Blog Post Engagement Prediction

Purushartha Singh

April 30, 2021

Contents

1 Abstract

The goal of the project is to develop a machine learning model which can predict the engagement of a blog post based on the information about the post's current engagement, metadata, and content of the post. The indicator of engagement being used is the number of comments received for the post in the next 24 hrs. The project can potentially give more insight into the working of social media and how different features of any given post contribute to the overall engagement of a post.

2 Introduction

Internet has seen the rise of social media and other forms of user generated content over the last decade. In terms of the way social media platforms share the content, the algorithms which are utilized to distribute said content remain a closely guarded secret [1]. While social media platforms has taken over the functionality of sharing content, blogs remain the original form of sharing user generated content and have very openly accessible distribution channels.

In fact, back when this dataset was originally collected, blogs remained primarily independent of social media distribution channels and as such, can provide interesting insight into what makes a given blog post do better without over-arching algorithms to guide the engagements.

This can, in turn, give more organic insight into what would make a good social media post. Given the current levels of monetization within social media, being able to distinguish such trends to give better consistent results even with the ever changing suggestion algorithms would give a fundamental edge to this automated model.

2.1 Goals

The following would be the goals of this project:

1. Analyzing the domain and features of the multivariate dataset.

2. Recompute the baseline results from the original study.

3. Reduce the feature set size by performing feature selection by filter and wrapper methods. [2]

4. Use new regression architectures to get statistically significant improvements on the current baseline model.

5. Develop a dense Neural Network Architecture to compare to the standard regression models.

6. Explore deeper insights based on the prediction assessments and the result model.

3 Related Work

3.1 Related Work on the dataset area

There has been plenty of research work that has been done on creating regression models to predict effects on the future. For instance, Asur and Huberman used tweets from the popular website Twitter for constructing a linear regression model to predict box office revenues of movies in advance of their releases [3]. In another paper, Krebs et al. [4] were able to predict the emotional state of the user based on the comments they posted on Facebook. These indicate the clear potential for use of social media and online content to predict many aspects of the future with great success.

The current project is based on the work done by Kristian Buza [5] which predicted the number of comments a blog post received as feedback using various regression models such as the multi-layer perceptron (MLP), support vector machine (SVM), RBF-networks, regression trees (REPtree, M5P-tree), nearest neighbor models, multivariate linear regression, and bagging approaches. This study was not as thorough with the use of various models. Other such studies such as the ones by Yano and Smith [6] predicted the feedback on political blog posts by using naive bayes, linear and elastic regression, and topic-poisson models.

The baseline was selected from a set of experiments performed by Baratsas [7] where he utilized the Random forest method to get Root mean squared error scores of 23.699 and Sum of Mean squared error value of 858.742.

3.2 Related work on the Pattern Recognition Approach

Random forest classifier utilizes a large set of decision trees which collectively operate as an ensemble with each individual tree in the random forest giving it's own class prediction and the majority choice from the trees determining the choice output of class for the model[8]. A paper by Üstüner et al. [9] uses random forest methodology and compares it to other types of classification models such as Naive Bayes Classifier, Support Vector Machines, and Linear Discriminant Analysis with random forest model (RF) performing the best out of all the parametric classifiers. Another paper by Toosi et al. [10] uses Support Vector Machine (Radial and Linear), Regularized Dual Averaging (RDA), and Radom Forest Models to assessing classifications of Mangrove Trees on the coastline of Iran. RF was again found to be the best out of these models for the task.

Gradient Boosting Machines are a collection of weak learners ensemble into one model to create a strong learner. Gradient Boosting Machines can be used for both regression or classification tasks. In a study by Alonso et al [11], a comparative analysis was drawn between the random forests regression and gradient boosting regression models on wind energy prediction. The gradient boosting method was shown to be slightly better than random forests in some cases.

4 Data

4.1 Original Information

For the final project of the class, I have chosen to work on prediction of blog post popularity using the UCI repository data set of Blog Feedback[12]. This dataset was retrieved from a data crawl of Hungarian blog post websites in 2011-12 by Dr. Krisztian Buza[5].

The dataset contains 60021 data points containing 281 features. The last of these is the dependent variable and the other features are the independent variables which will be used to create a regression model to predict the number of blog post comments which a given blog will receive in the next 24 hrs. The data set is split 9:1 to get a training set of 52397 and a test set of 7624.

The features of the dataset can be roughly divided into 4 primary categories as listed in the original study:

1. **Basic features:** These constitute the metadata of the post such as the information regarding the current and past levels of engagement of the post alongside statistical data regarding the post and post author.

2. **Textual Features:** These are the bag of words representation of the 200 most descriptive words from the collective content of the dataset.

3. **Weekday features:** These are the features which go over the time and day when the post was uploaded and the day of the week the prediction is calculating for.

4. **Parent features:** Each comment/document has a parent assigned based on the trackback links contained by each blog post (website, user, section of the blog, etc.). These features contain statistical information about the parents of the post such as the mean, standard deviation, etc. for all posts by the parent.

4.2 Viability Assessment

As stated in the article by Monik Raj Behera [13] we can look at the Root Mean Square Distribution of the features to assess the viability of the feature set. Notice that since there are more than 250 features, a correlation matrix for all of them would not be very interpretable. Therefore, the variables with the highest independent correlation with the dependent variables were taken (namely feature number 5, 6, 10, 11, and 21). As seen from figure 1, the features with the highest correlation to the dependent variable are not independent. As such, feature selection can be done to reduce the number of total features and once that is done, the number of features can be significantly reduced. For the reduced set of features, (assuming it to be reduced by at least 1/10th the original features) the training set of 52397 (9:1 ratio as per the rule of 10) will be sufficient to form a regression model. Since the results have not been generated for the new model with the reduced feature set, it is not possible to determine the statistical significance of the results using the power test (as suggested in the links for the assignment) to see if a larger dataset would be needed to get statistically backed improvements.

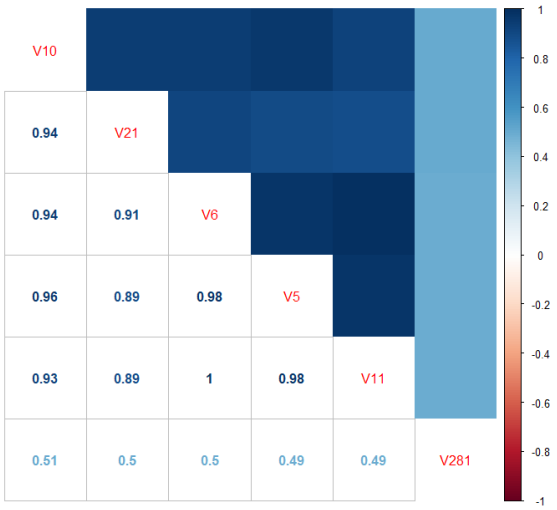


Figure 1: Correlation matrix for the highest independent correlation variables and the dependent variables

Using the inverse power law [14], we can see that the $\sqrt{60000} \approx 245$ is the sufficient number of features that can be supported by the size of the dataset. Since filter method will be reducing the dataset to 28 (top 10%) which can further be reduced by the wrapper method, we can say that the size will be sufficient for the regression task.

