Comparison of two selected ML models for predicting deaths due to COVID-19 outbreak

Dominik Siekierski
Faculty of Electrical Engineering
Warsaw University of Technology
Warsaw, Poland
01101513@pw.edu.pl

***Abstract*—The accuracy of numerical data predictions depends on the ML algorithm and on the quality and quantity of the data set. A well-chosen ML model will allow to obtain better results. Comparison of two used algorithms will allow to analyse their quality for predicting deaths due to COVID-19 outbreak.**

Keywords—ML model, COVID-19, predicting deaths, algorithms comparison, MLP, SGD

I. Introduction

The main goal was to compare two ML models completely different from each other. The problem that is being solved is the prediction of the death of people suffering from the COVID-19 virus which, due to its global character and the long time during disease not manifest any symptoms, is widely known and discussed in the world. Used information relates to the epidemic status of this virus in Poland and comes from two separate sources with different default properties. This means that the models will be compared not only because of their difference and correctness of the final results but also the change in the amount of input data features. This will determine the flexibility of the algorithms. The first model analysed is the Multi-layer Perceptron (MLP) regressor. This model optimizes the squared-loss using LBFGS (Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm) or stochastic gradient descent [1]. The next analysed model is the linear model fitted by minimizing a regularized empirical loss with SGD. SGD stands for Stochastic Gradient Descent: the gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule [2]. More detailed information will be provided later.

II. Feature selection

Data on COVID-19 behaviour is taken from two sources. First was collected by GeoSiN Scientific Club Members from University of Warmia and Mazury (Olsztyn, Poland). The second source has data from the European Center for Disease Prevention and Control. Although these are different sources, most likely the original data is data officially disclosed by the Główny Inspektorat Sanitarny. The institutions behind the correctness of the data are reliable and therefore these sources and not others were selected. For easier data processing the information was downloaded from sites containing files in the CSV extension. This page [3] contains data that comes from the first source. There is information on the course of the epidemic such as the number of people hospitalised, in quarantine or the number of tests performed. The second page [4] contains data from a second source. They are only basic so they contain data on confirmed cases of illness, recovery and death. The data contain few features compared to the first source. This amount can influence the results but it can be useful for a good analysis of the models used.

Some data had to be processed into the correct numerical one. It was then possible to analyse and divide the data to feed the model. Some data had to be processed into the appropriate numerical type. The wrong type data also appeared and was removed or replaced in such a way that it would not create significant changes. The data could then be analysed and split to feed the model. The relevant functionalities of the pandas (manipulation tool, built on top of the Python programming language) library were used to load and process the data. To import the algorithms of the machine learning models used here a library of scikit-learn (scientific toolboxes built around SciPy) has been used.

III. Data analysis

Due to the nature of used data, a deeper analysis is unlikely to identify new and different dependencies. Thanks to the logarithmic scale, it can be seen on Figure 1 that the number of tests carried out increases faster than the number of confirmed cases. This may be related to greater self-awareness. This condition may also be affected by better test availability.

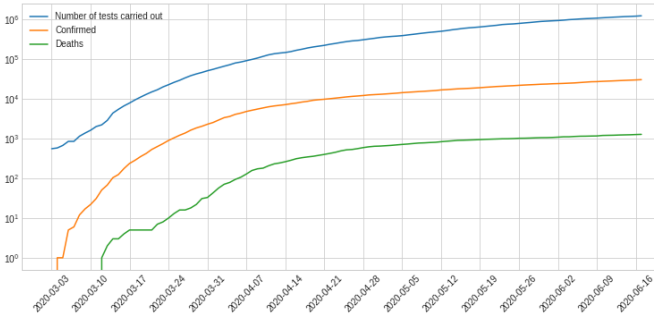

Figure 1. Logarithmic charts of selected data from the first source.

In the final interval number of deaths is increasing more slowly than number of cases found. This may be due to better medical care for patients. This is a new situation that takes time to adapt and develop certain procedures. The logistic process is also important, including the time between finding or suspecting a disease and isolating such a person from the rest of society. The next chart shows a very sharp change in the number of people in quarantine. As a society we have a lot of contact with people every day. One confirmed case may cause isolation of the entire workplace

in which this person worked. The initial off-day of this disease increase the number of people an infected person may have contacted. Major feature changes regarding the number of people in quarantine can be seen on Figure 2.

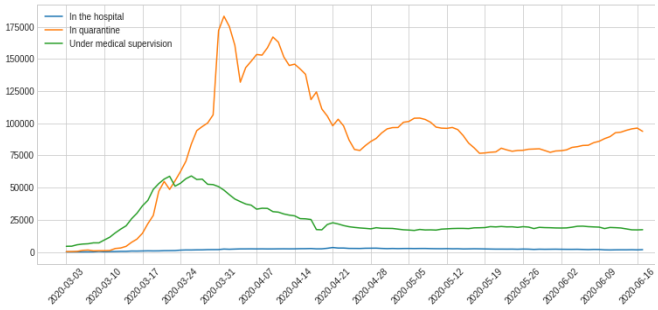

Figure 2. Line charts of selected data from the first source.

