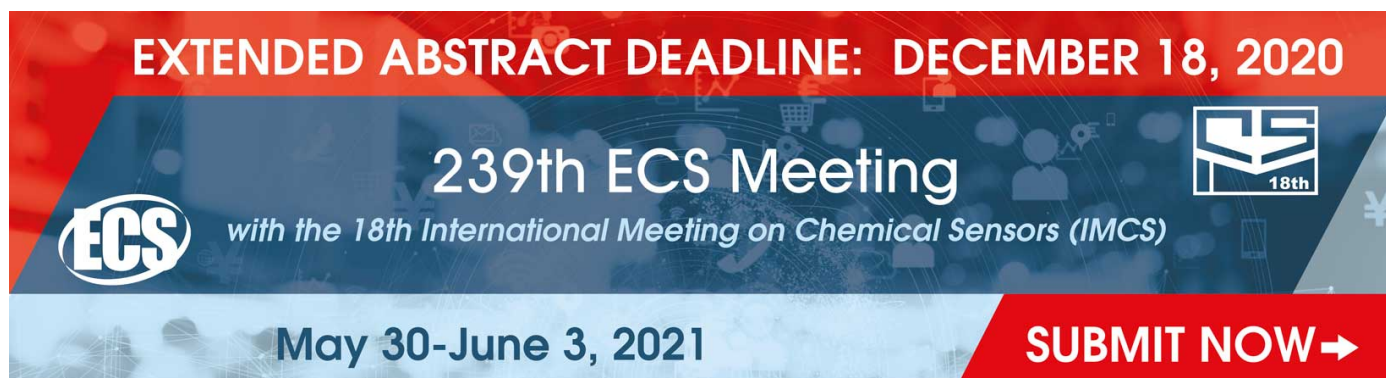


PAPER • OPEN ACCESS

Machine learning application for magnetohydrodynamic pump research

To cite this article: N V Tarchutkin and I A Smolyanov 2020 *IOP Conf. Ser.: Mater. Sci. Eng.* **950** 012019

View the [article online](#) for updates and enhancements.



EXTENDED ABSTRACT DEADLINE: DECEMBER 18, 2020

239th ECS Meeting
with the 18th International Meeting on Chemical Sensors (IMCS)

May 30-June 3, 2021

SUBMIT NOW →

Machine learning application for magnetohydrodynamic pump research

N V Tarchutkin, I A Smolyanov

Department of Electrical Engineering and Electrotechnology Systems, Ural Federal University, Yekaterinburg, Russia

Nik_tar@mail.ru

Abstract. The article is devoted to finding out best machine learning model used for the analysis of magnetohydrodynamic pump. The machine learning model was created on the basis of data obtained as a result of numerical simulation of the unit using COMSOL Multiphysics. The paper compares errors of output data collected by using various machine learning methods for out-of-sample data.

1. Introduction

Machine Learning (ML) is a form of predictive analytics that uses a set of training data to develop a mathematical model to extrapolate conclusions from input data. At the moment, machine learning is used as a research tool in thermoelectricity [1], construction [2], chemistry [3], engineering [4] and other scientific fields. This popularity is due to the ability to analyze large, complex and streaming data and find in them valuable information that allows to make predictions based on input data. However, they also have a disadvantage associated with a lack of understanding of the physical basis of the solution. In addition, for the machine learning approach to be successful in presenting the physics of the problem, a large and statistically reliable dataset must be used.

2. Formulation of the problem

The paper compares various ML algorithms used to analyze the data obtained during the simulation of the induction pump in the COMSOL Multiphysics software package (figure 1). The model is based on the geometric parameters of a real installation (table 1).

The data used for ML are obtained at various current amplitudes and current frequencies of the pump windings. Used libraries: scikit-learn (ML algorithms), pandas (data extraction), plotly (data visualization).

The objective of this work is to compare the error of the ML algorithms using out-of-sample data.



