COMPARING GRADIENT BOOSTING REGRESSION, LINEAR REGRESSION AND NEURAL NETWORKS IN PARLIAMENT PROPOSAL VOTING RESULT PREDICTION

CS-C3240 Spring 2021

1. INTRODUCTION

Decision-making in parliaments happens after a long process of information collection, expert hearings and legislation as well as regulation proposal preparation. When a proposal is formed and submitted to the parliament the proposal is presented in a plenary meeting and members of parliament (MP) cast their vote on whether to accept or reject the proposal. The votes can't be cast remotely. If a proposal is accepted the long process comes to an end. If the proposal in turn gets rejected, it has to be revised or all in all scrapped and thus all work hours used for preparation will have been used in vain. MP's can be absent from plenary meetings if they have other work-related matters at the same time, they're sick or on a maternity or paternity leave.

I find it interesting how the number of affirmative and negative votes as well as the absent-entries are divided with a specific proposal topic and time. Are there some kind of patterns emerging in terms of times of year when MP's are often absent? Are there similarities between the rejected proposals? What's common with the accepted proposals? The aforementioned questions can be examined by taking a look at previously gathered information through statistical analysis. Machine learning on the other hand enables *predicting an outcome of a proposal in a plenary meeting voting* based on the proposal topic and date of voting: what's the distribution between "yes" ("AanestysTulosJaa"), "no" ("AanestysTulosEi"), "empty" ("AanestysTulosTyhja") and "away" ("AanestysTulosPoissa"). In this project the proposal topic equals the proposal's headline handled as text. The application and model could be further developed so that the topics would be derived from the headlines and handled as classes what, however, lies beyond the scope of this project.

This report begins with introducing the data and formulating the problem in section 2. The discussion then moves on to splitting the data to training, testing and validation sets and continues by elaborating on the three different machine learning models considered for this particular application in section 3. In section 4 results obtained with the three different models will be introduced and in section 5 conclusions will be drawn.

2. PROBLEM FORMULATION

In this machine learning application data points represent individual plenary meeting proposal votings. A data point has the following features: headline of the proposal (column "KohtaOtsikko") as text and date of plenary meeting voting on the corresponding proposal. The label of a data point includes the number of affirmative votes, number of negative votes and number of absent entries.

Data of previous Finnish parliament's plenary meetings can be obtained from https://avoindata.eduskunta.fi/ where observations are stored from year 1996 onwards. Selected observations contain only those with the attribute "KieliId" having value 1 meaning that the observations are in Finnish. Observations of the corresponding attribute having value 2 are the same as the ones with value 1 but in Swedish. Upon exploration of the data it was found that some datapoints do not have a value in column "KohtaOtsikko". For example the parliamentary session of 17.12.1998 lacks values for all reports discussed in the session. Since the session's then non-digitally handled minutes are readable online no reason for the missing of values can be justified. To enable focus on later decisions and to access valid data in this project the earliest date included in training the model is chosen to be 16.12.2010.

3. METHOD

This particular machine learning application is an example of supervised learning and further of multilabel regression. Figure 1 shows that there is some correlation between the “yes” and “away” answers which in conclusion may refer to proposals getting accepted more easily when many MP’s are absent. This kind of inference seems reasonable and conclusions otherwise will be stated in subsequent sections.
The label values are numerical and the label set is continuous consisting of natural numbers. The feature space consists of feature vectors having the headline text (“KohtaOtsikko”) together with the voting date (“IstuntoPvm”) transformed into numerical features with Sklearn library [1]. Both the headline text and voting date were combined into a single column in order to utilize Sklearn’s text processing methods. This preprocessing resulted in a feature space where term frequency and informativity were taken into account. In further applications it would be better to handle the headlines and dates separately with the headlines classified into topics since it would provide more insight into the distribution of votes between different kinds of proposal topic classes.

Obtained 6600 datapoints from 16.12.2010 to 20.12.2020 were divided into training, testing and validation set with percentages 0.64, 0.2 and 0.16 of the total number of datapoints. The splitting has been done by utilizing Sklearn library’s train_test_split function twice. This way of doing single splitting twice and using the particular function enables datapoints to be selected into each subset randomly and therefore reduces the risk of overfitting through training the model with datapoints only from a certain period of time.