The pictures presented were from the first source that had more features. Below are charts based on the second source. It is comforting that the growth of the deceased is not so large and the number of healthy people who have undergone the disease has an increasing growth rate. It is clearly visible in the Figure 3 below.

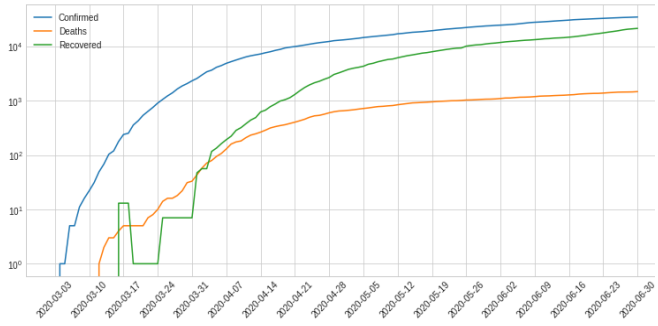

Figure 3. Logarithmic charts of selected data from the second source.

Some changes that are visible in the waveforms may be caused by different stages of isolation and restrictions introduced by the state authorities. After deeper analysis, conclusions can definitely be drawn as to the actual usefulness of restrictions. Which significantly affect the increase in morbidity and which have no effect. In this case, the number of days must be taken into account when the virus does not initially show any symptoms.

IV. Algorithms Description

The objective of the neural network is to transform the inputs into meaningful outputs [5]. In the MLP network, the layers are connected by neurons with different weights and the neurons are arranged in different ways. In each layer, the nodes receive input only from nodes from the previous layer and pass their output only to knots of the next layer. A MLP contains at least one hidden layer and each input vector is represented by a neuron. Number of hidden neurons varies and can be determined by trial and error so that the error is minimal [6]. Multi-layer Perceptron regressor trains iteratively since at each time step the partial derivatives of the loss function with respect to the model parameters are computed to update the parameters [1]. A regulating component can be added to the loss function that reduces the parameters of the model to prevent over-fitting.

The maximum number of iterations is 1000 and the flag to finish learning earlier when it does not improve it allows to get good results without overtraining. Early stopping determine whether to terminate training when validation score is not improving. It will automatically set aside 10% of training data as validation and terminate training when validation score is not improving. One of the parameters is the number of neurons in hidden layers, the values were the same for all data sources. This parameter was chosen in such a way as not to increase the time needed for calculations too much but also to allow results to be obtained which will not diverge significantly from the original course. At the beginning I used Adam as a solver for weight optimization. Adam refers to a stochastic gradient-based optimizer proposed by Kingma, Diederik, and Jimmy Ba [5]. But this optimizer has poor results for tasks that have a small amount of data. Therefore, it has been changed to LBFGS which is an optimizer in the family of quasi-Newton methods. In this and subsequent models random state flag has been set to achieve the same results again. The data was divided, allocating 80% to the learning process and the rest for test data.

Stochastic Gradient Descent is a simple yet very efficient approach to fitting linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines and Logistic Regression [4]. Stochastic gradient descent efficiently estimator maximum likelihood logistic regression coefficients from sparse input data [7]. Although SGD has long existed in the machine learning community, it has recently gained considerable attention in the context of large-scale learning. SGD has been successfully applied to large and rare machine learning problems commonly encountered in text classification and natural language processing. Given that data is scarce classifiers in this module easily scale to problems with more data. SGD in scikit-learn library introduces a simple routine procedure for learning stochastic gradient descent. Which supports various loss and penalty functions to match linear regression models. It is ideal for regression problems with a large number of training samples.

The loss function that was used in the model is squared loss. It refers to the ordinary least squares fit. Most of the parameters were left with default values due to measurable results of such a model and very fast learning time on learning data. The penalty for the model has been set as the default parameter to l2. It is the standard regularizer for linear SVM models. This model uses the function which construct a Pipeline from the given estimators. This estimator is Standard Scaler which standardize features by removing the mean and scaling to unit variance. Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set and next mean and standard deviation are then stored to be used on later data using transform [8]. Data set standardization is often used to get good results for many machine learning estimators. Because they may differ significantly from those expected if individual features not look like standard normally distributed data. The data was divided, allocating 80% to the learning process and the rest for test data.