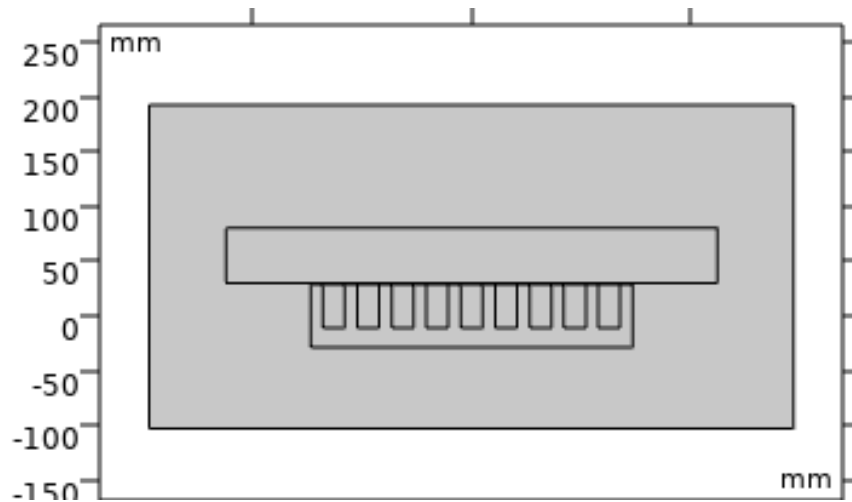


Figure 1. Scheme of the simulated induction pump.

TABLE 1. Initial data.

Description	Unit	Value
Inductor length	mm	295
SE length	mm	450
The number of turns in the slot	-	135
Number of slots	-	9
Inductor thickness	mm	58
Inductor width	mm	77
SE width	mm	78
Rectangular slot height	mm	40
Rectangular slot width	mm	20
Tooth width	mm	11.5
SE height	mm	50
Gap between the inductor and the SE	mm	1

3. Overview of algorithms

At the moment, the scikit-learn library contains several algorithms. Each of them receives a set of input and output data, based on which the dependencies between the variables are detected. The following is a description of the functions used in this paper.

4. Linear Regression

The model is based on the ordinary least squares method. The dependence of the output variable on the input is given by the equation:

$$y = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \dots + \omega_n x_n, \quad (1)$$

where n is a number of input variables, $\omega_1, \omega_2, \dots, \omega_n$ are the slope parameters, x_1, x_2, \dots, x_n are input variables, and ω_0 is an intercept.

The model selects the coefficients ω_i in such a way as to minimize the residual sum of squares between the predicted values of y and the actual values at the points from dataset [5]:

$$\min_{\omega} \|X\omega - y\|_2^2, \quad (2)$$

where $\|\cdot\|_2$ is the Euclidean norm.

5. SGD Regressor

Stochastic Gradient Descent (SGD) Regressor builds a plane using the method specified by the loss variable which can take the following values:

- loss="squared_loss": ordinary least squares;
- loss="huber": Huber loss;
- loss="epsilon_insensitive": linear Support Vector Regression;
- loss="squared_epsilon_insensitive": same as previous, but becomes squared loss past a tolerance of epsilon.

In this work model uses a method based on Huber loss function [5]. The same equation as in the case of Linear Regression is used to create the plane, but the choice of coefficients is aimed at minimizing the sum of Huber losses for all points of dataset. Losses for each point are calculated as follows:

$$L_{\delta}(a) = \begin{cases} \frac{1}{2}a^2 & |a| \leq \delta \\ \delta\left(|a| - \frac{1}{2}\delta\right) & \text{otherwise} \end{cases}, \quad (3)$$

where a is the difference between actual and predicted values, δ is the slope of the linear part of the function.

6. Ridge

The model uses the least squares method with L2 regularization:

$$\min_{\omega} \|X\omega - y\|_2^2 + \alpha \|\omega\|_2^2, \quad (4)$$

where α is the regularization strength ($\alpha \geq 0$) [5].

7. Lasso

Least Absolute Shrinkage and Selection Operator (Lasso) uses the ordinary least squares method with L1 regularization [5]. Optimization is performed according to the following formula:

$$\min_{\omega} \frac{1}{2n} \|X\omega - y\|_2^2 + \alpha \|\omega\|_1, \quad (5)$$

where $\|\cdot\|_1$ is the taxicab norm (or Manhattan norm).