4.3 Data Analysis

4.3.1 Dependent Variable

The train data has 52397 observations, and the test data has 7624 observations. The dependent variable is highly skewed. 64.05% of the dependent variables in train data are zero, and among the non-zero ones, about 75% of them are less than 10, however, the dependent variable can also be as high as 1424. This can be seen in the figure 2.

4.3.2 Textual Features

Feature #63-262 represented the bag of words (BoW) representation of the top 200 most frequent words in the blog post body. The distribution of the frequency of the words follows Zipf's Law [15] which states that the frequency of occurrence of any word in a corpus is inversely proportional to its rank in the frequency table. This can be seen as the most common words in the blog post content (stop words, articles, and propositions) have counts above 10,000 while the least frequent words (Nouns and specific adjectives) appear very infrequently (less than 10) as can be seen from the distribution graph in figure 3.

The range of word frequency is 4-34k. This makes the irrelevant words which appear in all the the summaries very prevalent but poor features. To improve the feature's quality for the bag of words representation, we utilize the Term Frequency - Inverse Document Frequency (tf-idf) [16] method, which essentially takes the ratio of the number of times a

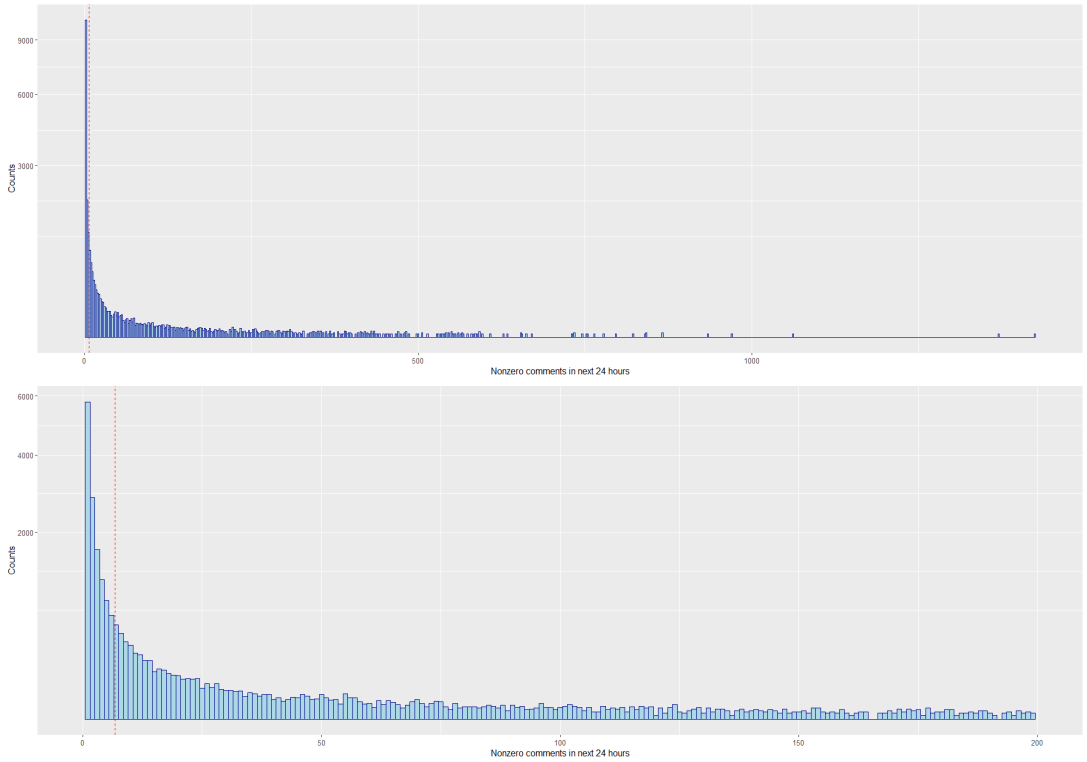


Figure 2: The distribution of the dependent variable is highly skewed in the dataset with the full data (above) and zoomed in data (below) with mean denoted by the red line

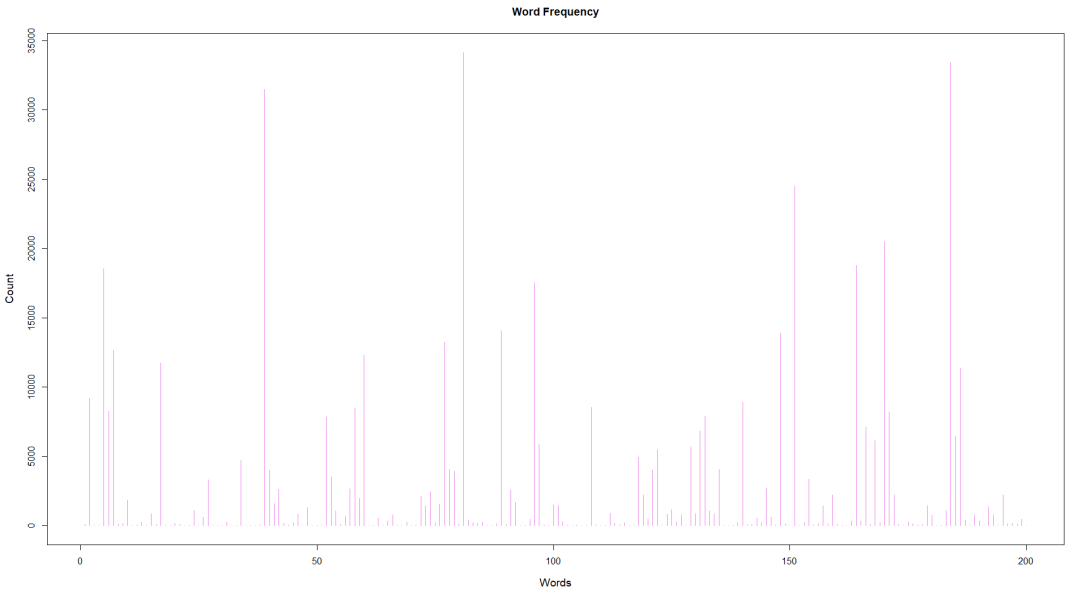


Figure 3: Word Frequency

word appears in the corpus and divides it by the number of documents the word appears in. This allows for the features to become more informative and reduces noise as this statistic increases the weight of terms that do not appear that often while reducing the importance assigned to high frequency words which appear in all documents since they

