Predicting popularity of Reddit posts

Submitted To

Prof. Shahriar Asta

Submitted By

Gautam Dawar

Table of Contents

Introduction and data insights*

Reddit is a social website to share content related to a plethora of topics called "posts". The readers also have and option to "upvote" or "downvote" a post which is an indicator of the post's popularity. While the popularity of the post depends on its content, however there are a variety of other features which can influence the popularity of the post. The purpose of the project is to predict the total votes received by a post or its "popularity" based on a number of features about the post such as its title, time of post, author etc. The dataset has the posts data and related features from year 2006 and has target column "score" with whole number scores ranging from 0 to 583 and 58 feature columns having both numerical and categorical features. The data points exceed 12,500 in the dataset.

Project Outline*

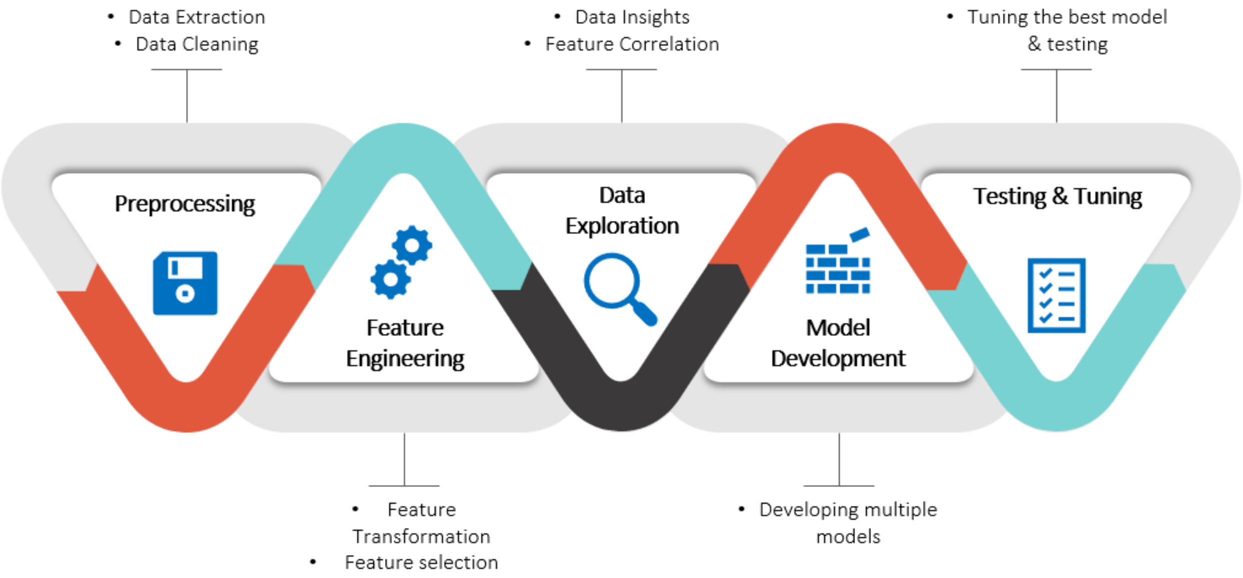


Figure 1: Project Outline

Fig.1 shows the project outline to predict the reddit posts. All the stages of the project are described in the upcoming sections as follows –

1. **Data Preprocessing** – Cleaning the data from null or missing values and removing sparse features
2. **Feature Engineering** – Generating new features from the existing features to help our model learn
3. **Data Exploration** – Exploring the dataset for statistical insights and dependencies of features to target

4. **Model Development** – Deploying multiple machine learning models and comparing their performance using R2 and RMSE evaluation metrics
5. **Tuning and Testing** – The best model is then chosen and tested in the OOT (out of time) dataset so verify if the model has generalized enough to predict scores of completely unseen reddit posts data.

1 Data Preprocessing*

The data was full of null values and contained features filled entirely with values such as 'None' or '[deleted]'. Hence these columns were then removed by implementing a code which skimmed through all the features in search for features conataining only empty strings or null values.

After that, columns with sparse values were removed as well. Features having sparse values, i.e. a feature is considered sparse if it contains more than 80% null values. A code block was implemented (see appendix) which identified such columns. Please note, the 80% threshold of sparsity was set by us after experimenting with multiple different values.

Then statistically insignificant features were identified, that is features having categorical data but containing more than 90% of a single category. Such feature could not be used to help the model learn and hence such features were dropped. With this featues were shortlisted which would not help our machine learning model learn, and hence they were dropped from the dataframe, leaving us with 13 features.

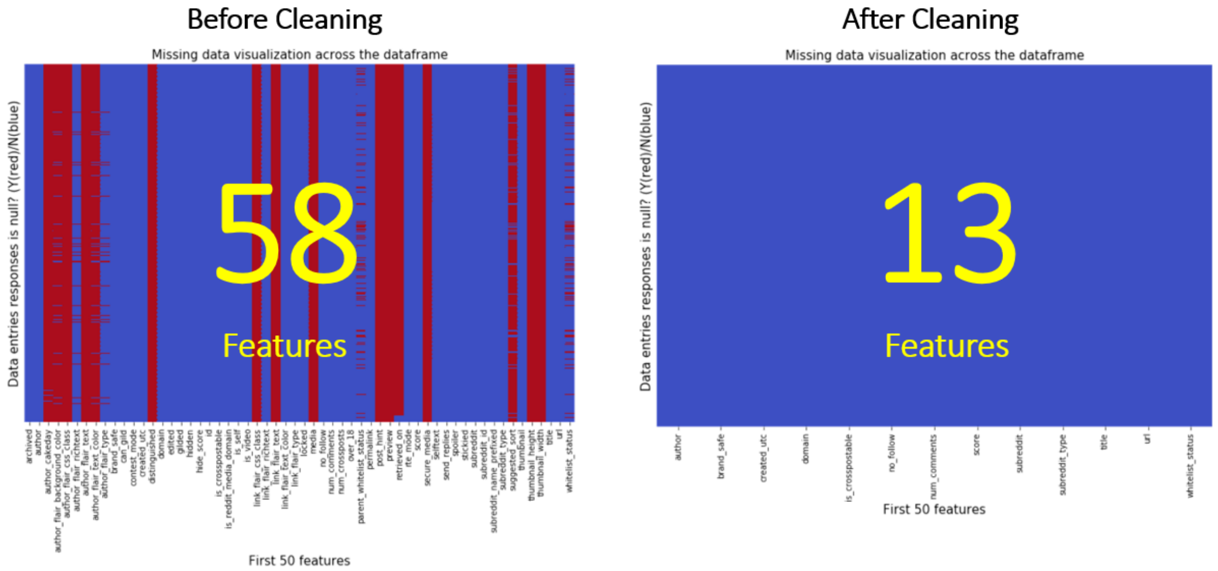
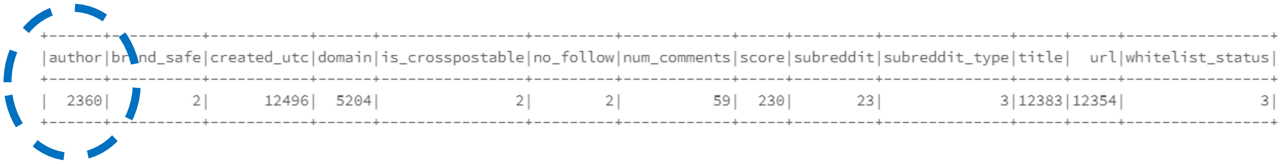


Figure 2 Heat map of dataframe before and after cleaning. Red means null values

2 Feature Engineering

With the 13 original features selected from the section above, new features were engineered to help the machine learning model train well with increased variable information. The features were engineering from the following original features –

2.1 Author Feature*

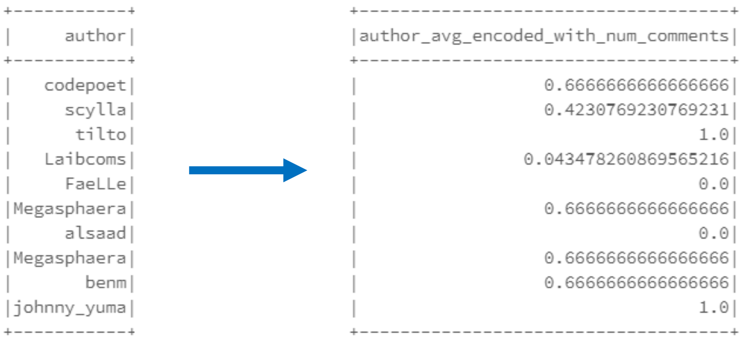


```
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|author|brand_safe|created_utc|domain|is_crosspostable|no_follow|num_comments|score|subreddit|subreddit_type|title|  url|whitelist_status|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|  2360|         2|      12496|  5204|               2|        2|          59|  230|       23|             3|12383|12354|               3|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
```

Figure 3 The distinct value counts of the 13 original selected features from the dataset

The author feature had 2360 distinct values hence onehot encoding would have resulted in a sparse matrix and label encoding would give unfair weights to underrepresented authors. So, to tackle this problem, an innovative approach was opted to label the authors. The original feature of the 13 selected 'num_comments' was used to get the average number of comments received by each author using aggregate functions.