V. Performance Metrics

The first measure comes from the built-in function of the model. This is the coefficient of determination (R^2) of the

prediction. The best possible result is 1.0, although it may be negative because model may be arbitrarily worse. A constant model that always predicts the expected value of y disregarding the input features, would get a R^2 score of 0.0 [3]. R^2 is the percentage of variation in effort explained by the variable size. A positive value represents a positive correlation. Larger coefficient values correspond to stronger correlations on the contrast negative values mean negative correlations [6].

Mean Absolute Error (MAE) measures the average magnitude of the errors in a set of predictions, without considering their direction [9]. It's average value over the test data sample of the absolute differences between prediction and actual value where all individual change have same weight.

Mean squared error (MSE) measures the average of the squares of the errors. That is average squared difference between predicted values and what is estimated. MSE is the risk function corresponding to the expected quadratic loss of error.

Root mean squared error (RMSE) is a quadratic scoring rule which measures the average magnitude of the error. Expressing the formula in words, the difference between forecast and corresponding observed values are each squared and then averaged over the sample.

Execution time is the last parameter measured. The specificity of the algorithm of the function used allows reliable measurement of the use of components. It is the sum of the system and user CPU time of the current process. It does not include time passed during sleep. Point of the returned value is undefined so that only the difference between the results of consecutive calls is correct.

VI. Results

The results, due to the short time of parameter modification and the small amount of data, deviate significantly from the actual values. The Figure 4 below shows that the MLP model based on data from the first source has high bias and variance seems unexpectedly small.

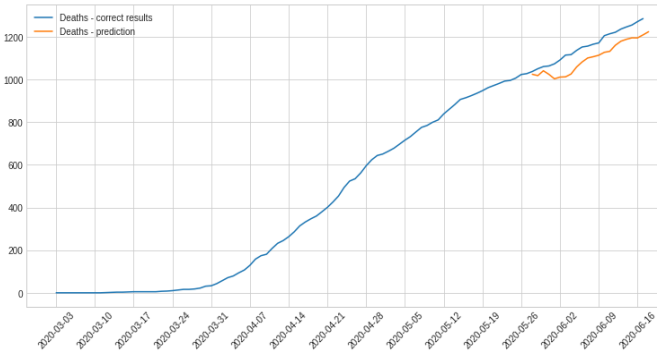

Figure 4. Graph showing the number of deaths from the first source with MLP model prediction.

This is certainly due to the large amount of data features. The second model for this data set also shows interesting results. It can be assumed that the initial data have significantly influenced the upward trend of the predicted values. It can certainly be said and seen on Figure 5 that the model generalizes results and is not over-trained.

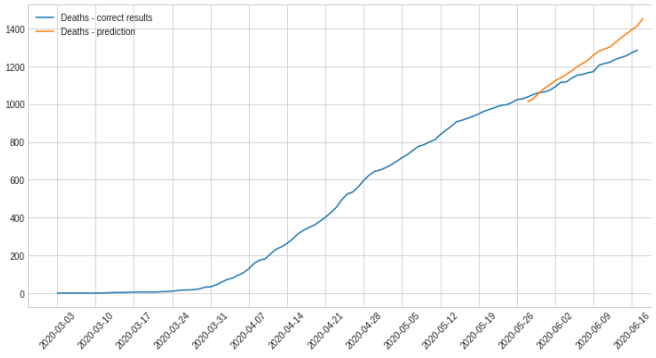

Figure 5. Graph showing the number of deaths from the first source with SGD model prediction.

Different model algorithm allows to achieve different results for the same data set. Then there will be results from the same model types but learning from scratch on data from a second source. Although there is more data. Number of features in this set is much smaller. The Figure 6 below shows the unexpected collapse of the expected values. It is caused by a sudden increase in the number of people recovering from the disease in this part of the chart.

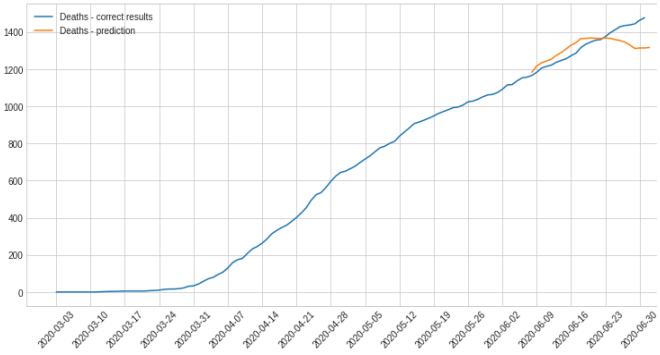

Figure 6. Graph showing the number of deaths from the second source with MLP model prediction.

This is not consistent with actual data, but it does indicate that the level of regulation of the trained model is not low. Recent results come from a model based on the SGD algorithm. The results are visible on Figure 7.