	AanestysTulosJaa	AanestysTulosEi	AanestysTulosTyhjia	AanestysTulosPoissa
AanestysTulosJaa	1.000000	-0.247961	-0.029475	-0.776770
AanestysTulosEi	-0.247961	1.000000	-0.063661	-0.409985
AanestysTulosTyhjia	-0.029475	-0.063661	1.000000	-0.029815
AanestysTulosPoissa	-0.776770	-0.409985	-0.029815	1.000000

Figure 1. Correlation matrix of the labels.

Three different machine learning models – gradient boosting for regression, linear regression and neural networks – were compared in model selection. GradientBoostingRegressor and LinearRegression together with MultiOutputRegressor were provided by Sklearn library and the neural network model by Keras library. The codes of Xavier Torres [2] and DataTechNotes [3] were modified for this particular application after the data preprocessing was conducted as noted in sections above. The gradient boosting model and linear regression model fit one regressor per label but do not take correlations between labels into account. In gradient boosting decision trees operate as weak learners: decision trees are constructed and added to the model sequentially for loss function minimization through gradient descent procedures. Each added tree further reduces the loss. In linear regression a linear model is fitted to data: a linear regression model consists of a hypothesis space where features are mapped to labels through a weight vector which determines a coefficient for each feature component resulting in a linear equation. Neural networks in turn take each feature into one input unit and add a weight to it at a neuron. The weighted combination of features or the previous layer’s outputs at each neuron are then transformed with an activation function ultimately reaching a prediction for the label. Gradient boosting has the number of boosting stages to perform as a hyperparameter example whereas the number of neurons in the first and last hidden layer and activation function type are examples of neural networks’ hyperparameters. In this application Sklearn library’s default and DataTechNotes’ model’s hyperparameters were unchanged; they will be discussed in a subsequent chapter.

The models’ error was measured using mean squared error loss which emphasizes the effect of large differences between true and predicted labels. As a convex and differentiable function easily applicable to estimate numerical label prediction errors squared error loss is the most appealing choice of loss function given this application. Squared error loss has been used for training, validating and testing in this project.

4. RESULTS

In figure 2 are presented the training and validation errors for each of the three models. The figure shows that "AanestysTulosEi" and "AanestysTulosTyhja" received slightly better validation scores using gradient boosting regression than other models. Considering how little the lowest validation errors of "AanestysTulosEi" and "AanestysTulosTyhja" differ between gradient boosting regression and neural networks looking at the models' errors on "AanestysTulosJaa" and "AanestysTulosPoissa" it's clear that overall neural networks performs best of the three models.

GRADIENT BOOSTING REGRESSION	AanestysTulosJaa	AanestysTulosEi	AanestysTulosTyhja	AanestysTulosPoissa
Training	444.666	374.215	7.877	159.054
Validation	476.381	396.360	9.067	166.397
Test	502.587	405.979	9.792	205.774

LINEAR REGRESSION	AanestysTulosJaa	AanestysTulosEi	AanestysTulosTyhja	AanestysTulosPoissa
Training	317.508	305.663	8.426	58.056
Validation	1692.827	986.269	32.904	1852.197
Test	1750.120	1242.920	44.972	2650.903

NEURAL NETWORKS	AanestysTulosJaa	AanestysTulosEi	AanestysTulosTyhja	AanestysTulosPoissa
Training	293.490	295.133	8.510	27.559
Validation	399.490	396.939	9.795	67.006
Test	408.956	372.311	10.371	88.908

Figure 2. Training, validation and test errors of the three models. The lowest validation errors are noted in red and the best performing model's test error in green.

As figure 2 shows the difference between training and validation error is rather small for gradient boosting regression whereas for linear regression the difference is rather enormous. For neural networks the difference is mediocre compared to the other two models. If the difference between training, validation and test errors of each model are compared, gradient boosting regression gets most consistent results.