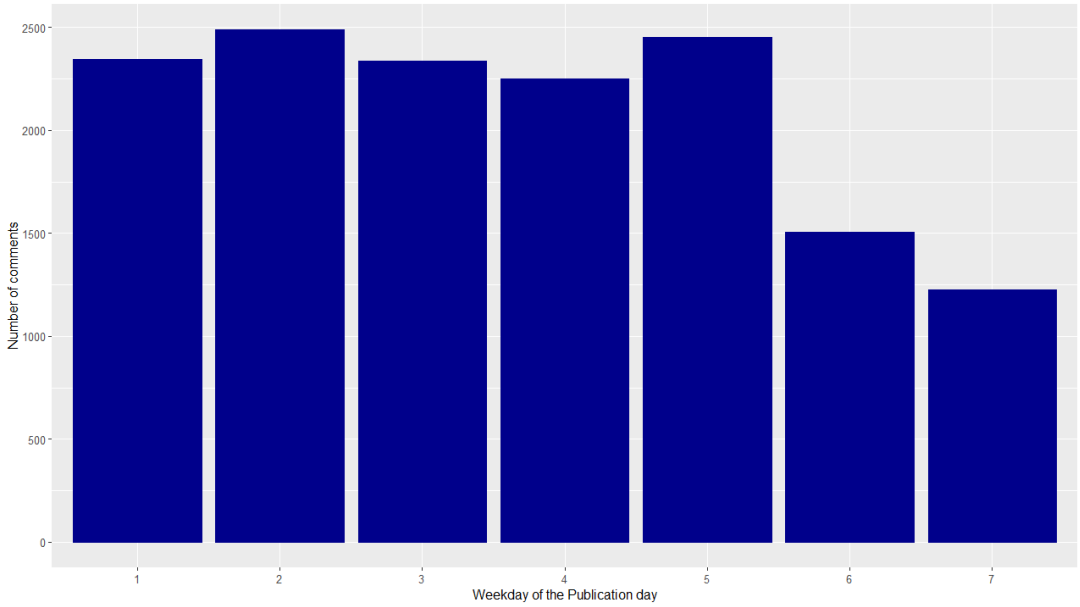


Figure 4: Day of the week and results

do not give any relevant distinguishing features about any of the documents. The modified features dramatically reduced to be between e-4 and 1547. Both the original BoW representations and the tf-idf features were not correlated with the dependent variable and as a result, the did not pass through the feature reduction steps as will be discussed in the subsequent sections of the report.

4.3.3 Day of the Week

One of the features records which day of the week the blog post is made. On taking a look at the correlation between post day and the eventual popularity of the post, it was found that the posts made of weekdays ended up being more popular compared to posts made on weekends as shown in figure 4. It was a very interesting observation since this was counter intuitive to the general assumption that posts made on the weekend would be more popular.

5 Methodology

5.1 Models

In this project, regression model of the predicted number of activity on the blog post was trained using five different regression models. Three of these methods are also a part of the original study by Buza[5] but will be expected to do better because of reduction of feature space using Feature selection. The description of each of the utilized models is provided below:

1. Linear Regression: Linear approach to modeling the relationship between a scalar response and one or more explanatory variables.

2. REP tree: Fast decision tree learner. Builds a decision/regression tree using information gain/variance and prunes it using reduced-error pruning (with back fitting).

3. M5 tree: M5P tree based on the M5 algorithm for inducing trees with regression models

4. Random forests: Ensemble learning method that is constructed by training on a number of decision trees and predicting the mean of the individual trees.

5. Gradient boost: Produces a prediction in the form of an ensemble of weak prediction models in a stage-wise fashion along with optimization of an arbitrary differentiable loss function.

6. Dense Neural Network: a dense Neural Network was trained for the regression task using the Keras API on python. The Network consisted of 4 dense layers, a dropout layer, and was trained with a learning rate of 0.001 for 20 epochs, batch size of 64 and ADAM optimizer. The visualization of the Neural Network Layers can be seen in the figure 12

5.2 Metrics

The original study [5] utilized the following custom metrics to assess the quality of the results:

1. Hit@10: Considering 10 pages with highest predicted number of feedbacks, count how many received the same in the actual test set scores.

2. AUC@10: Considering the 10 pages with highest number of feedback in reality, rank the pages according to their predicted number and area under curve.

Given the non conventional nature of these metrics, the project intends to use the more standard metrics for assessing the quality of the developed model by using the following measures as the way to compute and compare the original and new results:

1. Root mean squared error: Measures the square root of the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value.

2. Mean absolute error: Risk metric corresponding to the expected value of the absolute error loss

3. Median absolute error: Median of all absolute differences between the target and the prediction.

4. Max error: Maximum residual error, a metric that captures the worst case error between the predicted value and the true value.

5. R2 score: Represents the proportion of variance (of the dependent variable) that has been explained by the independent variables in the model. It provides an indication of goodness of fit and therefore a measure of how well unseen samples are likely to be predicted by the model, through the proportion of explained variance.[17]

It must be noted that most of the results were consistent throughout all the metrics and MSE and R2 were used as the primary metrics for the purposes of visualization and discussion since these two provided the most insight in terms of the interactions of all the models and methodologies.

6 Approach

The implementation of the code was done using Python and R. Environment of Sublime Text and R Studio was utilized with separate scripts for feature selection, data visualiztion, principle component analysis and the neural network model.

For the baseline, the original data was taken as input and the scikit library for python [17] was used to execute the 4 regression models on the unprocessed data. The results were calculated using the same library. These results were stored in an excel spreadsheet. This process was repeated by shuffling the training data randomly to get a representative value for all the metrics used to assess the results.

The data was analyzed using the R script and this was followed by feature selection, and Principle Component Analysis. The script outputted specific column indices as the output which was used to select features from the original data. Additionally, the scikit library [17] was used to produce new set of 20 features using PCA. These features were stored alongside with the output for both the training and testing data in separate csv files which were then run with the same regression script to get outputs for the PCA steps. Similarly, specific features were used to create models for the feature selection section to get results for that section of the project.