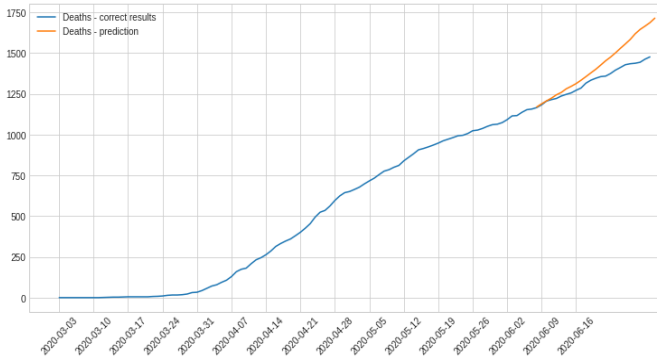

Figure 7. Graph showing the number of deaths from the second source with SGD model prediction..

It can be seen that, unlike earlier data, the beginning of the value has a very small error. The algorithm strongly reflects linearity due to the fact that the initial data is taken into account with the same weight as the final one.

VII. Performance

Two different types of machine learning models were analysed. The two sources that have been included here, though they relate to the same issue are slightly different. The Table 1 shows the results for a particular model depending on the data source.

TABLE I. Model results by measuring the coefficient of determination. Based on testing data.

Source	'MLPRegressor' model coef. R^2	'SGDRegressor' model coef. R^2
First ('dtandev')	0.53	-0.2
Second ('CSSEGISandData')	0.41	-0.52

The coefficient of determination has already been described. The MPL model did much better. Each of the algorithms has a better value for the first source which is due to the greater number of features in this data. The greater difference between the SGD model results may indicate that the configuration of this algorithm is less resistant to a smaller number of data features. Negative results of the SDG model mean that the model does not follow the data trend. The Table 2 confirms earlier assumptions.

TABLE II. Model results by measuring MAE, MSE and RMSE. Based on testing data.

Metric	'MLPRegressor' First source	'MLPRegressor' Second source	'SGDRegressor' First source	'SGDRegressor' Second source
MAE	50.93	62.92	75.08	99.41
MSE	2989.01	5693.79	7604.48	14783.66
RMSE	54.67	75.46	87.2	121.59

The first model coped much better with the task. The greater difference between MAE and RMSE in both models for the second source means the occurrence of errors with higher values during the forecast period. The Table 3 presents that learning time of algorithms depending on the data source.

TABLE III. Model results by measuring Execution Time. Based on testing data.

Source	'MLPRegressor' Execution Time [s]	'SGDRegressor' Execution Time [s]
First ('dtandev')	6.6635	0.008
Second ('CSSEGISandData')	2.1856	0.0057

First source with more data features significantly increased time needed to perform calculations. First model has a much greater difference in performance time. This is due to large number of neurons in MLP model and specificity of the algorithm.

VIII. Summary

The number of features in the learning data greatly affects the results. It can be concluded that in many cases it will be better to add more features of the data than to increase the amount of data without changing their specifics. MLP model due to large number of parameters, makes it possible to obtain good results. Unfortunately it is characterized by high resource consumption. Number of hidden neurons must be properly configured. Setting too low value may cause mismatch and setting this value too high may cause overshooting of the model. The SGD algorithm is simple to implement and needs a small amount of resources to perform calculations. Unfortunately, it is quite sensitive to feature scaling. Both models were not analysed in terms of the selection of parameters for this task. Obtaining better results is undoubtedly possible.

IX. Bibliography

[1] Scikit-learn developers, “MLPRegressor,” 2007. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html. [Accessed 29 June 2020].

[2] Scikit-learn developers, “SGDRegressor,” 2007. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDRegressor.html. [Accessed 29 June 2020].

[3] D. Tanajewski, “2020 Poland coronavirus data (COVID-19 / 2019-nCoV),” 2020. [Online]. Available: https://github.com/dtandev/coronavirus. [Accessed 29 June 2020].

[4] CSSE at Johns Hopkins University, „Novel Coronavirus (COVID-19) Cases, provided by JHU CSSE,” 2020. [Online]. Available: https://github.com/CSSEGISandData/COVID-19. [Data uzyskania dostępu: 29 June 2020].

[5] Taylor & Francis, “Multiple linear regression, multi-layer perceptron,” *Hydrological Sciences Journal*, vol. 61, no. 6, pp. 1001-1009, 2016.

[6] A. B. Nassif, D. Ho and L. F. Capretz, “Towards an early software estimation using log-linear regression and a multilayer perceptron model,” *The Journal of Systems and Software*, no. 86, pp. 144-160, 2013.

[7] B. Carpenter, *Lazy Sparse Stochastic Gradient Descent for Regularized Mutlinomial Logistic Regression*, 2018.

[8] Scikit-learn developers, “Standard Scaler,” 2007. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html. [Accessed 29 June 2020].

[9] Medium, “MAE and RMSE — Which Metric is Better,” 2016. [Online]. Available: https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d. [Accessed 29 June 2020].