the MAE metric, only in the case of the model for the UTS property, the number of models for the best result was 10.

3.4.2 Research conclusions

Random Forest

- For each of the mechanical properties, the top five models are very close to each other in terms of prediction quality and could be used interchangeably although there is a difference in performance. The parameter 'criterion' with the value of 'mse' turned out to be better than 'mae' in almost all cases (except for the models for the UTS property, where the quality was measured using the R^2 metric), which means that optimizing the mean square error brings better results than optimizing mean absolute error (all worse models were optimized in this way),
- It can also be seen that choosing the 'n_estimators' parameter (number of trees) from the value 10 gave the worst models, which clearly indicates that the more trees, the better the model will be, but also more complicated,
- The 'max_depth' parameter does not significantly affect the quality, when for each of the mechanical properties the best models were trained with each of the available values,
- The "max_features" parameter with "auto" value has not worked for any of the best models, which may mean that a better choice would be to impose a specific number of dimensions in advance taken into account when building trees. It can also be noticed that higher values of this parameter (12, 11, 10) performed better in the best models in terms of the mean absolute error metric, in the case of the coefficient of determination, smaller values (8, 9, 10) prevail,
- The "min_samples_split" parameter (the minimum number of samples needed to split a node) was in most of the best models with the value 2, which means that the trees being built have a large number of leaves. The only exception were the models for the A property, in which the quality was measured by the determination coefficient metric, where the tested parameter appeared in 4 out of 5 models with a value of 4. This means that trees with fewer leaves performed better,
- Visible differences between the best and the worst models in terms of quality in both metrics allow for the conclusion that the process of tuning parameters was needed in this case

Gradient boosting

The 'learning_rate', 'sub_sample' and 'param_booster' parameters had the greatest impact on the quality of the predictions because their values for the best models are in most cases the same,

- The 'max_depth' parameter with a value of 6 occurred only in the best configurations for models predicting the K property. In the case of other properties, the value of this parameter for the best models was different. In the case of the worst models, the most common value was 10,
- The 'param_booster' parameter with the 'dart' value was more successful in the case of the best configurations for the MAE metric and in the case of models for UTS properties assessed with the R^2 metric, it proves that in these cases the use of the drop-out mechanism positively influenced the quality of predicted values. The 'gblinear' value did not occur in any of the best configurations, it can be stated that the linear regression models do not give good results with the boosting gradient.

Multilayer perception

- In all the best configurations, the number of neurons in the hidden layer ('units' parameter) was 30 or 40 (with a predominance of 40). This is understandable due to the large number of dimensions that are passed to the input layer,
- The maximum number of epochs in most cases was 1000, which means that 500 epochs were insufficient in most cases,
- The batch size had little effect on the best results as it occurred with both values
- The regularization constant value for balances in most cases has a single value for the best configurations and it can also be seen that for the properties of UTS, YS and HB, the value 0.1 works best.

Comparison of the quality of the tested models

To assess which machine learning algorithms worked best for the collected data, Table 2 presents the best results for each algorithm in terms of the R^2 metric. It can be clearly indicated that the best algorithm turned out to be Gradient Boosting. It achieved the best results for each of the mechanical properties. For the UTS and YS properties, the Ensemble Averaging algorithm worked well, with worse results in the remaining cases. The biggest surprise is the result of the Random Forest algorithm for the HB property, where it took second place and in the case of the remaining properties it was in the last and penultimate places.

Table 2 Table showing the summary of the best results for all used algorithms for all mechanical properties

Properties	Algorithm	R^2
UTS	Gradient boosting	0.8243
	Ensemble averaging	0.810
	Random forest	0.8081
	Multilayer perceptron	0.8011
YS	Gradient boosting	0.8513
	Ensemble averaging	0.836
	Multilayer perceptron	0.8287
A	Random forest	0.8054
	Gradient boosting	0.7296
	Random forest	0.7121
	Ensemble averaging	0.710
HB	Multilayer perceptron	0.6890
	Gradient boosting	0.8791
	Random forest	0.8674
	Ensemble averaging	0.855
K	Multilayer perceptron	0.8416
	Gradient boosting	0.7694
	Random forest	0.7230
	Ensemble averaging	0.713

3.4.3 Objectives and observations achieved

The aim of the work was to develop a solution that would enable to predict the final parameters of selected castings, and the training data set required to be supplemented. The solution is based on data from the ADI research. The data was collected from the literature and had been assessed by experts. Then an algorithm was developed to fill in the missing data. To achieve results, 4 complex machine learning algorithms were selected and tested: Random Forest, Multilayer Perceptron, Gradient Boosting and Ensemble Averaging. As a result of the conducted research, prediction models of mechanical properties of castings were developed on the example of ADI, the best ones turned out to be those trained with the Gradient Boosting algorithm. Further verification consisted of the physical execution of experimental castings. It should be remembered that the standardization of

the hardness measurement was performed using the hardness conversion table described in the literature.