The last step was the construction of the Dense Neural Network using the keras API for Python alongside the TensorFlow API. The results for the Neural Network were also calculated and added to the spreadsheet. All this information was compiled for the presentation and report.

7 Timeline

This section illustrates the project timeline as was shared in the proposal alongside the changes made to the timeline during the course of the project.

Table 1: Weekly expected timetable for the project

Expected Timeline	Information
Week 1 (3/24–3/30):	Initial Preprocessing Working on the Project Proposal
Week 2 (3/31–4/6):	Implement Baseline Model
Week 3 (4/7–4/13):	Implementation of additional Regression models

Continued on next page

Table 1: Weekly timetable for the project. This second page has been split automatically.
– continued from previous page

Week	Info
Week 4 (4/14–4/20):	Implementation of PCA
Week 5 (4/21–4/27):	Implementation of DNN model
Week 6 (4/28–5/3):	Final Report & Presentation

Table 2: Weekly expected timetable for the project

Expected Timeline	Information
Week 1 (3/24–3/30):	**Initial Preprocessing** **Working on the Project Proposal**
Week 2 (3/31–4/6):	Implement Baseline Model
Week 3 (4/7–4/13):	**Implementation of additional Regression models** **Implementation of PCA**
Week 4 (4/14–4/20):	**Feature selection** **DNN research**
Week 5 (**4/21–4/25**):	Implementation of DNN model
Week 6 (**4/26–4/30**):	Final Report & Presentation

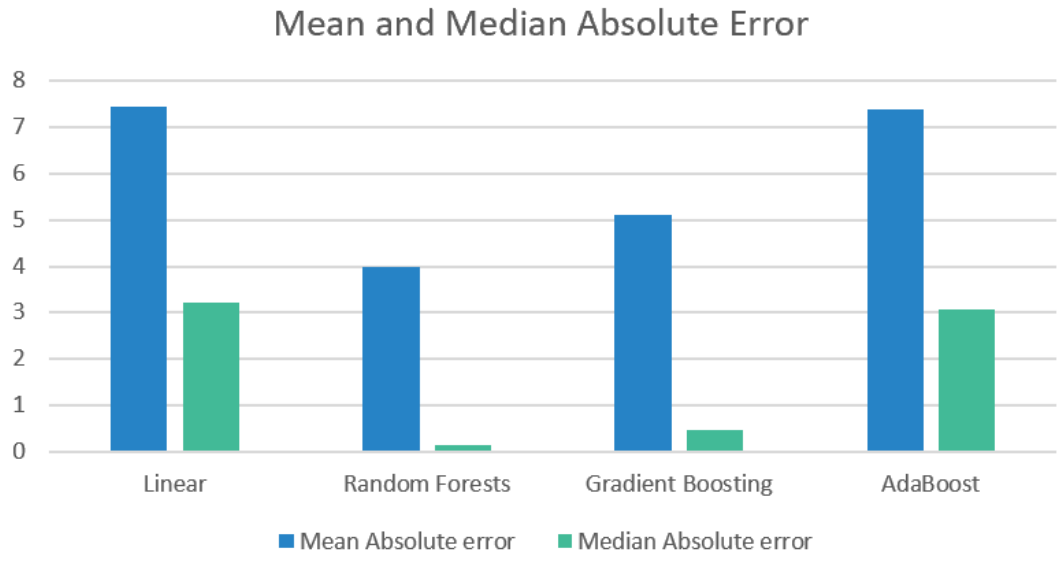
The timeline remained mostly consistent throughout the project. The only major changes made were the fast tracking of PCA sections of the project as per the feedback received during the mid project submission. This gave more time to perform feature selection.

It must also be noted that the original timeline was not accurate as the deadline was expected to be the finals week. Therefore, the amount of time spent on the optimization of the Neural Network was not sufficient. This was done to meet the deadline expectation, specially since the final report and presentation would need more than 2 days to finish.

8 Results

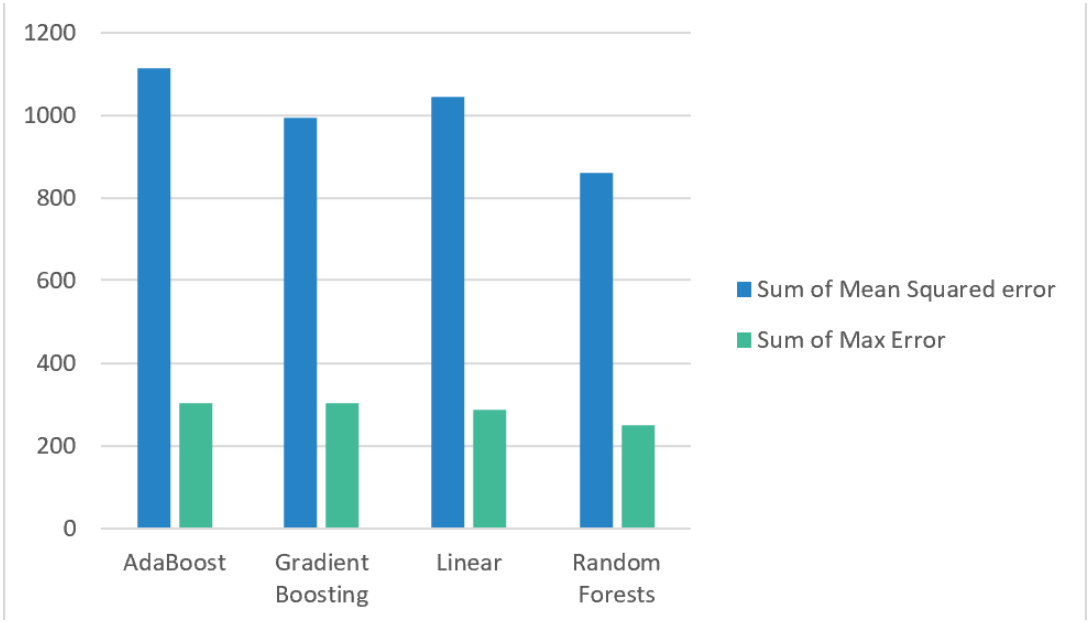
8.1 Baseline

In the first part of the project, regression was performed on the complete feature dataset using 4 regression models discussed above: Linear regression, Random forests, Gradient boost and Adaptive Boost for baseline results. The evaluation criteria for this purpose was mean squared error, mean absolute error, median absolute error, max error and R2 score. The default models from Sci-kit learn were used for this implementation [17]. The results can be seen in the figure 5 and 6 below for the initial additional metrics. The results for the two shortlisted metrics can be seen in figure 7 and 8 As these