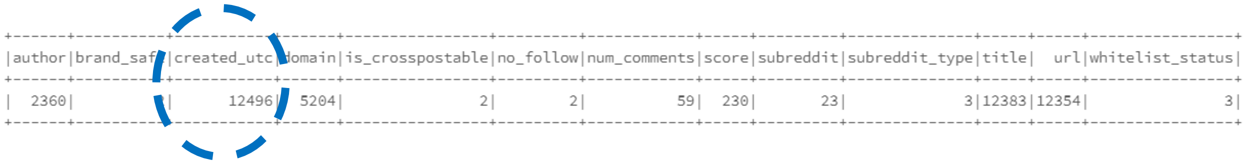
Window function was used to calculate the average. The average comments obtained for each author were then utilized as their respective labels and a new feature was engineered.



```
+-----------+          +-----------------------------------+
|     author|          |author_avg_encoded_with_num_comments|
+-----------+          +-----------------------------------+
|   codepoet|          |                 0.6666666666666666|
|     scylla|          |                 0.4230769230769231|
|      tilto|          |                                1.0|
|   Laibcoms|          |               0.043478260869565216|
|      FaeLLe|          |                                0.0|
|Megasphaera|          |                 0.6666666666666666|
|     alsaad|          |                                0.0|
|Megasphaera|          |                 0.6666666666666666|
|       benm|          |                 0.6666666666666666|
|johnny_yuma|          |                                1.0|
+-----------+          +-----------------------------------+
```

Figure 4 Example of new feature created on author

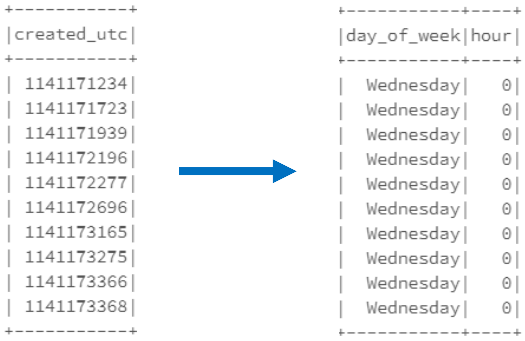
2.2 Created_utc Feature



```
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|author|brand_safe|created_utc|domain|is_crosspostable|no_follow|num_comments|score|subreddit|subreddit_type|title|  url|whitelist_status|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|  2360|         2|      12496|  5204|               2|        2|          59|  230|       23|             3|12383|12354|               3|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
```

Figure 5 The distinct value counts of the 13 original selected features from the dataset

The created_utc feature refers to the time stamp for the reddit posts. The pyspark function ‘to_timestamp’ was used to create a readable instance of the date and subsequently the day of the week and hour of posting was extracted to be used as feature columns. Below is the example of the feature columns obtained afterwards.

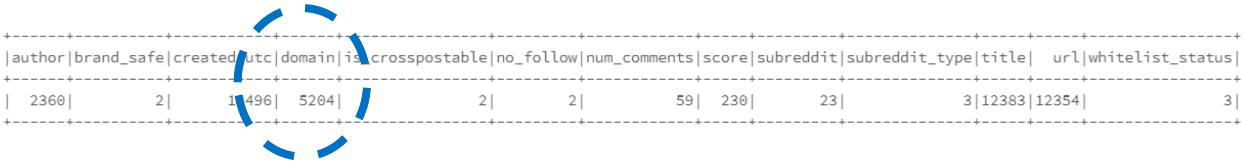


```
+-----------+
|created_utc|
+-----------+
| 1141171234|
| 1141171723|
| 1141171939|
| 1141172196|
| 1141172277|
| 1141172696|
| 1141173165|
| 1141173275|
| 1141173366|
| 1141173368|
+-----------+
```

```
+-----------+----+
|day_of_week|hour|
+-----------+----+
|  Wednesday|   0|
|  Wednesday|   0|
|  Wednesday|   0|
|  Wednesday|   0|
|  Wednesday|   0|
|  Wednesday|   0|
|  Wednesday|   0|
|  Wednesday|   0|
|  Wednesday|   0|
|  Wednesday|   0|
+-----------+----+
```

Figure 6 Example of new feature created on created_utc

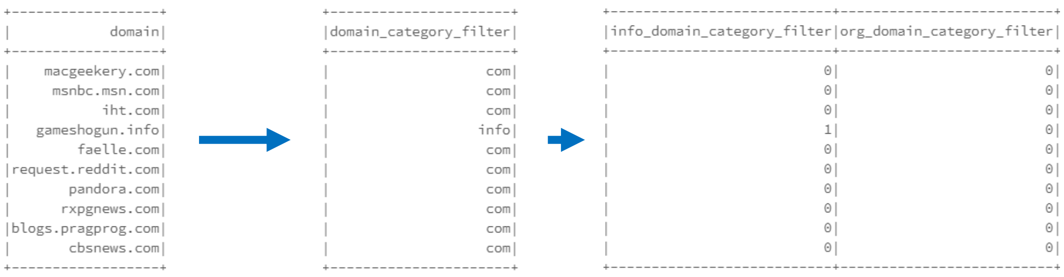
2.3 Domain Feature*



```
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+---------------+
|author|brand_safe|created_utc|domain|is_crosspostable|no_follow|num_comments|score|subreddit|subreddit_type|title|  url|whitelist_status|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+---------------+
|  2360|         2|     1[illegible]496|  5204|               2|        2|          59|  230|       23|             3|12383|12354|              3|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+---------------+
```

Figure 7 The distinct value counts of the 13 original selected features from the dataset

The domain feature had the domain name the posts belonged to. Hence to created a fewer categories for or machine learning model to learn from, we planned to reduce the original 5204 categories to a mere 90 by extracting the domain extension from the string such as ‘.com’ or ‘.edu’. But even in this we had many categories which were under-represented, or had a count frequency of less than 1% of the entire dataset (100 counts). After that one hot encoding was performed on the remaining 9 categories. Hence, we made a filter to club all such categories into one. Hence the example of the categories is shown below.



