Analysis of the Possibility of Using Exploration and Learning Algorithms in the Production of Castings

Dorota Wilk-Kołodziejczyk[1,2,] *, Mateusz Witkowski[1], Krzysztof Jaśkowiec[1,2], Marcin Małysza[1,2], Adama Bitka[2], Łukasz Marcjan[1]

1. Department of Applied Computer Science and Modelling, Faculty of Metals Engineering and Industrial Computer Science, AGH University of Krakow, Czarnowiejska 66, 30-054 Krakow, Poland
2. Łukasiewicz – Krakow Institute of Technology, Zakopiańska 73, 30-418 Kraków, POLAND

*Corresponding address: e-mail: dwilk@agh.edu.pl,Dorota.wilk@kit.lukasiewicz.gov.pl

The paper is devoted to the study of the effectiveness of selected machine learning algorithms - Linear Regression, Decision Tree, Random Forest, AdaBoost, Gradient Boosting, XGBoost, SVR, K-Nearest Neighbors - in the issue of supporting the production of ADI cast iron. Knowledge and information on the production of cast iron and the effects of its chemical composition and heat treatment parameters on mechanical properties were obtained through an extensive literature review. Articles on topics similar to the subject of this work were also analyzed in the context of practical application of machine learning algorithms. As part of the work, hyperparameter tuning and data augmentation experiments were conducted to test the impact of the optimizations performed on the final results of the generated models. During the evaluation, metrics such as root mean square error and coefficient of determination were used with prior cross-validation allowing for more realistic results. For each of the predicted mechanical parameters, Gradient Boosting proved to be the best algorithm. The work culminated in a web application with a graphical user interface allowing interaction with the best predictive models created during the study.

The input data set consists of 513 records aggregating information on ADI cast iron production parameters:
•Chemical composition of cast iron – percentage of elements:
o C – Carbon
o Si – Silicon
o Mn – Manganese
o Mg – Magnesium
o Cu – Copper
o Ni – Nickel
o Mo – Molybdenum
o S – Sulfur
o V – Vanadium
o Cr – Chromium
o P – Phosphorus
o Ti – Titan
o Sn – Tin about Al-Glin
• Heat treatment process parameters:
o Austenitization temperature – in degrees Celsius
o Austenitization time – in minutes
o Ausferitization temperature – in degrees Celsius
o Ausferitization time – in minutes
• Product thickness – in millimeters:
• Mechanical parameters.

The purpose of this experiment was to check the model results for default parameters in order to obtain a reference point for subsequent tests. One configured parameter in this study was random_state in models using randomness. This was intended to obtain repeatable results. Model dictionary was passed as the input parameter "models_dict".

Brinell hardness models - HB:
• The best results for fitting the model to the training set were achieved by: DecisionTreeRegressor and ExtraTreeRegressor
• The best results for fitting the model to the training set with 5-fold cross-validation were obtained by: XGBRegressor
• The best results for fitting the model to the test set were achieved by: GradientBoostingRegressor
• The worst-fitting model for each set is: SVR

Models for tensile strength - Rm:
• The best results for fitting the model to the training set were achieved by: DecisionTreeRegressor
• The best results for fitting the model to the training set with 5-fold cross-validation were obtained by: ExtraTreesRegressor
• The best results for fitting the model to the test set were achieved by: ExtraTreesRegressor
• The worst-fitting model for each set is: SVR Models for yield strength - Rp02:
• The best results for fitting the model to the training set were achieved by: DecisionTreeRegressor and ExtraTreeRegressor
• The best results for fitting the model to the training set with 5-fold cross-validation were obtained by: ExtraTreesRegressor
• The best results for fitting the model to the test set were achieved by: XGBRegressor
• The worst-fitting model for each set is: SVR.

Elongation models - A5: • The best results for fitting the model to the training set were achieved by: DecisionTreeRegressor and ExtraTreeRegressor

• The best results for fitting the model to the training set with 5-fold cross-validation were obtained by: ExtraTreesRegressor
• The best results for fitting the model to the test set were achieved by: XGBRegressor
•The worst-fitting model for each set is: SVR Impact strength models - K
• The best results for fitting the model to the training set were achieved by: DecisionTreeRegressor and ExtraTreeRegressor
• The best results for fitting the model to the training set with 5-fold cross-validation were obtained by: ExtraTreesRegressor
• The best results for fitting the model to the test set were achieved by: XGBRegressor
• The worst-fitting model for each set is: SVR Summary: Analyzing the results of the basic study, it was noticed that algorithms based on decision trees are significantly overfitted. This is especially noticeable in the DecisionTreeRegressor and ExtraTreeRegressor models. In most studies, the mentioned algorithms matched the training set 100%, but when verifying the results using cross-validation or on the test set, the same algorithms performed much worse. Models using decision trees but based on ensemble learning were also overfitted, but showed significantly better generalization abilities, i.e. operating on new, previously unknown data that were not included in the training set. This is especially visible in the results on the test set. The exception in this case is the AdaBoostRegressor model, which uses weak regression models as base models, which, if the data is highly non-linear and contains many outliers, may negatively affect the results of the entire committee. LinearRegression and KNeighborsRegressor are models that assume a linear or locally linear relationship between the input parameters and the purpose of the calculation. When the actual problem is non-linear, the models mentioned may have difficulty fitting effectively. This is most likely why they performed so poorly in the study. The SVR model performed the worst in the entire study and this is most likely related to the default value of the "C" parameter, which for this algorithm is 1. Such a small value means that the model is very tolerant of prediction errors and tries to find a solution very close to linear regression.

References

[1] KochanskiA, PerzykM, KlebczykM. Knowledge in imperfect data. Advances in Knowledge Representation, 2012, 181-210.
[2] YescasMA. Prediction of the Vickers hardness in austempered ductile irons using neural networks. International Journal of Cast Metals Research, 2003, 15(5): 513-521.
[3] RojekI, Mik D, Kotlarzet P, al. Intelligent system supporting technological process planning for machining and 3D printing. Bulletin of the Polish Academy of SciencesTechnical Sciences, 2021, 69(2): 233178742.
[4] LiashchynskyiP. Grid search, random search, genetic algorithm: A big comparison for NAS. arXiv preprint arXiv, 2019, 1912:06059.
[5] Wu J, Chen X Y, Zhang H, et al. Hyperparameter optimization for machine learning models based on Bayesian optimization. Journal of Electronic Science and Technology, 2019, 17(1): 26-40.
[6] NaftalyU, IntratorN, HornD. Optimal ensemble averaging of neural networks. Network: Computation in Neural Systems, 1997, 8(3): 283.
[7] PavlyshenkoB. Using stacking approaches for machine learning models. In: 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP). IEEE, 2018, 255-258.
[8] AwadM, Khanna R. Support vector regression. Efficient learning machines: Theories, concepts, and applications for engineers and system designers. Efficient Learning Machines, Theories, Concepts, and Applications for Engineers and System Designers, 2015, 67-80.
[9] MammoneA, TurchiM, CristianiniN. Support vector machines. Wiley Interdisciplinary Reviews: Computational Statistics, 2009, 1(3): 283-289.
[10] SuX G, YanX, Tsai CL. Linear regression. Wiley Interdisciplinary Reviews: Computational Statistics, 2012, 4(3): 275-294.
[11] SongY S, Liang J Y, Lu J, et al. An efficient instance selection algorithm for *k* nearest neighbor regression. Neurocomputing, 2017, 251: 26-34.
[12] NoriegaL. Multilayer perceptron tutorial. School of Computing. Staffordshire University, 2005, 4(5): 444.