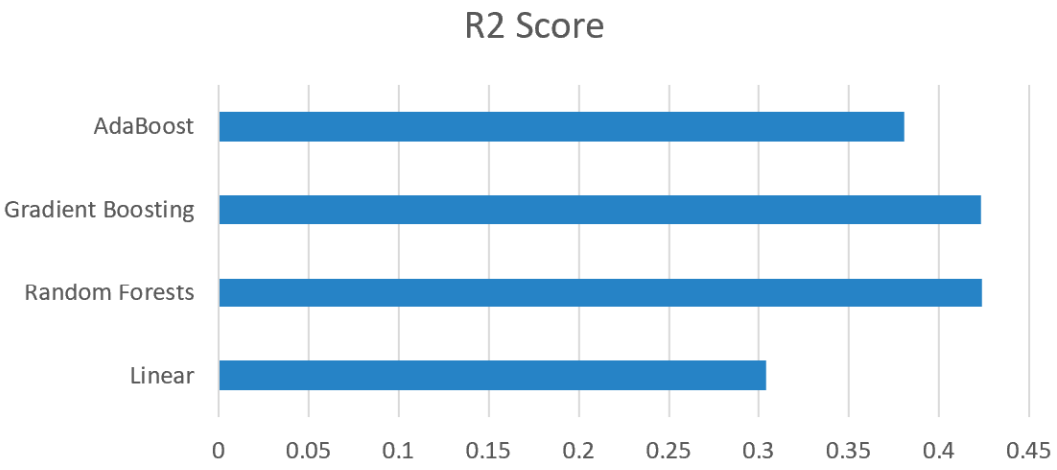
	Linear	Random Forests	Gradient Boosting	AdaBoost
Mean Absolute error	7.423905375	3.99591641	5.108939541	7.364177695
Median Absolute error	3.188733053	0.15	0.465847125	3.055949026

Figure 5: Mean absolute error and Median Absolute Error for the Baseline results with histogram visualization (above) and table of values (below)



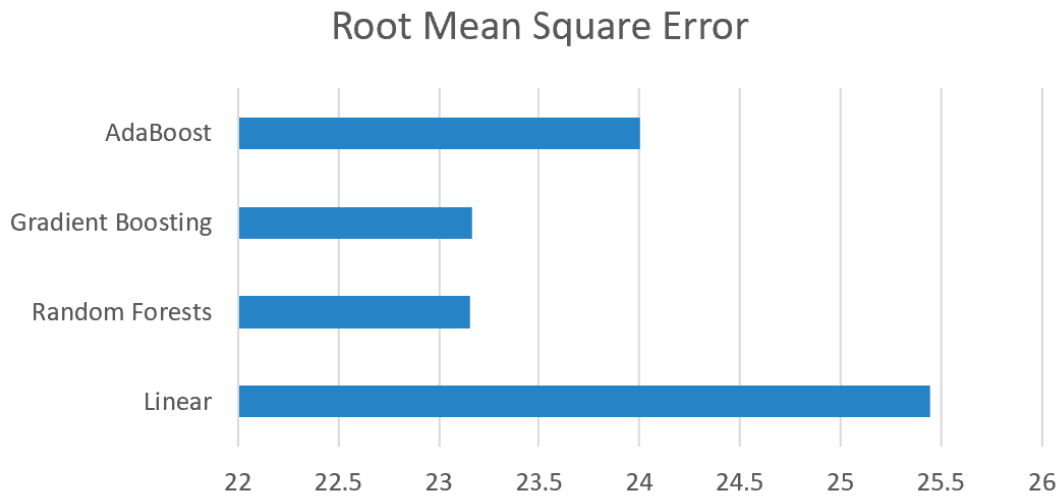
	Linear	Random Forests	Gradient Boosting	AdaBoost
Mean Squared error	292.2153626	207.9403101	390.5263124	305.416858
Max Error	109.002758	119.8	167.1890085	118.6735484

Figure 6: Sum of Mean Squared Error and Max error for the Baseline results with histogram visualization (above) and table of values (below)



	Linear	Random Forests	Gradient Boosting	AdaBoost
R2 Score	0.304099028	0.42370161	0.423305863	0.380625435

Figure 7: R2 metric for the Baseline results with histogram visualization (above) and table of values (below)



	Linear	Random Forests	Gradient Boosting	AdaBoost
RMSE	25.44321513	23.15378368	23.16173223	24.00352102

Figure 8: Root Mean Squared Error for the Baseline results with histogram visualization (above) and table of values (below)

	Baseline						
Regression models	**Mean Absolute error**	**Median Absolute error**	**Mean Squared error**	**Max Error**	**R2 Score**	**RMSE**	**STD**
Linear	8.411848261	3.761166439	647.3571962	527.3234202	0.304099028	25.44321513	25.44308718
Random Forests	5.176535441	0.34	536.0976988	416.38	0.42370161	23.15378368	23.15285466
Gradient Boosting	5.182563545	0.568585251	536.46584	511.2151051	0.423305863	23.16173223	23.16044409
AdaBoost	8.537843187	5.153703445	576.1690211	517.5743029	0.380625435	24.00352102	23.79614988

Figure 9: Table of all the results performed on the baseline

results indicate, the best performance for the baseline is achieved using the random forest classifier with the Gradient Boosting Algorithm being a close second. All the future results will be collectively compared to the baseline in the subsequent sections. The complete table of all the results for all the metrics used for the baseline experiments alongside the standard deviations and means is provided in the figure 9. It must be noted that Standard Deviation is high due to the highly skewed nature of the original data.