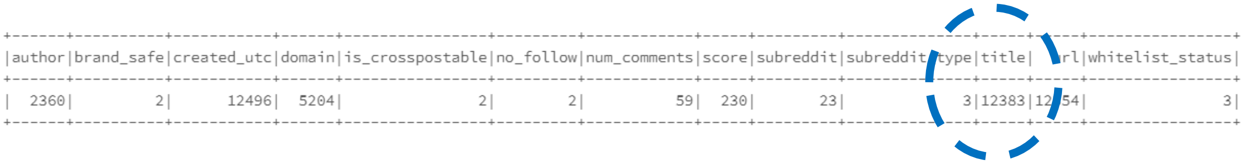
```
+------------------+
|            domain|
+------------------+
|     macgeekery.com|
|      msnbc.msn.com|
|            iht.com|
|     gameshogun.info|
|          faelle.com|
|request.reddit.com|
|         pandora.com|
|         rxpgnews.com|
|blogs.pragprog.com|
|          cbsnews.com|
+------------------+
```

```
+----------------------+
|domain_category_filter|
+----------------------+
|                   com|
|                   com|
|                   com|
|                  info|
|                   com|
|                   com|
|                   com|
|                   com|
|                   com|
|                   com|
+----------------------+
```

```
+---------------------------+--------------------------+
|info_domain_category_filter|org_domain_category_filter|
+---------------------------+--------------------------+
|                          0|                         0|
|                          0|                         0|
|                          0|                         0|
|                          1|                         0|
|                          0|                         0|
|                          0|                         0|
|                          0|                         0|
|                          0|                         0|
|                          0|                         0|
|                          0|                         0|
+---------------------------+--------------------------+
```

Figure 8 Example of new feature created on domain

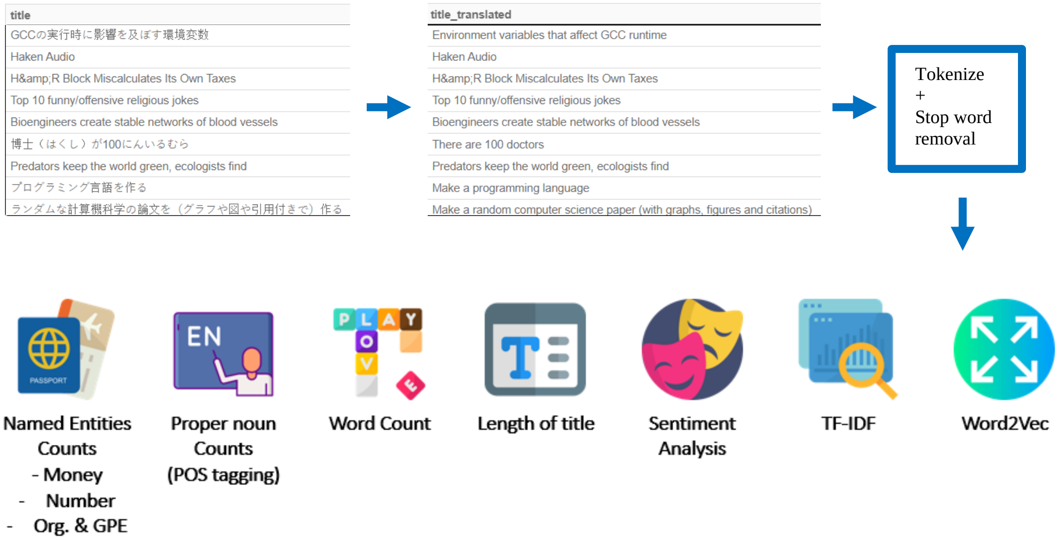
2.4 Title Feature*



```
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|author|brand_safe|created_utc|domain|is_crosspostable|no_follow|num_comments|score|subreddit|subreddit type|title|   rl|whitelist_status|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|  2360|         2|      12496|  5204|               2|        2|          59|  230|       23|             3|12383|12 54|               3|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
```

Figure 9 The distinct value counts of the 13 original selected features from the dataset

1. It had to be translated using “google translate library” first since it contained multiple languages.
2. Afterwards, the translated title was tokenized and stop words were removed.
3. On the final feature obtained, Named Entity count was extracted (anything that fits in the description of People, Nationalities, Building, company, country, books, events, law, language etc).
4. In addition to Money, number and Organization and Geopolitical Entity counts. This was done to gauge the effect of mentioning money, organization or a place would have on score.
5. Afterwards, POS tagging was used to extract count of propernouns.
6. Moreover, word count and length of title were generated as well. The work on part of speech tags and ner extraction was done using Spacy’s pretrained models.
7. Additionally, Sentiment analysis (using Spacy pretrained model) on title translated feature.
8. vectorization of stop words removed title features was done using TFIDF and Word2Vec was used to get set of features to train our model with. 2 vectorization techniques (TFIDF and Word2Vec) were chosen so that we can check the results of both. Also, word2vec has better capabilities to encode the sentence since it is considering the context of each word as well eg. Words ‘computer’ and ‘keyboard would be closely related.

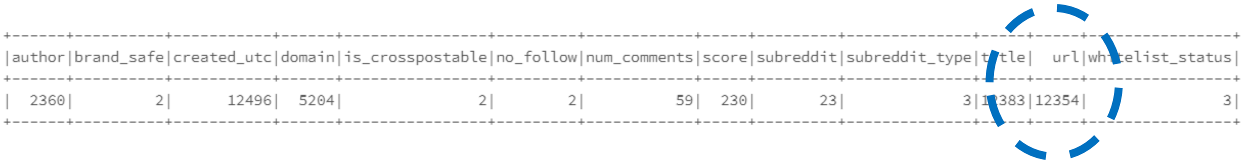


title
GCCの実行時に影響を及ぼす環境変数
Haken Audio
H&amp;R Block Miscalculates Its Own Taxes
Top 10 funny/offensive religious jokes
Bioengineers create stable networks of blood vessels
博士（はくし）が100にんいるむら
Predators keep the world green, ecologists find
プログラミング言語を作る
ランダムな計算機科学の論文を（グラフや図や引用付きで）作る

title_translated
Environment variables that affect GCC runtime
Haken Audio
H&amp;R Block Miscalculates Its Own Taxes
Top 10 funny/offensive religious jokes
Bioengineers create stable networks of blood vessels
There are 100 doctors
Predators keep the world green, ecologists find
Make a programming language
Make a random computer science paper (with graphs, figures and citations)

Figure 10 The number of techniques used to extract useful features from feature title

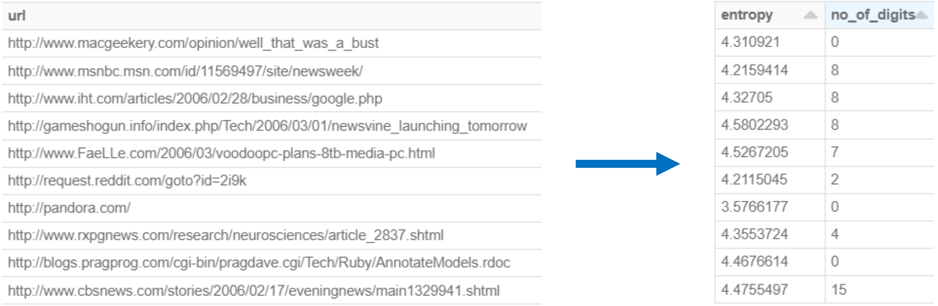
2.5 URL Feature



```
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|author|brand_safe|created_utc|domain|is_crosspostable|no_follow|num_comments|score|subreddit|subreddit_type|title|  url|whitelist_status|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|  2360|         2|      12496|  5204|               2|        2|          59|  230|       23|             3|12383|12354|               3|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
```

Figure 11 The distinct value counts of the 13 original selected features from the dataset

The URL feature had the exact url of the reference of the reddit post. This url was then used to extract entropy (function ‘entro’) and the number of digits present inside the URL feature. Entropy of a string is the probability of other characters occupying that same length of characterspace for the string. Or in simple words, if you get 1 string among N possible strings, then the string has entropy of N (assuming equal probability is assigned to all the strings). The output of the feature is as follows –

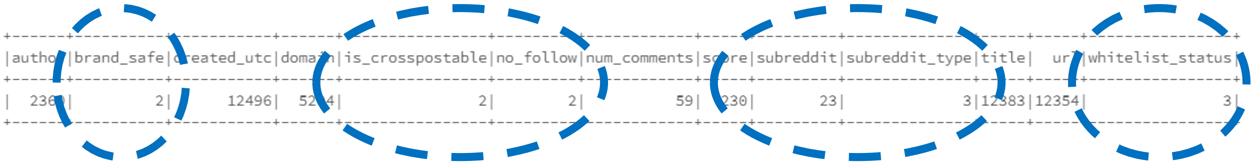


url
http://www.macgeekery.com/opinion/well_that_was_a_bust
http://www.msnbc.msn.com/id/11569497/site/newsweek/
http://www.iht.com/articles/2006/02/28/business/google.php
http://gameshogun.info/index.php/Tech/2006/03/01/newsvine_launching_tomorrow
http://www.FaeLLe.com/2006/03/voodoopc-plans-8tb-media-pc.html
http://request.reddit.com/goto?id=2i9k
http://pandora.com/
http://www.rxpgnews.com/research/neurosciences/article_2837.shtml
http://blogs.pragprog.com/cgi-bin/pragdave.cgi/Tech/Ruby/AnnotateModels.rdoc
http://www.cbsnews.com/stories/2006/02/17/eveningnews/main1329941.shtml

entropy	no_of_digits
4.310921	0
4.2159414	8
4.32705	8
4.5802293	8
4.5267205	7
4.2115045	2
3.5766177	0
4.3553724	4
4.4676614	0
4.4755497	15