8. *Lasso Lars*

The model uses the ordinary least squares method with the least-angle regression (LARS) algorithm. Optimization is performed according to the same formula as for the Lasso method.

9. *ElasticNet*

The model uses the least squares method with both L1 and L2 regularizations [5]. Optimization is performed according to the following formula:

$$\min_{\omega} \frac{1}{2n} \|X\omega - y\|_2^2 + \alpha \rho \|\omega\|_1 + \frac{\alpha(1-\rho)}{2} \|\omega\|_2^2, \quad (6)$$

where ρ is the component of L1 regularization ($0 \leq \rho \leq 1$).

10. *Passive Aggressive Regressor*

The model analyzes the data set points in turn and adjusts the values of slope parameters and intercept by using ε -insensitive hinge loss function:

$$l_{\varepsilon}(\omega, (x, y)) = \begin{cases} 0 & |\omega \cdot x - y| \leq \varepsilon \\ |\omega \cdot x - y| - \varepsilon & \text{otherwise} \end{cases}, \quad (7)$$

where ε is the permissible error value.

The initial values of the coefficients are taken equal to 0. Then, the loss function is calculated alternately for each data point. If the value of the loss function is 0, the coefficients remain unchanged, otherwise the coefficients are adjusted according to the formula [6]:

$$\omega_{i+1} = \omega_i + \text{sign}(y_i - \hat{y}_i) \frac{l_i}{\|x_i\|_2^2} x_i. \quad (8)$$

11. *Theil-Sen Regressor*

The model calculates the slopes of the planes, each of which passes through three points that are not on the same line and do not have the same X coordinates. The result is a plane with a slope determined by the spatial median of the calculated slopes.

The intercept is determined from the condition of minimizing the sum of the distances from dataset points to the plane [7].

12. *Support Vector Machines*

With linear approximation, the model solves the problem of minimizing the objective function [8]:

$$\min_{\omega, \omega_0, \zeta, \zeta^*} \frac{1}{2} \|\omega\|_2^2 + C \sum_{i=1}^n (\zeta_i + \zeta_i^*). \quad (9)$$

The following conditions are met:

$$y_i - X_i \omega - \omega_0 \leq \varepsilon + \zeta_i, \quad (10)$$

$$X_i \omega + \omega_0 - y_i \leq \varepsilon + \zeta_i^*, \quad (11)$$

$$\zeta_i, \zeta_i^* \geq 0, i = 1, \dots, n. \quad (12)$$

13. *K-Nearest Neighbors*

To determine the value of a variable at a specific point, the model calculates the average value of the variable for the nearest points from the training data. In this work $k=4$. An example of predicted point with its nearest neighbors is shown in figure 2.