8.2 Principle Component Analysis

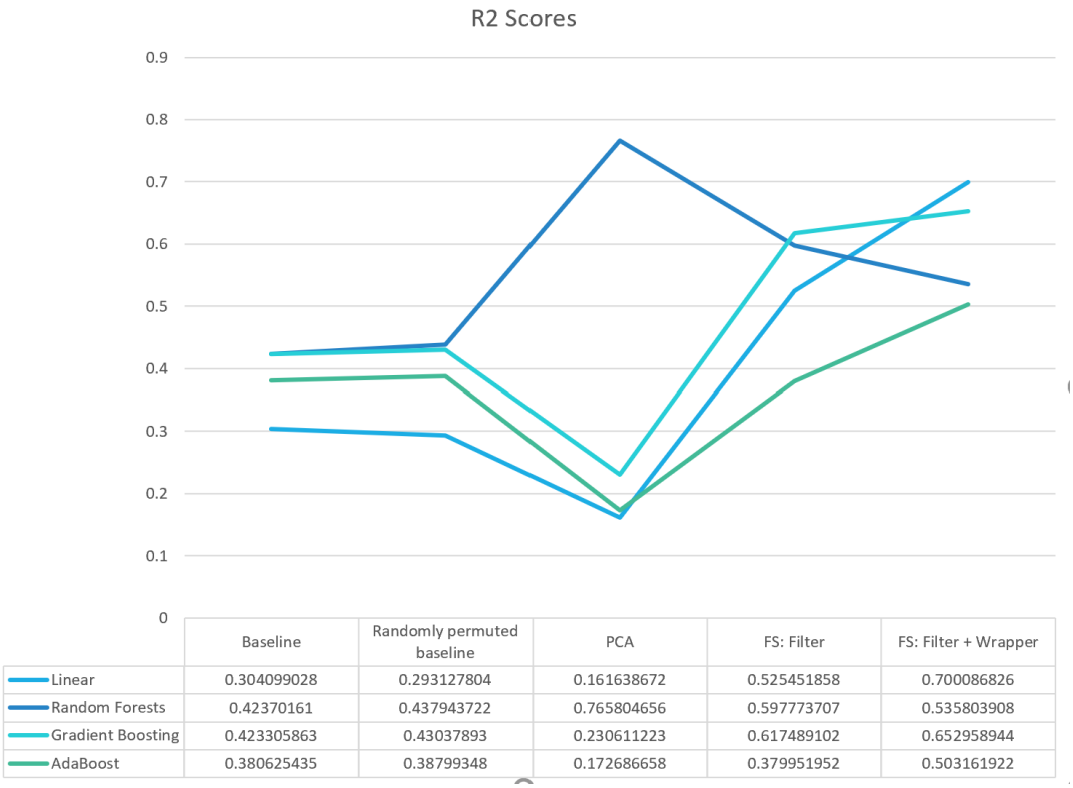
The results for Principle Component Analysis were very erratic. It was decided to use PCA reduction to 28 dimensions as the primary PCA model as it produced best results. A lot of the other PCA reductions produced negative R2 values indicating that the results were not consistent with the data and were discarded. Overall, the performance reduced drastically over all the methods with the exception of Random Forest Method which, in fact, had the best results out of all the experiments. All other models performed worse than baseline using the PCA. The results for PCA have been shown in the table in figure 10 and 11 alongside the results of all other methodologies.

8.3 Feature Selection

Top 10% features were selected based on the F-Regression methodology which looked at the ANOVA scores of each of the features and compared them to the results columns, selecting the top 28 ranked features. Out of these, further 10 were selected using iterative elimination method using random forest as the guiding algorithm. The following were the key features that were used for the new set of experiments:

- Statistics about the average, standard deviation, min, max, and median of the source of the post
- Total number of comments before the observation of next 24 hours of comments
- Comments in the last 24 hours
- Number of comments in the first 24 hours after the post was published
- Length of time between the publication of the post and observation
- Length of the blog post

The results of the feature selection were very positive. There was a general improvement for all the models compared to baseline results for the R2 metric while all performed marginally worse than baseline for the RMSE metric. The best performing models were



	Baseline	Randomly permuted baseline	PCA	FS: Filter	FS: Filter + Wrapper
Linear	0.304099028	0.293127804	0.161638672	0.525451858	0.700086826
Random Forests	0.42370161	0.437943722	0.765804656	0.597773707	0.535803908
Gradient Boosting	0.423305863	0.43037893	0.230611223	0.617489102	0.652958944
AdaBoost	0.380625435	0.38799348	0.172686658	0.379951952	0.503161922

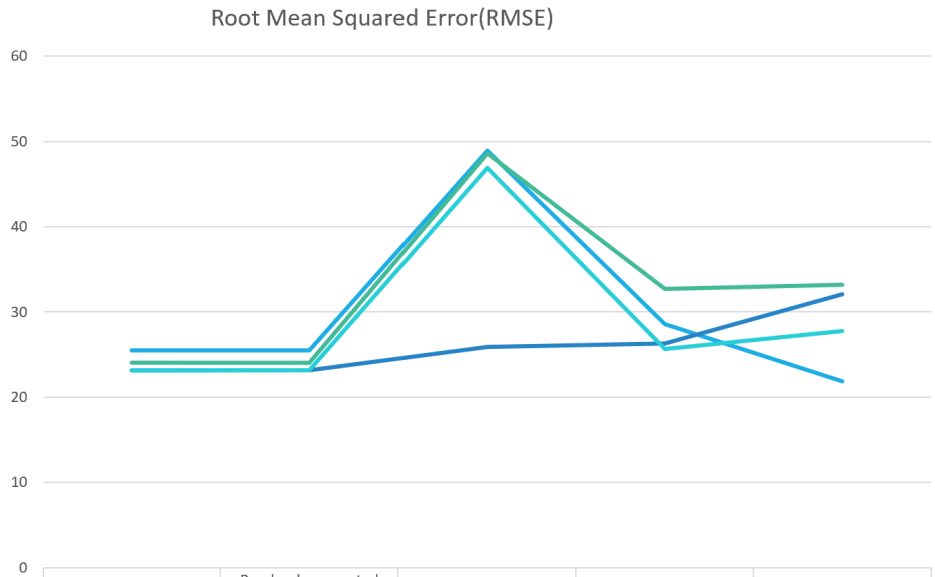
Figure 10: R2 metric for the all the results results with visualization (above) and table of values (below)

linear regression using Filter+Wrapper methods. This is consistent with the expectation since the reduction in the number of features is expected to increase the accuracy of the linear model. The only model with worse accuracy using the wrapper method was the Random Forest model. For just the filter method, Gradient Boosting had the bet performance. The results are shared in figure 10 and 11 alongside the results of all other methodologies.

8.4 Neural Network

The Neural Network utilized was a sequential Dense Neural Network with 4 Dense layers and one dropout layer. The shape and sequence of the layers is displayed in the figure 12. The network had used a **batch size of 100** with **10 epochs** and **validation split of 0.1**. The ADAM optimizer was used and RMSE was used as the loss function. The **learning rate** was tuned to **0.001**. The activation function for each layer was ReLu since the regression cannot be negative. The entirety of 280 features was passed in as the input.