Figure 12 Example of feature engineering with URL feature

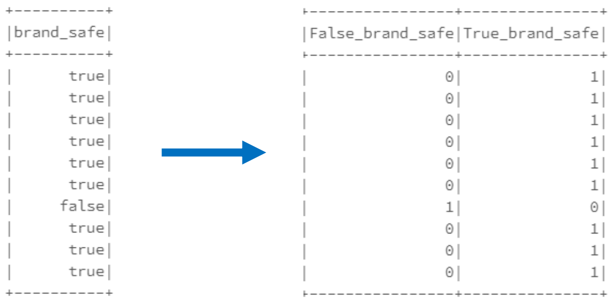
2.6 Brand_safe, is_crosspostable, no_follow, subreddit, subreddit_type, whitelist_status Feature*



```
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|author|brand_safe|created_utc|domain|is_crosspostable|no_follow|num_comments|score|subreddit|subreddit_type|title|  url|whitelist_status|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
|  2360|         2|      12496|  5204|               2|        2|          59|  230|       23|             3|12383|12354|               3|
+------+----------+-----------+------+----------------+---------+------------+-----+---------+--------------+-----+-----+----------------+
```

Figure 13 The distinct value counts of the 13 original selected features from the dataset

All the remaining original features having Boolean type data or categories less than equal to 3 where all directly transformed via onehot encoding. A custom one hot encoding function was prepared to add columns in the dataset with the headers of the category which is being encoded.



```
+----------+
|brand_safe|
+----------+
|      true|
|      true|
|      true|
|      true|
|      true|
|      true|
|     false|
|      true|
|      true|
|      true|
+----------+
```

```
+---------------+--------------+
|False_brand_safe|True_brand_safe|
+---------------+--------------+
|              0|             1|
|              0|             1|
|              0|             1|
|              0|             1|
|              0|             1|
|              0|             1|
|              1|             0|
|              0|             1|
|              0|             1|
|              0|             1|
+---------------+--------------+
```

Figure 14 Example of onehot encoding

With all the feature engineering, the count of features increased from 13 to 77.

3 Data Exploration

3.1 Exploring Scores

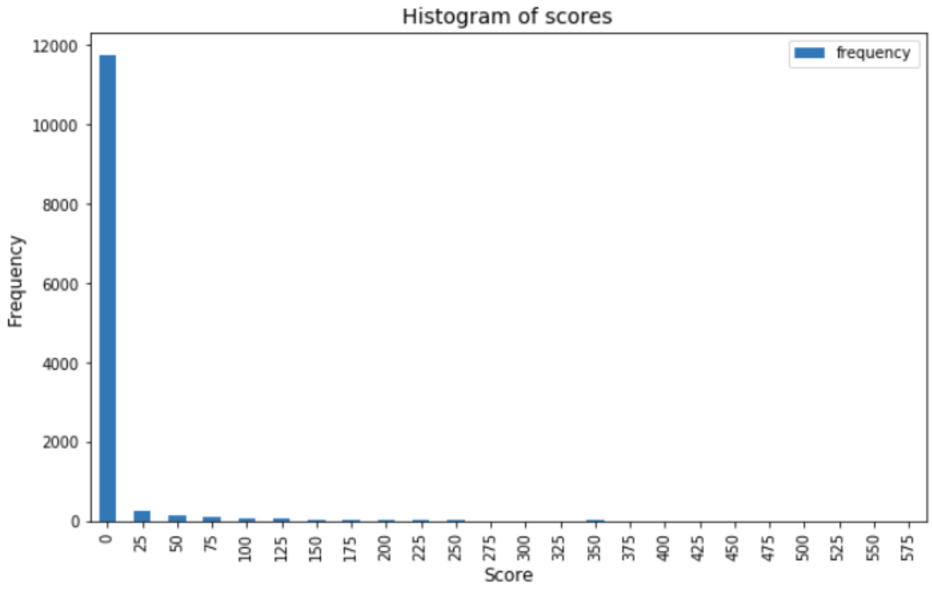


Figure 15 Histogram of value counts of scores in bins of 25

Fig 15 shows the distribution of scores in buckets of 25. It is evident from the countplot of scores that the target variable is highly skewed towards lower score especially score 0 (94% of the target variable has score less than 25)

3.2 Exploring Authors*

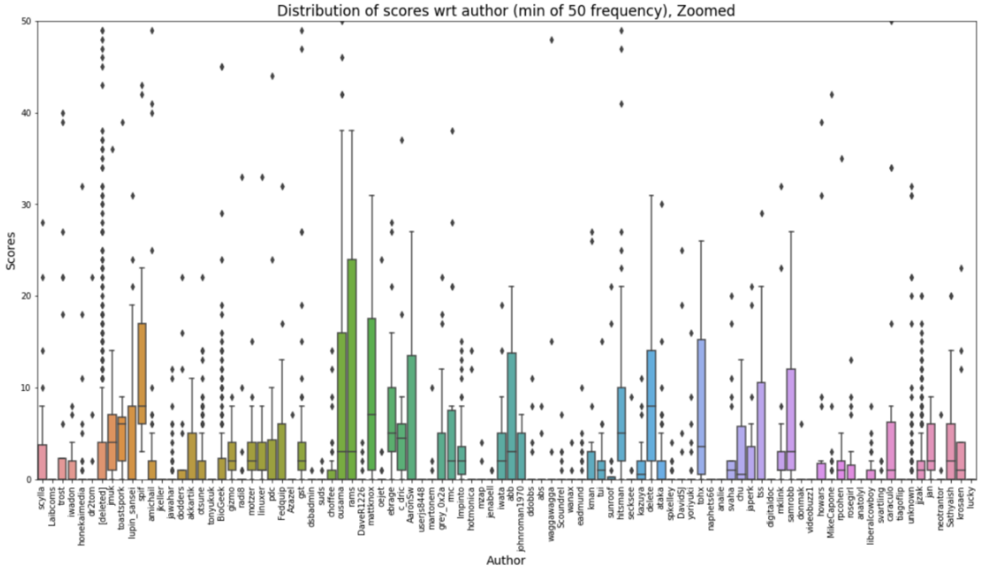


Figure 16 Boxplot Distribution of Author vs Score

From the exploration showin in Fig.16, the choice of authors have an impact on the score because certain authors have higher median scores than others. Moreover the black dots represent the outlier scores for the distribution. Since the data is highly skewed towards lower scores, which makes this graph expected. We cannot remove the outliers here as they might affect the final range of scores obtained, and hence might distort the learning boundaries of our machine learning algorithm.

3.3 Exploring created_utc*

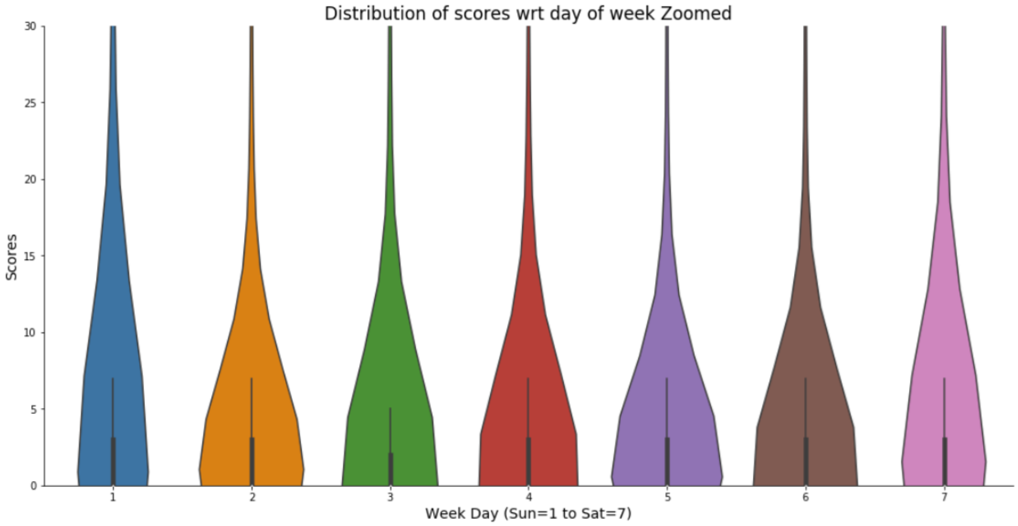


Figure 17 Violin plot of day of week vs score

In the violin plot shown in fig17, the distribution is shown like a boxplot, however the width of each 'violin' gives the kernel density distribution for that particular score. It is evident from the plot that the width of plot for day 1 and 7 are the lowest, meaning the probability of the lower scores is less on weekends.