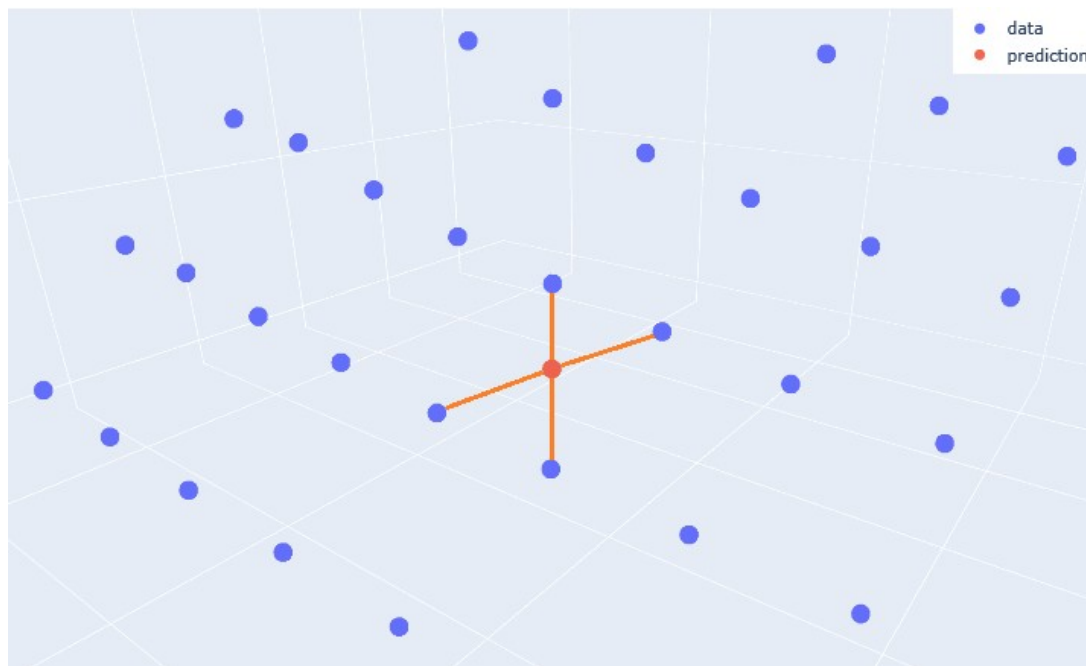


Figure 2. Prediction of output value based on KNN.

14. *Decision Trees*

The model creates an “if” condition tree, depending on the fulfillment of which the function will be assigned one of the values of the output variable obtained from the training data (figure 3).

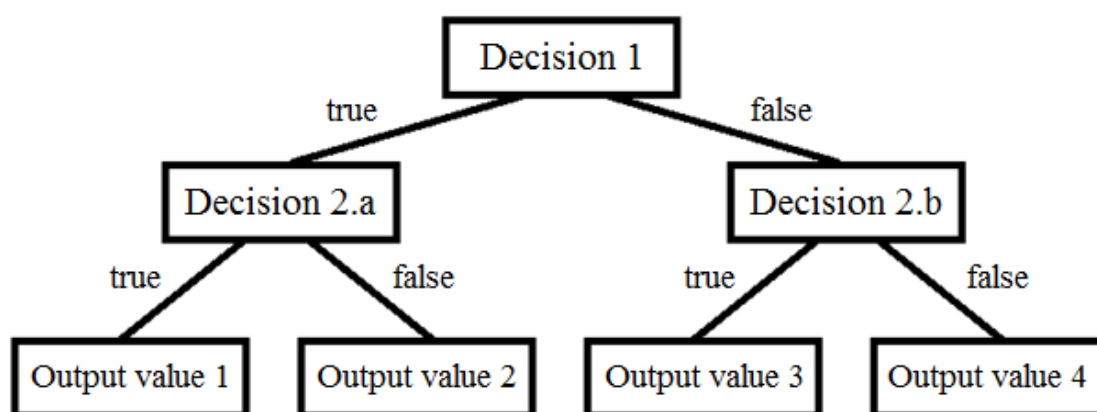


Figure 3. Example of Decision Tree.

15. *Analysis of results*

Comparison of algorithms is carried out according to the relative error of the results. To train the ML models, 80 results of solving the problem using COMSOL Multiphysics were used, while the verification was carried out using the other 63 solutions. Root-mean-square error (RMSE) and relative

error were calculated. For some models dataset was scaled in order to increase accuracy. The calculation results are presented in table 2.

Table 2. Errors of ML algorithms.

Model	Variable			
	B, mT		v, mm/s	
	RMSE	Relative error	RMSE	Relative error
Linear Regression	5.23	4.81	2156.38	241.7
SGD Regressor	9.36	12.65	3677.42	96.0
Ridge	5.24	4.81	2156.17	241.4
Lasso	5.24	4.81	2156.38	241.7
Lasso Lars	5.51	4.4	2156.26	241.5
ElasticNet	5.77	5.13	2150.19	233.5
Passive Aggressive Regressor	13.33	15.72	2156.89	273.3
Theil-Sen Regressor	7.25	6.41	2635.02	138.9
Support Vector Machines	4.30	4.96	2121.62	194.8
K-Nearest Neighbors	3.75	2.76	141.39	5.89
Decision Trees	16.77	19.75	1065.26	28.3

The noticeable difference in the errors of the same methods for different results is explained by the fact that the dependence of the magnetic flux density B on the amplitude and frequency of the current is close to linear, while for velocity v the dependence is nonlinear. Figures 4-6 show the set of input data used to create the model (green) and the results of machine learning (blue). It can be seen that the result of approximating strongly nonlinear dependences by most models cannot be called reliable.