The number of gradient boosting stages to perform meaning the number of trees constructed is 100 by default. Upon testing the value to be 200 and 300 it was proven that setting the hyperparameter to a larger value would have resulted in better performance although the computational complexity increased. The first hidden layer's dimension was set to 100 and the last hidden layer's to 32 whereas the rectified linear unit activation function was used in the neural network model. Upon setting the parameters to have a larger value the training error slightly increased while validation error slightly decreased. The computation, however, took much longer so the marginal improvement doesn't support using larger values for the dimensions. Also, using the sigmoid activation function instead of rectified linear unit improved validation error but made the training and testing errors larger. The activation function affects the network's ability to learn for different ranges of values [6] and the layers' dimension affects the overall performance but also model complexity. More complex the model, more computational resources are required and the risk of high variance comes to play. In contrast, a linear model can have high bias so the bias-variance tradeoff needs to be considered. Further for both models tuning the learning

rate may have impacted the models' performance. Tuning hyperparameters would have improved the models as the examples stated above let out. Also, in gradient boosting and linear regression the models fitted one regressor per label. The models could be further improved so that correlations among all four labels would be taken into account.

5. CONCLUSIONS

In this project gradient boosting regression, linear regression and neural networks were examined for multilabel prediction. From parliament proposal headlines and dates converted to numerical features the number of votes in four different categories were predicted utilizing the three examined machine learning models. The models were trained, validated and tested with 6600 datapoints each corresponding to one parliament proposal voting.

As noted in the previous section the model resulting in overall smallest validation error was neural networks. Although the chosen model for this application is neural networks, the test errors were calculated for all three models for comparison. As figure 2 shows, in the case of linear regression the typical problem of overfitting comes to play: training error is much smaller than validation and ultimately testing errors. Gradient boosting on the other hand made rather consistent predictions when comparing training, validation and test errors. This implies consistency regardless of data used; should the hyperparameters be tuned to optimal values gradient boosting may surpass neural networks in terms of model performance.

The true label values vary a lot for different datapoints which hinders prediction. In comparison with other labels "AanestysTulosTyhja" has the most same values and therefore receives the smallest loss. If more data was used for training and validation the results could have been better. As mentioned in the beginning of the report there is data available from 1998 though it might be useful to bear in mind that wording slightly changes with time. In further applications the headlines could be classified while being associated with the proposal date: this would provide insight into what kind of voting distribution is a proposal of a certain topic class at a certain time likely to get.

6. REFERENCES

List and number all bibliographical references at the end of the paper. The references can be numbered in alphabetic order or in order of appearance in the document. When referring to them in the text, type the corresponding reference number in square brackets as shown at the end of this sentence [2]. An additional final page (the fifth page, in most cases) is allowed, but must contain only references to the prior literature.

[1] Scikit-learn.org. (n.d.) *Working With Text Data — scikit-learn 0.24.1 documentation* [online]. Available at: <https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html> [Accessed 18 February 2021].

[2] Torres, X. (2020) torresxavier/Multioutput_Regression [online] GitHub. Available at: <https://github.com/torresxavier/Multioutput_Regression/blob/master/Example%20Regressor%20Chain%20-%20Multioutput%20Linear%20Regression-v1.ipynb> [Accessed 24 February 2021].

[3] DataTechNotes. (2019) Multi-output Regression Example with Keras Sequential Model [online]. Available at: <https://www.datatechnotes.com/2019/12/multi-output-regression-example-with.html> [Accessed 27 February 2021].

[4] A. Jung. (2021) Machine Learning. The Basics. [online]. Available at: <mlbook.cs.aalto.fi> [Accessed 18 February 2021].

[5] Brownlee, J. (2016) A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning [online]. Machine Learning Mastery. Available at: <https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/> [Accessed 28 February 2021].

[6] MissingLink.ai (n.d.) Hyperparameters: Optimization Methods and Real World Model Management - MissingLink.ai [online]. Available at: <https://missinglink.ai/guides/neural-network-concepts/hyperparameters-optimization-methods-and-real-world-model-management/> [Accessed 28 February 2021].