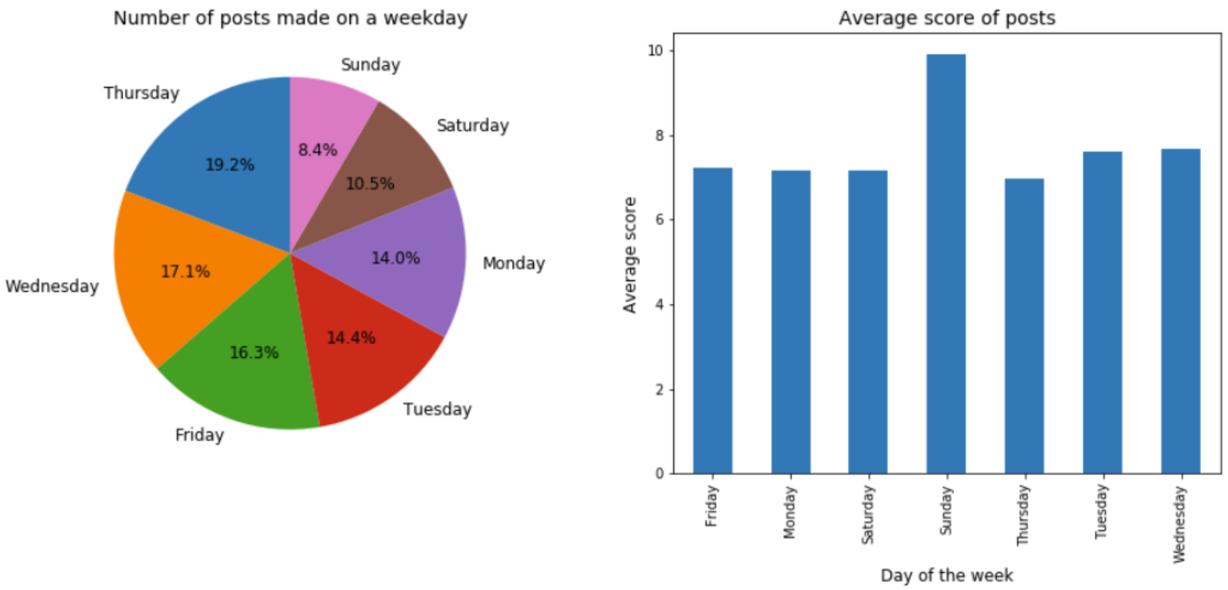


Figure 18 Pie chart and weekday vs avg. score histogram

Fig 18's pie chart suggests that the Saturday and Sunday receive the least amount of posts, however, the average score histogram shows that the average score is highest for Sunday.

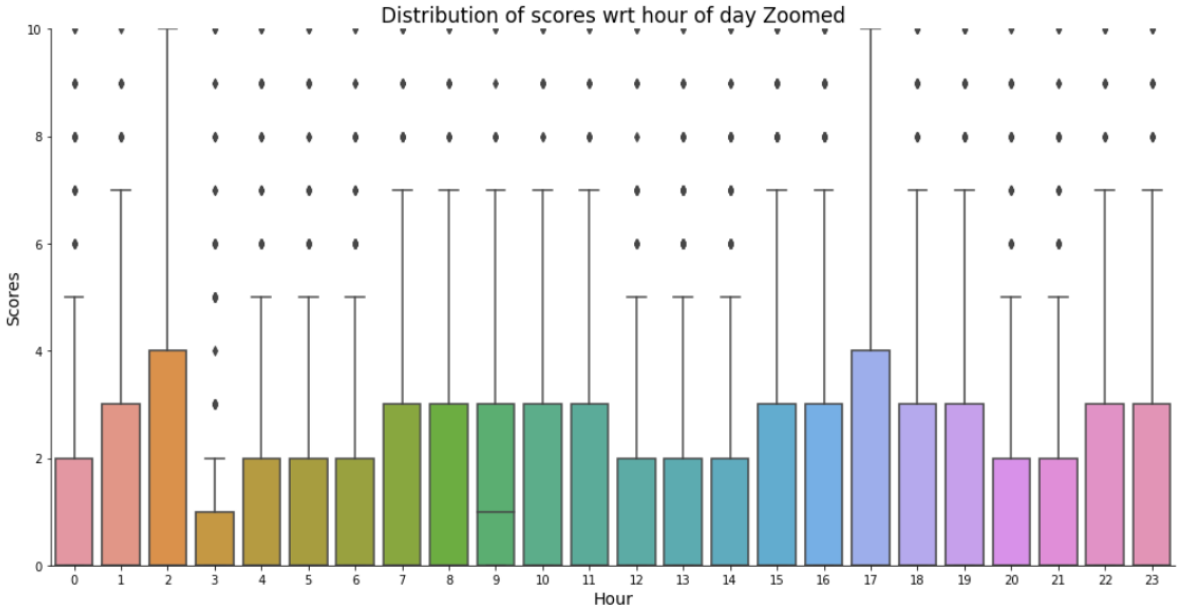
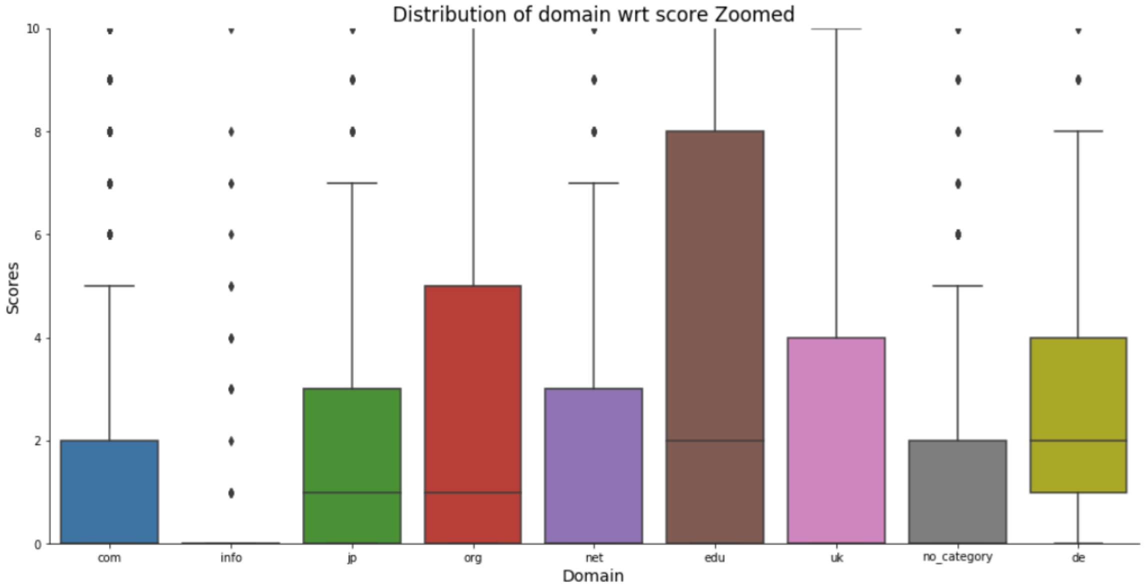


Figure 19 The distribution of scores with respect to hour of posting

Moreover, the fig19 shows the distribution of posting hour. It is evident that there is no significant change in the median score with changing hour of posting.