3.5 Verification of the obtained results with the use of a physical experiment

On the basis of the developed solution, selected chemical compositions of materials that meet the design requirements were indicated.

– Laboratory melts of these alloys (Table 3) were carried out and the obtained chemical composition of the test ingots was determined. The melts were carried out in the RADYNE medium frequency induction furnace in a crucible with an inert lining based on Al_2O_3 . A crucible with a capacity of 100 kg of the mass of the metal charge was used. The following were used as input materials:

– pig iron with the following composition:

C – 4.44%,
Si – 0.97 %,
Mn – 0.05 %,
P – 0.05 %,
S – 0.013 %,

– steel scrap with composition:

C – 0.1%,
Si – 0.02 %,
Mn – 0.3 %,
P – 0.02 %,
S – 0.02 %,

- ferroalloys,
- deoxidizers,
- FeSiMg17 master alloy
- FeSi75T inoculant.

Samples for chemical composition analysis were cast to metal molds (copper mold), and for the remaining laboratory tests and experimental castings—to green sand molds. During the melts, it was found out that, while maintaining the appropriate technological regime (type, temperature and time of metallurgical treatments), all these alloys do not pose any major difficulties in terms of the technology of melting, pouring and solidification in ceramic molds. The chemical analysis of the melts was carried out using

Table 3 Mechanical properties of selected alloys (minimum values or their limits)

Parameter	GJS-1400-1	GJS-1200-2	GJS-1000-5	G20NiMn-CrMo5-4-3	G120MnCrMo13-1	G22NiCrMo5-6
UTS [MPa]	1400	1200	1000	1300	900–1100	930–1080
YS [MPa]	1100	850	700	1100	350–450	825
A [%]	1	2	5	6	5	10
HB [–]	380–480	340–440	300–360	388–477	200–250	300–375
E [GPa]	165	167	168			212
ν [–]	0.27	0.27	0.27			

Table 4 Obtained mechanical properties of ductile iron hardened with isothermal transformation (ADI)

Thermal treatment	UTS [MPa]	YS [MPa]	A [%]	HRC [–]	KC [J]
Preliminary tests-smelting 1					
Variant 1	1480	1130	2.4	43	54.5
Variant 2	1170	830	5.6	36	72.4
Variant 3	950	580	8.2	27	80.6
Verification tests-smelting 2					
Variant 4	1430	1280	1.7	39	56.1
Variant 5	1315	1170	2.4	37	62.2
Variant 6	1200	1020	6.4	33	79.4
Variant 7	1030	780	5.7	25	77.5

Variant 1: austenitization 900 °C/2 h; isothermal quenching in salt bath 270 °C/3 h
 Variant 2: austenitization 900 °C/2 h; isothermal quenching in salt bath 330 °C/2.5 h
 Variant 3: austenitization 900 °C/2 h; isothermal quenching in salt bath 375 °C/2.5 h
 Variant 4: austenitization 850 °C/2 h; isothermal quenching in salt bath 300 °C/3 h
 Variant 5: austenitization 900 °C/2 h; isothermal quenching in salt bath 320 °C/2 h
 Variant 6: austenitization 900 °C/2 h; isothermal quenching in salt bath 340 °C/2 h
 Variant 7: austenitization 950 °C/2 h; isothermal quenching in salt bath 360 °C/1.5 h

the emission spectrometry method on the POLYVAC 2000 device (Hilger Co, Great Britain) in accordance with the certified research procedures developed at the Foundry Research Institute in Krakow (currently the Łukasiewicz Research Network—Krakow Institute of Technology) (Table 4).

3.6 Summary

The collection of data is an important aspect in the case of creating an intelligent system or its part (module) aimed at supporting the production of metal products, an important aspect is the collection of data. In the presented case, the collected data concerned the production of ADI. An important aspect of this solution was the attempt to solve the problem related to the lack of data. Lack of data was considered in the context of the IT system. This deficiency is related both to the number (in the context of the number of trials) and to the parameters measured during the process (not all parameters important for the system were and are measured or recorded). To develop and create predictive models and decision support modules, in the event of not owning complete data, the decision was made to use the area of production of products for which modern methods are applied and some experience and knowledge are possessed so that the results obtained as a result of the system operation can be verified not only experimentally (by making appropriate castings), but also so that some adjustments can be made during the course, based on the knowledge and experience of production technologists. This subject is taken up in scientific research.