Initally, the network output was poor and the model was outperforming baseline but not some of the other methodologies. This was due to overfitting of the data. Modifications to the learning rate, batch size, and number of epochs was made to improve the model and get the best accuracy of the entire project by a fair margin. The runtime summary of the epochs for the final run is shown in figure 13 and the comparison of the R2 and RMSE metric for the neural network compared to other models best performing



	Baseline	Randomly permuted baseline	PCA	FS: Filter	FS: Filter + Wrapper
Linear	25.44321513	25.46480191	48.95339128	28.59294648	21.89102938
Random Forest	23.15378368	23.1102777	25.87356625	26.32412843	32.0626919
Gradient Boosting	23.16173223	23.15984236	46.89646294	25.67087543	27.72296454
Adaboost	24.00352102	24.06206049	48.6297659	32.68371729	33.17085595

Figure 11: Root Mean Squared Error for all the results with visualization (above) and table of values (below)

Model: “Sequential”

Layer (type)	Output Shape	Param #
Dense (Dense)	(None, 128)	36096
Dense (Dense)	(None, 64)	8256
Dropout	(None, 64)	0
Dense (Dense)	(None, 32)	2080
Dense (Dense)	(None, 1)	33

Total params: 46,465

Trainable params: 46,465

Non-trainable params: 0

Figure 12: Architecture of the Neural Network Layers

```
Epoch 1/10
472/472 [==============================] - 27s 5ms/step - loss: 8543.9346 - val_loss: 50.9983
Epoch 2/10
472/472 [==============================] - 2s 5ms/step - loss: 161.2144 - val_loss: 12.6052
Epoch 3/10
472/472 [==============================] - 2s 5ms/step - loss: 75.5639 - val_loss: 6.9807
Epoch 4/10
472/472 [==============================] - 2s 5ms/step - loss: 70.1905 - val_loss: 13.4106
Epoch 5/10
472/472 [==============================] - 2s 5ms/step - loss: 53.0967 - val_loss: 9.6243
Epoch 6/10
472/472 [==============================] - 3s 5ms/step - loss: 30.4918 - val_loss: 11.5390
Epoch 7/10
472/472 [==============================] - 3s 5ms/step - loss: 29.8845 - val_loss: 5.5024
Epoch 8/10
472/472 [==============================] - 2s 5ms/step - loss: 30.7917 - val_loss: 6.1216
Epoch 9/10
472/472 [==============================] - 3s 6ms/step - loss: 21.6838 - val_loss: 4.7662
Epoch 10/10
472/472 [==============================] - 2s 3ms/step - loss: 29.1210 - val_loss: 4.4862
239/239 [==============================] - 0s 834us/step - loss: 15.5782
```

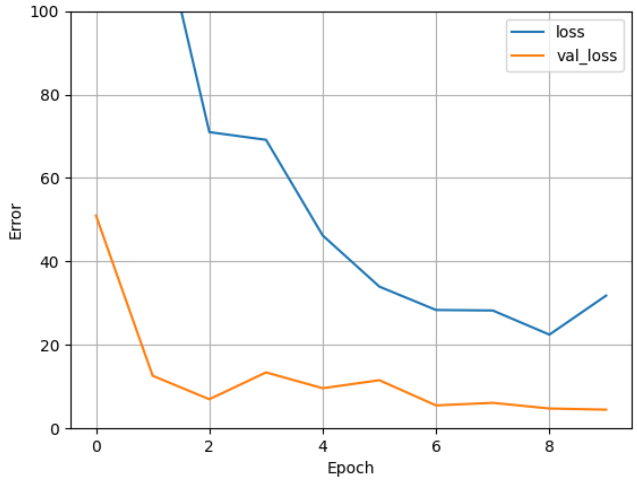


Figure 13: Detailed training information of the training of the Neural Network

scores is shown in figure 14

9 Discussion

Given the dimensionality of the original data, it was quite fascinating to observe the boost to performance gained through the use of feature selection. Almost all models showed improvement after using the filter and wrapper methods over the baseline. The most prominent result was the improvement to the Linear Regression model. Given the simple nature of the algorithm, the reduction in the number of dimensions greatly improved the results over the baseline and with just features, it even outperformed the more complex models. This indicated that important features were linearly correlated with the dependent variables and the model did not need excessively complex architectures to solve it