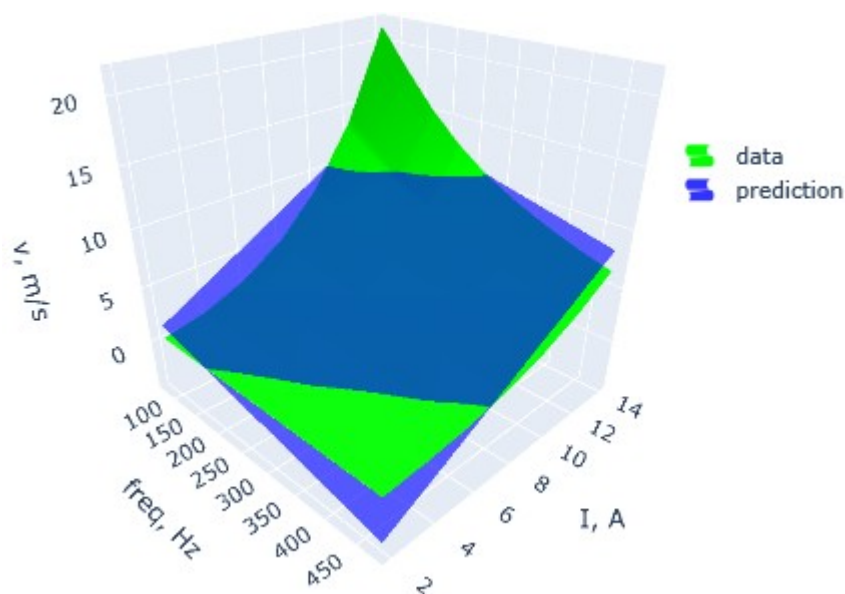


Figure 4. Dependence of the average velocity of the SE metal (m/s) on the frequency and amplitude of the inductor windings current (Linear Regression).

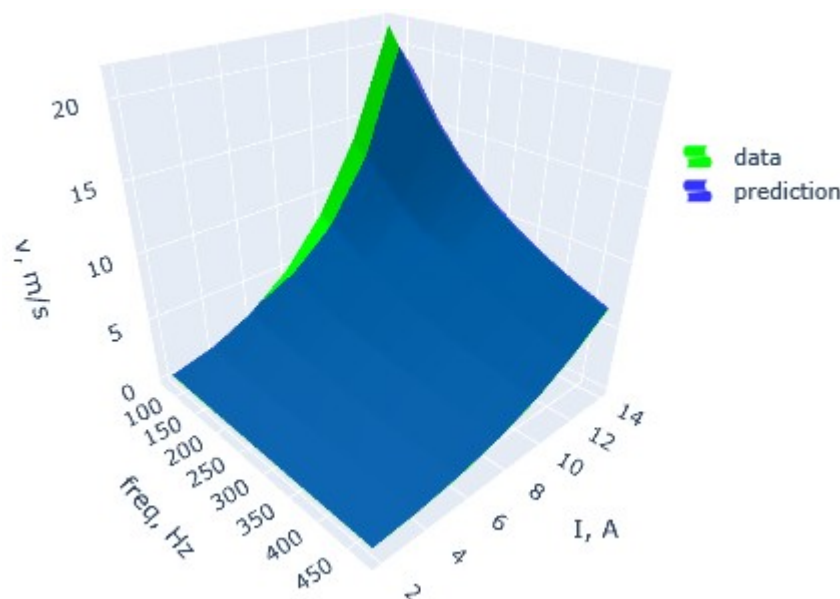


Figure 5. Dependence of the average velocity of the SE metal (m/s) on the frequency and amplitude of the inductor windings current (K-Nearest Neighbors).