3.4 Exploring Domain*



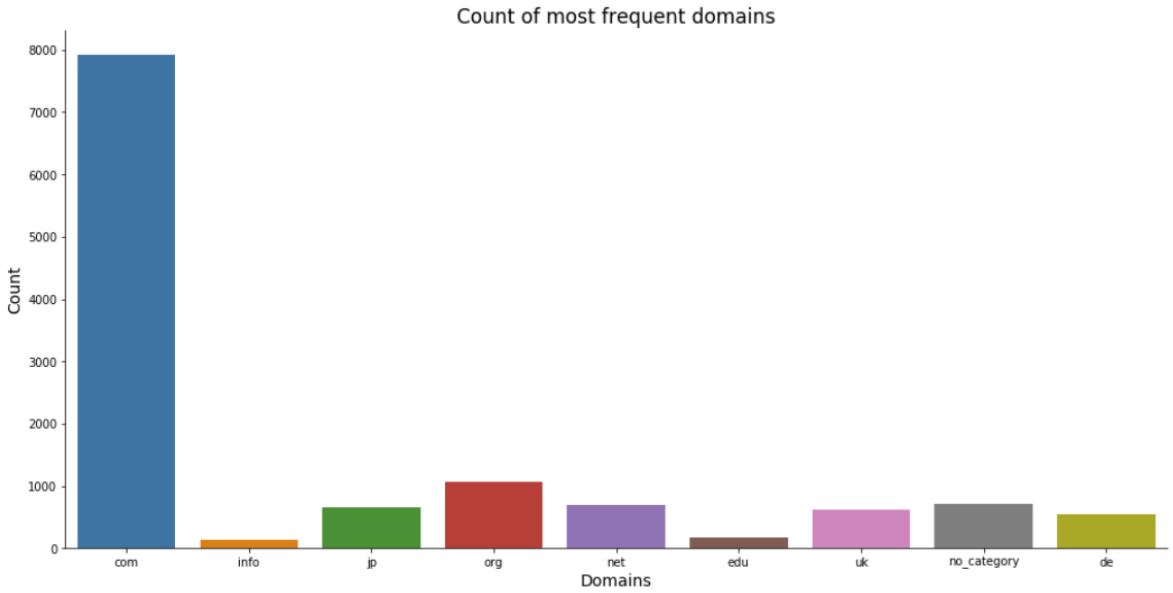


Figure 20 Count plot of domains (frequency more than 100) and Boxplot Distribution of Domain vs Score

The count plot of the domains suggests that the domain category ‘com’ represents most of the posts, whereas the boxplot of domain categories and score show that ‘com’ has one of the lowest median score. Moreover the category ‘edu’ which was underrepresented in the count plot, has one of the highest median score according to the boxplot in fig20.

3.5 Exploring Subreddit*

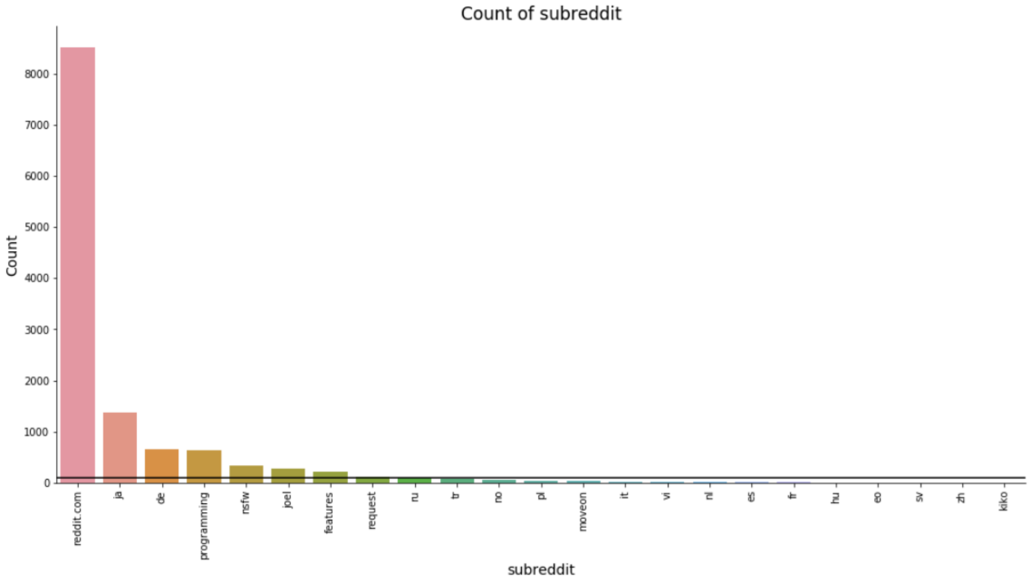


Figure 21 Count plot of subreddit feature

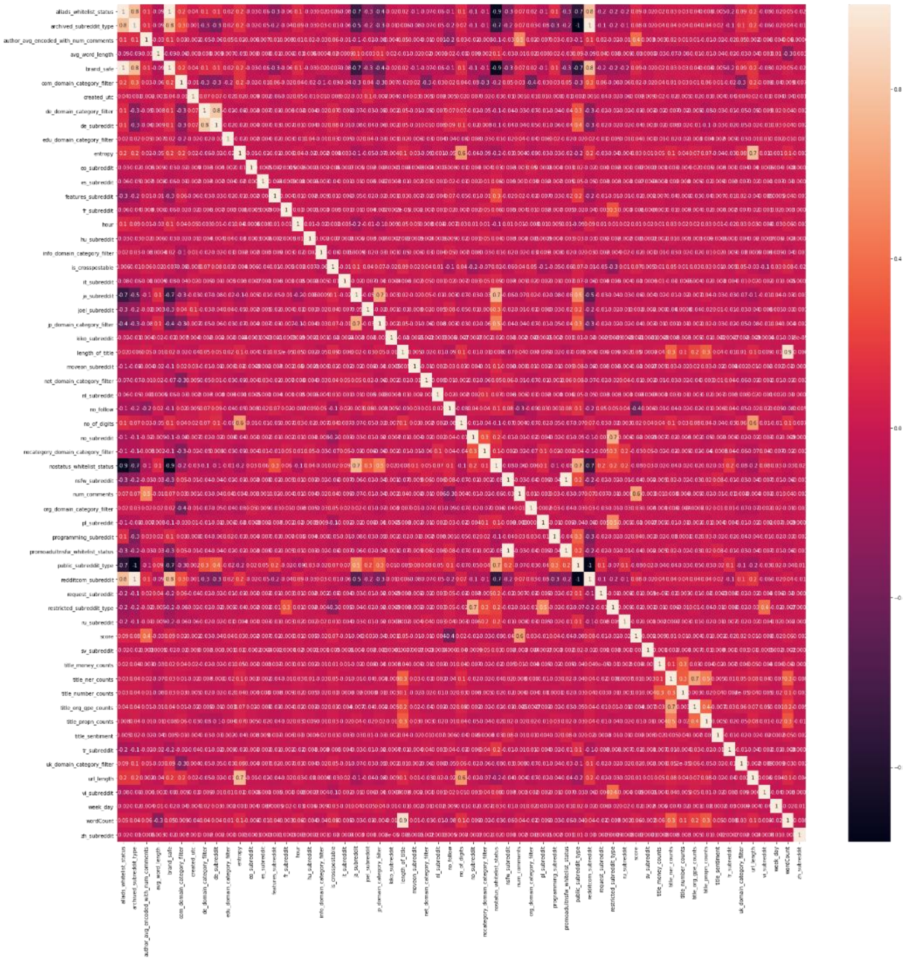
From the fig 21 countplot, it can be observed that the reddit.com is the over-represented subreddit. Moreover the black horizontal line represents the line of 100 count frequency. The categories lying after ‘tr’ can be considered as under-represented and can be called categorical outliers. We are keeping these outliers as well in the dataset as their values might help the algorithm understand the range of learning parameters.