The article presents the IT tools originally compiled by the authors designed to aid in the material-technological

conversion process, which is related to various activities resulting in the development of a machine metal part [1]. In particular, the process may refer to changing the target application of an element without changing its shape. Another approach allows for consolidation, i.e. the possibility of reconciling different construction and technological versions. It can also refer to regeneration, i.e. the renewal of old technological concepts. Conversion can also refer to both design and technology changes, and as such it should be understood as a change in the manufacturing technology, resulting in increased durability and quality of components, reference to new technological trends in the context of product manufacture, and reduction in operating costs of the component being created. Prototype castings of chosen parts were made using the developed algorithm and then examined in the field. The prototype series of castings, parts of the machines working in forests and farms, were transferred to operational tests at selected agriculture farms. The production of these castings was preceded by a computer simulation of the casting technology developed and by testing the selected casting alloys. During operational tests, the durability of the experimental cast tools produced as part of the project was compared with the durability of conventional forged and welded tools. The analysis covered the manufacturing technology and operating conditions of the components, including also testing of materials currently being used for these components. Important aspect of the decision-making process is to determine if the conversion is economically viable. If it is and if being at the same time is possible then the design of the components should be modified to allow their manufacture by casting technology. Due to technological change, the shape of the developed design is also modified. The parameters of a given material should

be selected in accordance with the requirements. Often, the selected casting alloy meets the construction and operating requirements applied to the previously used material. If an alloy fulfilling these requirements is not available, research works are undertaken to develop a suitable material. One of the methods of obtaining an alloy with the required parameters is the modification treatment, i.e. changing the chemical composition or conducting a heat treatment or thermochemical treatment. The essential aspect of this procedure is to enable a technological process to proceed according to order in such a way as to minimize the occurrence of casting defects.

Funding This study was funded by This work was supported by project TECHMATSTRATEG 1 "Development of innovative working elements of machines in the forestry sector and biomass processing based on high-energy technologies of surface modification of the surface layer of cast elements"; TECHMATSTRATEG1/348072/NCBR/2017.

Declarations

Conflict of interest The authors have not disclosed any conflict of interests.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors. All applicable international, national, and/or institutional guidelines for the care and use of animals were followed (In case animals were involved).

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Olson D. Prediction of austenitic weld metal microstructure and properties. *Weld J*. 1985;64(10):281s–95s.

2. Kochanski A, Perzyk M, Klebczyk M. Knowledge in imperfect data. In: *Advances in knowledge representation*. London: IntechOpen; 2012. p. 181–209
3. Arafeh L, Singh H, Putatunda SK. A neuro fuzzy logic approach to material processing. *IEEE Trans Syst Man Cybern Part C (Appl Rev)*. 1999;29(3):362–70.
4. Yescas MA. Prediction of the Vickers hardness in austempered ductile irons using neural networks. *Int J Cast Met Res*. 2003;15(5):513–21.
5. Yescas M, Bhadeshia H, MacKay D. Estimation of the amount of retained austenite in austempered ductile irons using neural networks. *Mater Sci Eng A*. 2001;311(1):162–73.
6. PourAsiabi H, PourAsiabi H, AmirZadeh Z, BabaZadeh M. Development a multilayer perceptron artificial neural network model to estimate the Vickers hardness of Mn–Ni–Cu–Mo austempered ductile iron. *Mater Des*. 2012;35:782–9.
7. Savangouder RV, Patra JC, Bornand C. Artificial neural network-based modeling for prediction of hardness of austempered ductile iron. In: *International conference on neural information processing*. Cham: Springer; 2019. p. 405–413.
8. Savangouder RV, Patra JC, Bornand C. Prediction of hardness of austempered ductile iron using enhanced multilayer perceptron based on Chebyshev expansion. In: *International conference on neural information processing*. Cham: Springer; 2019. p. 414–422.
9. Russell SJ, Norvig P. *Artificial intelligence: a modern approach*. Financial Times Prentice Hall; 2019.
10. Sambridge M. Parallel tempering algorithm for probabilistic sampling and multimodal optimization. *Geophys J Int*. 2013;196(1):357–74.
11. Glover F. Tabu search: a tutorial. *Interfaces*. 1990;20:74–94.
12. Kumar A, Chakrabarti D, Chakraborti N. Data-driven pareto optimization for microalloyed steels using genetic algorithms. *Steel Res Int*. 2012;83(2):169–74.
13. Radiša R, Ducić N, Manasijević S, Marković N, Cojbašić Ž. Casting improvement based on metaheuristic optimization and numerical simulation. *Facta Universitatis*. 2017;15(3):397–411.
14. Norm: EN 1564:2012
15. Breiman L. Random forests. *Mach Learn*. 2001;45(1):5–32.
16. Breiman L. Bagging predictors. *Mach Learn*. 1996;24(2):123–40.
17. Friedman JH. Greedy function approximation: a gradient boosting machine. *Ann Stat*. 2001;29(5):1189–232.
18. Chen T, Guestrin C. XGBoost: a scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, ACM Digital Library; 2016. p. 785–94.
19. Negnevitsky M. *Artificial intelligence: a guide to intelligent systems*. Pearson Education; 2005.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.