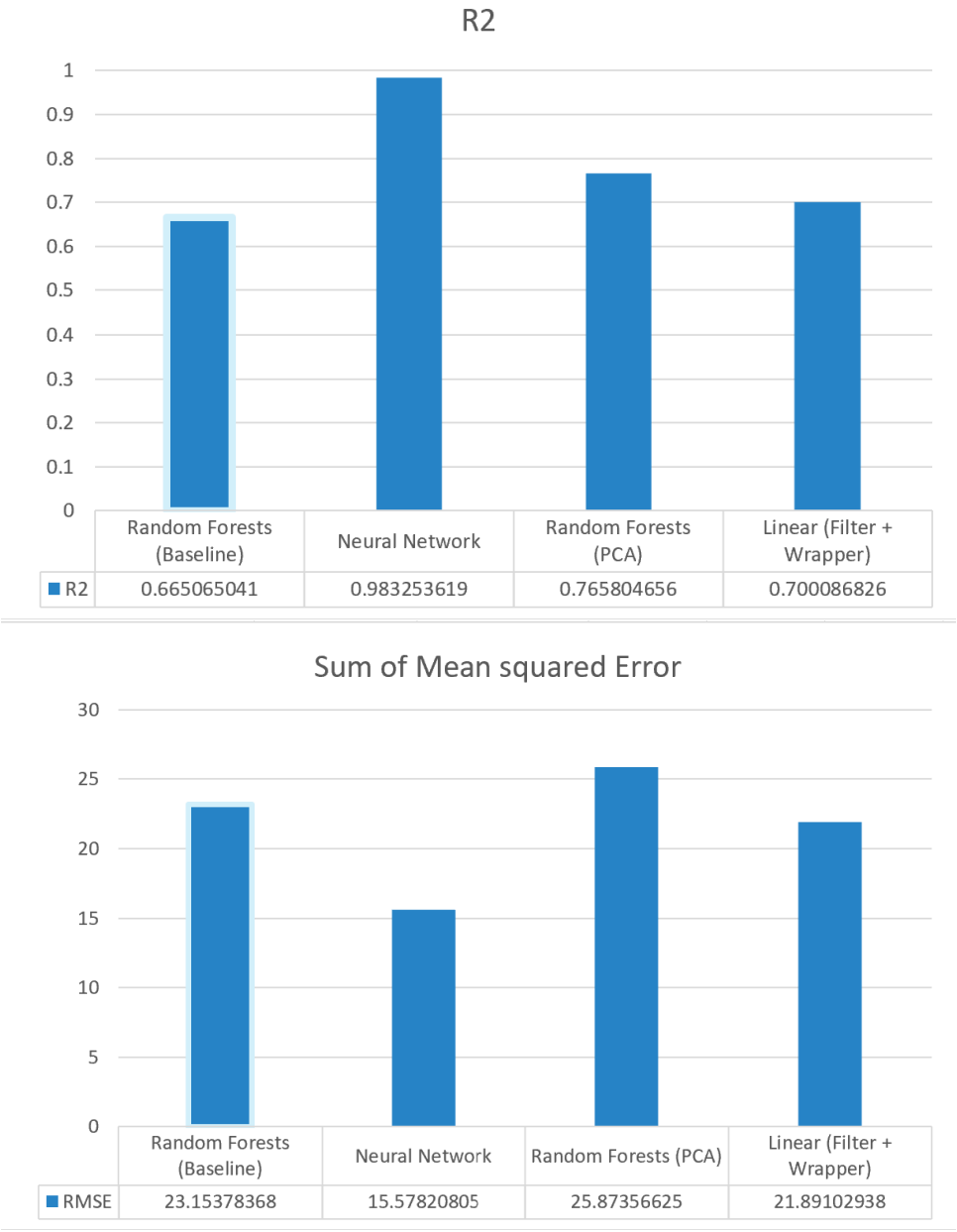


Figure 14: The R2 and RMSE performance comparison of the NN with the baseline and other best performing models

after the feature selection.

A big takeaway was the use of different metrics to assess the performance of the models. While most of them had similar assessments of the models, R2 was able to provide different insights to other models when looking at the scores of PCA. The negative values strongly indicated poor performance despite acceptable RMSE results. This was the primary reason why it was kept as one of the two main metrics used for all the discussion and results out of the 5 mentioned in the beginning of the report.

It was also very interesting to see the strengths and weaknesses of the different regression models. Given the ensemble nature of Random Forest model, it performed the best with the PCA features since it was able to distinguish the non linear patterns that were encoded within those dimensions in terms of regression. All other models performed poorly on this metric which indicates that PCA often causes data to be redistributed in ways which are hard to interpret linearly.

The Neural Network performed the best but needed a large amount of optimizations to execute to its ability. Given the boost in performance compared to the traditional and ensemble models, it is not surprising to see Neural Networks be preferred for most tasks. It must be noted that the model needed a lot more tuning and knowledge of model creation to perform well. The trade-offs were certainly worth the decreased RMSE scores ($\approx 40\%$) and increased R2 scores (≈ 0.2).

10 Conclusion

The project was an amazing opportunity to work on a real life dataset and utilize the various methodologies that were picked up during the course of the semester. It was fascinating to see the improvement of results and debugging the issues in the code using the resources provided while simultaneously learning new things such as implementation in python and utilization of new different models not covered in class in conjunction with the discussed regression models. The comparison of these models was also able to help me identify the specific strengths and weaknesses of each of these approaches.

In terms of the data set, gaining insight into the various ways in which online interactions works was very rewarding. The notion of 'popular' is very subjective, but turning that into an 'objective' dependent variable and finding a way to link it to various pieces of information we had about each post allowed for a great in depth analysis of what really matters when we are looking for engagement in today's world where such information has monetary value.

Perhaps the most interesting thing to find was that what was said in the post had very mild correlation with the engagement and the size and timing of the post was a lot more relevant to the prediction. It was also interesting to observe the snowball effect where a post which was generating a lot of buzz had a lot more chance to get higher values for engagement later on too. Such insights and empirical evidence for the same through the

experiment were very rewarding.

In terms of ways to improve upon this experiment and potential future work, it would be very difficult to improve on the scores achieved by the neural network using any other classical or ensemble model. In fact, the accuracy of the Neural Network highlights the non linear distribution of the data which Neural Networks handle well.

References

[1] C. Figueiredo and C. Bolaño, "Social media and algorithms: Configurations of the lifeworld colonization by new media," *The International Review of Information Ethics*, vol. 26, Dec. 2017. [Online]. Available: http://informationethics.ca/index.php/irie/article/view/277

[2] C. M. Bishop, *Pattern recognition and machine learning, 5th Edition*, ser. Information science and statistics. Springer, 2007. [Online]. Available: https://www.worldcat.org/oclc/71008143

[3] S. Asur and B. A. Huberman, "Predicting the future with social media," *CoRR*, vol. abs/1003.5699, 2010. [Online]. Available: http://arxiv.org/abs/1003.5699

[4] F. Krebs, B. Lubascher, T. Moers, P. Schaap, and G. Spanakis, "Social emotion mining techniques for facebook posts reaction prediction," *CoRR*, vol. abs/1712.03249, 2017. [Online]. Available: http://arxiv.org/abs/1712.03249

[5] K. Buza, "Feedback prediction for blogs," *Data Analysis, Machine Learning and Knowledge Discovery*, pp. 145–152, 10 2014.

[6] T. Yano, W. W. Cohen, and N. A. Smith, "Predicting response to political blog posts with topic models," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, ser. NAACL '09. USA: Association for Computational Linguistics, 2009, p. 477–485.

[7] S. Baratsas, "Predicting blog comments with machine learning methods." [Online]. Available: https://www.baratsas.com/blog-comments-prediction

[8] Wikipedia - random forest classifier. [Online]. Available: https://en.wikipedia.org/wiki/Random_forest

[9] M. Üstüner, Gökdağ, G. Bilgin, and F. B. Şanlı, "Comparing the classification performances of supervised classifiers with balanced and imbalanced sar data sets," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, 2018, pp. 1–4.

[10] N. B. Toosi, A. R. Soffianian, S. Fakheran, S. Pourmanafi, C. Ginzler, and L. T. Waser, "Comparing different classification algorithms for monitoring mangrove cover changes in southern iran," *Global Ecology and Conservation*, vol. 19, p. e00662, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2351989419300617

[11] Á. Alonso, A. Torres, and J. R. Dorronsoro, "Random forests and gradient boosting for wind energy prediction," in *Hybrid Artificial Intelligent Systems*, E. Onieva, I. Santos, E. Osaba, H. Quintián, and E. Corchado, Eds. Cham: Springer International Publishing, 2015, pp. 26–37.

[12] K. Buza. Uci: Machine learning repository: Blog post feedback dataset. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/BlogFeedback#

[13] M. R. Behera. Towards data science - how 'big' should it be. [Online]. Available: https://towardsdatascience.com/how-big-should-be-your-data-fdace6e627e4

[14] Wikipedia - power law. [Online]. Available: https://en.wikipedia.org/wiki/Power_law

[15]

[16] "Advances in computing and network communications," *Lecture Notes in Electrical Engineering*, 2021. [Online]. Available: http://dx.doi.org/10.1007/978-981-33-6977-1

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.