3.6 Feature Correlation*

Figure 22 shows the correlation heatmap between all features. The best practice is that the features fed into the machine learning model should be independent. Hence using the heatmat, features were identified which were correlated amongst themselves by over 90%. The pairs were namely as follows –

['allads_whitelist_status', 'brand_safe’], ['allads_whitelist_status', 'nostatus_whitelist_status'], ['archived_subreddit_type', 'public_subreddit_type'], ['brand_safe', 'nostatus_whitelist_status'], ['length_of_title', 'wordCount'], ['nsfw_subreddit', 'promoadultnsfw_whitelist_status'], ['public_subreddit_type', 'redditcom_subreddit']

One from each pair was dropped to make the performance of the machine learning model better.



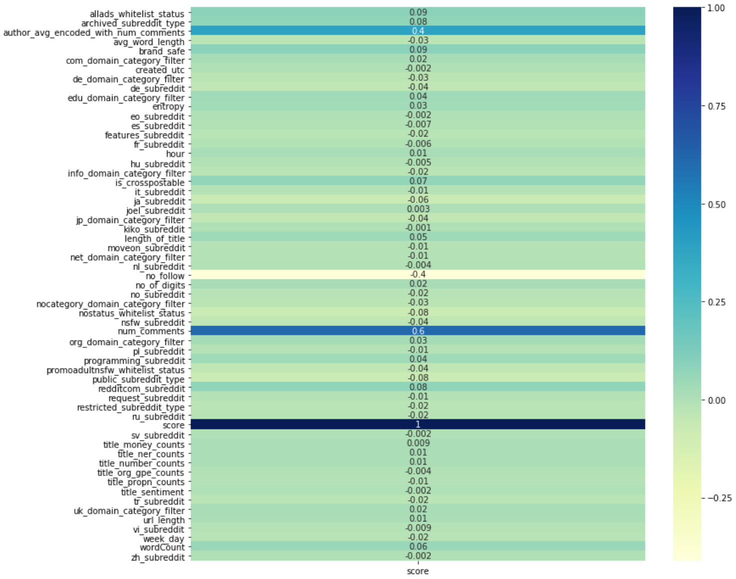


Figure 22 Correlation heatmap between all feature (top graph) and feature vs target (bottom graphs)

4 Model Development

4.1 Feature Selection*

Feature importance with random forest regressor was used to determine the feature's importance. The filter to select the features were of the importance of atleast 0.01 or more. With this, 6 featuers were selected. The features were **'num_comments', 'no_follow', 'author_avg_encoded_with_num_comments', 'wordCount', 'url_length'.**

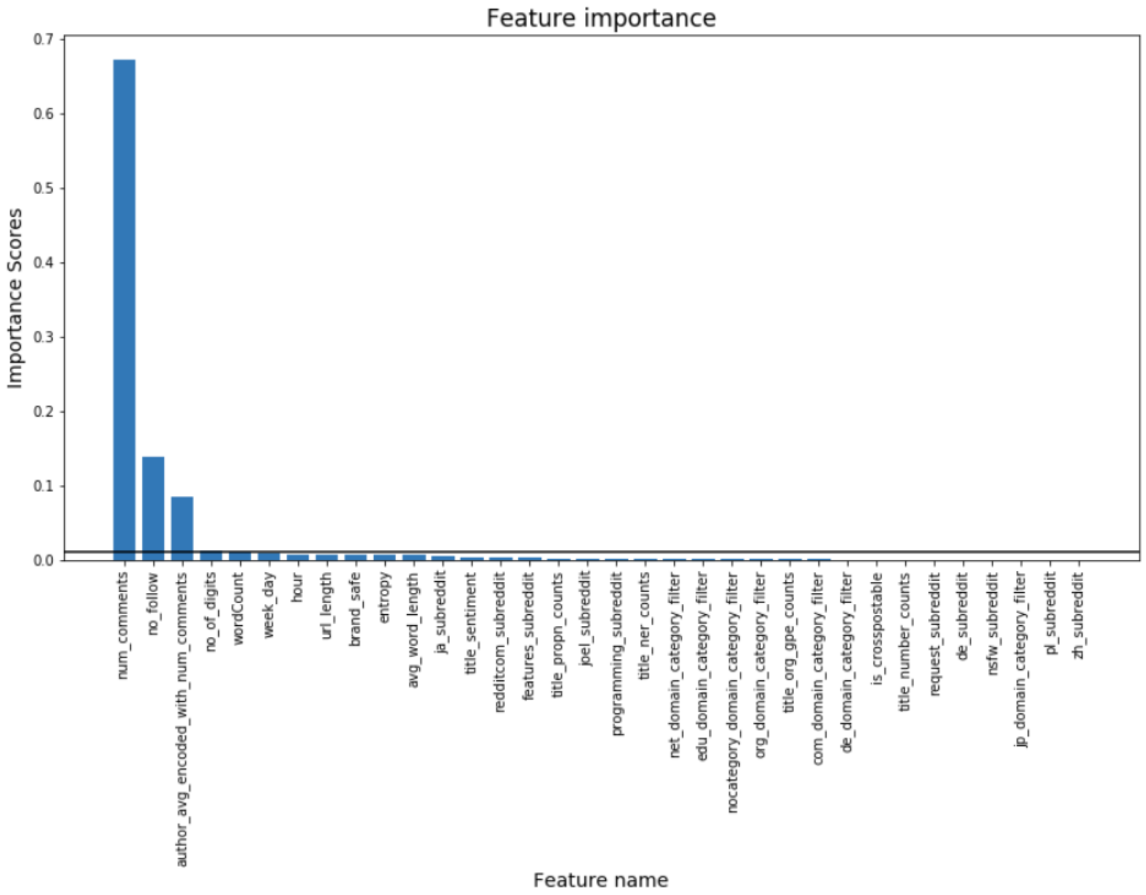


Figure 23 Feature importance bar graph with cutoff line at 0.01 importance score

4.2 Model Selection

A total of 3 machine learning models were implemented on this dataset.

4.2.1 Linear Regression

Linear regression is a supervised machine learning algorithm. It models a target based on the independent features and is most used to find relation between variables and forecasting. Since the data was skewed, hence the performance suffered in terms of R2 and RMSE. Hence the next model chosen was decision tree.

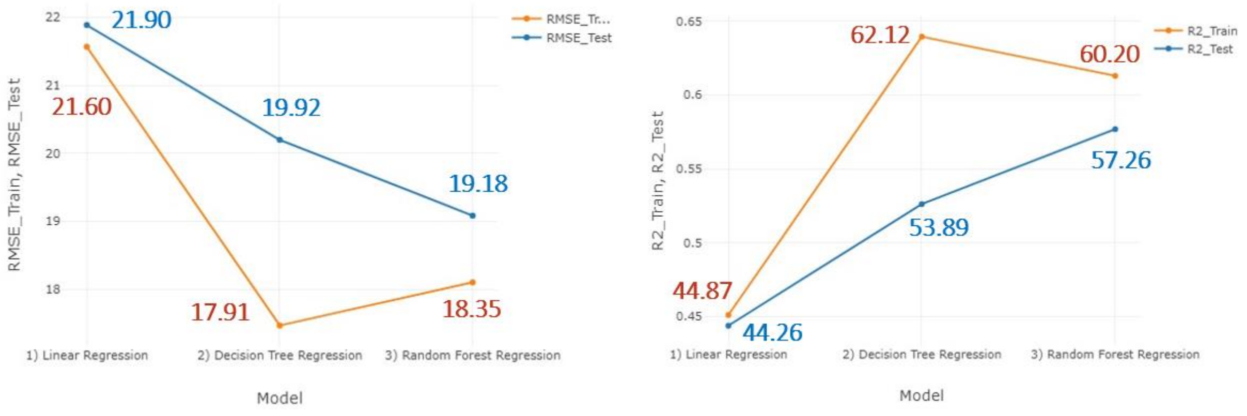
4.2.2 Decision Trees (Regression)

This method of supervised learning uses decision trees to go from observations about an item to the conclusion about the same. Here the target variable takes continuous values, hence it would be called regression tree. However, this resulted in overfitted model on the training data, hence Random Forest was the next choice.

4.2.3 Random Forest Regression*

This ensemble learning method uses multiple decision trees and take the mode or the mean of their prediction to arrive at a conclusion during training. This method helps avoid the overfitting the training which is often the case in decision trees.