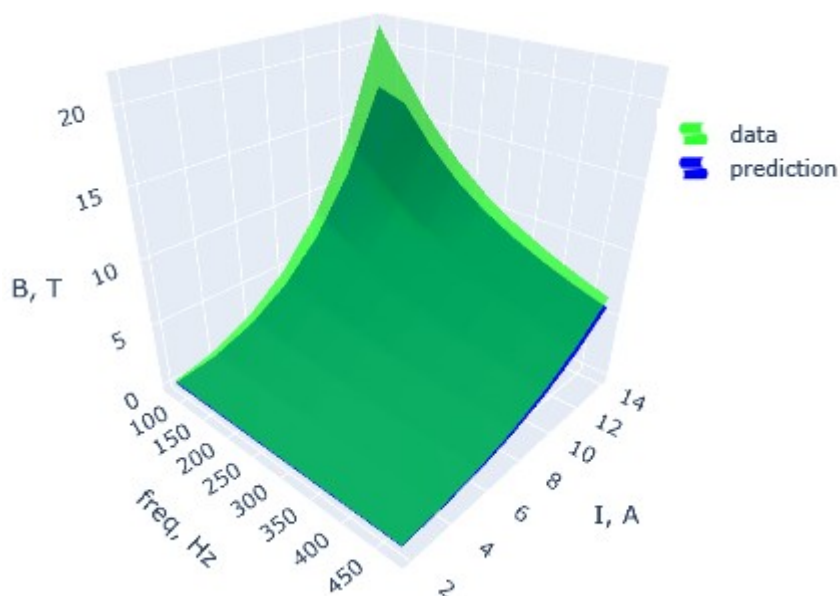


Figure 6. Dependence of the maximum magnetic flux density of the SE metal (T) on the frequency and amplitude of the inductor windings current (Decision Trees).

It should be noted that in the training data set, neighboring points are located in the nodes of the rectangular grid, while all test data points are located at the same distance from nearest training data, which significantly improves the quality of the KNN model. When examining data with an uneven distribution, it is necessary to use a KNN models with different values of the nearest neighbors. The quality of models is also affected by the number of training data points.

16. Conclusion

Based on the data presented, most of the methods considered in the work have high accuracy in predicting linear dependencies, while the K-Nearest Neighbors and Decision Trees methods show the best results with strongly non-linear dependencies.

References

- [1] Chen L, Tran H, Batra R, Kim C and Ramprasad R 2019 Machine learning models for the lattice thermal conductivity prediction of inorganic materials. *Computational Materials Science* **170** 109155
- [2] DeRousseau M A, Laftchiev E, Kasprzyk J R, Rajagopalan B and Srubar III W V 2019 A comparison of machine learning methods for predicting the compressive strength of field-placed concrete. *Construction and Building Materials* **228** 116661
- [3] Rizkin B A and Hartman R 2019 Supervised machine learning for prediction of zirconocene-catalyzed α -olefin polymerization. *Chemical Engineering Science* **210** 115224
- [4] Swischuk R and Allaire D 2019 A Machine Learning Approach to Aircraft Sensor Error Detection and Correction. *Journal of Computing and Information Science in Engineering* **4** no. 1
- [5] 1. Supervised learning – scikit-learn 0.21.3 documentation (2019). Available at: https://scikit-learn.org/stable/supervised_learning.html (accessed 25 November 2019).
- [6] Crammer K., Dekel O., Keshet J., Shalev-Shwartz S. and Singer Y. 2006 Online Passive-Aggressive Algorithms. *Journal of Machine Learning Research* **7** pp. 551–585
- [7] X. Dang, H. Peng, X. Wang and H. Zhang 2008 Theil-Sen estimators in a multiple linear regression model. *Tech. Rep.*
- [8] Smola, A. J. and Schölkopf, B. 2004 A tutorial on support vector regression. *Statistics and Computing* **14**(3) pp 199–222.