4.3 Model Evaluation*



RMSE

R2

Figure 24 The RMSE and R2 graphs of all three ML models

	RMSE train	RMSE test	R2 train	R2 test
Linear Regression	21.607664	21.904	0.448753	0.442681
Decision Tree	17.9117	19.9224	0.621207	0.538956
Random Forest	18.3593	19.1813	0.602038	0.572619

From fig 24, it is evident that the linear regression has high RMSE and low R2 value in training and testing. Whereas the decision trees perform quite well on the training set but poorly on the test set suggesting that it overfits on the training data. Finally, random forest has the optimal combination of the train and test errors. Its results suggest that it was able to generalize the training data well.

Moreover, the evaualtion metric chosen here is R2 and RMSE. R2 is the proportion of the variation in the outcome that is explained by the predictor model. Hence higher the R2, the better is the model. The formula of R2 is given below, where y_i is the actual value, y_i^{hat} is the predicted value and y_i^{cap} is the mean of the target values.

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(y_i - y_i^{hat}\right)^2}{\sum_{i=1}^{N}\left(y_i - y_i^{cap}\right)^2}$$

Whereas RMSE is the root mean square error which is the square root of the mean square average error performed by the model while predicting the outcome of the target values. The formula of RMSE is given below, where y_i is the actual value, y_i^{hat} is the predicted value and N is the total number of target values.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}\left(y_i - y_i^{hat}\right)^2}{N}}$$

The best model with the above evaluation appeared to be Random Forest, however a learning curve was also plotted to ensure that the model was not overfitting.

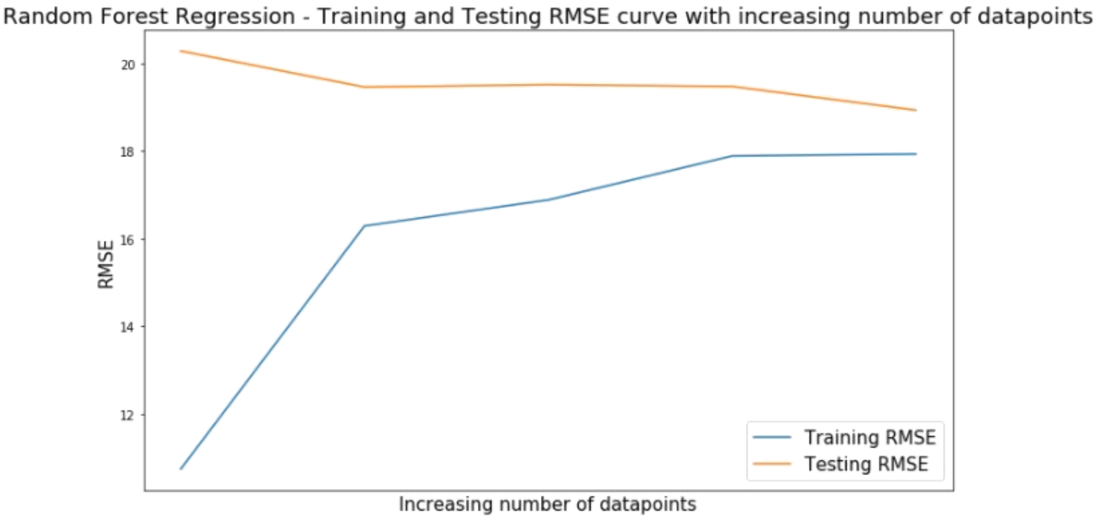


Figure 25 Random Forest learning curve

From the learning curve in the fig 25, it can be inferred that that with increasing number of data points, our model was able to learn well and reduce the rmse for the testing set. Also, the low gap between the train and test error suggests that the model has low variance or it is not overfit.

According to fig 26, the target was then split into bucket of score 0, 0-10, 10-30, 30-100 and 100+. In these buckets the model prediction performance was evaluates, i.e. how well the model was able to predict the correct scores under these bucket. It was observed that the model performed well in predicting the lower scores in bucket 0 and 0-10. But as the bucket size increased, the model’s accuracy dropped. This is due to the fact the data is highly skewed towards the lower score, hence model has more data to learn from the lower scores, hence the relatively more accurate prediction.

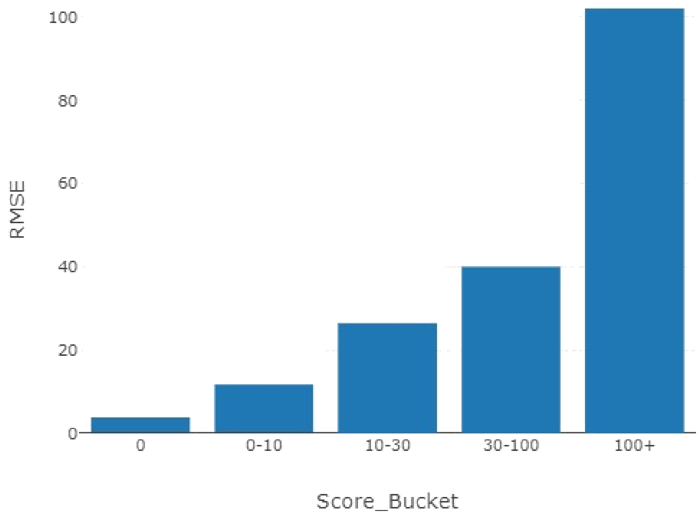


Figure 26 Score bucket or category wise evaluation

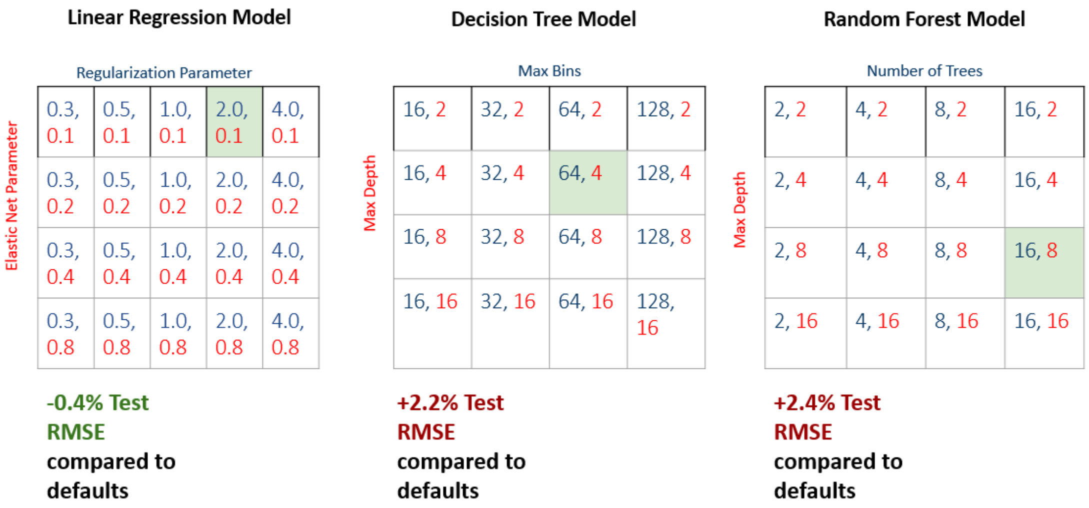
4.4 Best Model

Random forest is chosen as the best model as its results suggested that it was able to generalize the training data well and performed adequately. Moreover it has advantages such as –

- Optimal performance on both training & testing (lowest test error)
- Does not need scaling and normalization
- Parallelizable for fast training

5 Model Tuning and Tesing (Results)

5.1 Tuning



Linear Regression Model

Regularization Parameter / Elastic Net Parameter

0.3, 0.1	0.5, 0.1	1.0, 0.1	2.0, 0.1	4.0, 0.1
0.3, 0.2	0.5, 0.2	1.0, 0.2	2.0, 0.2	4.0, 0.2
0.3, 0.4	0.5, 0.4	1.0, 0.4	2.0, 0.4	4.0, 0.4
0.3, 0.8	0.5, 0.8	1.0, 0.8	2.0, 0.8	4.0, 0.8

-0.4% Test RMSE compared to defaults

Decision Tree Model

Max Bins / Max Depth

16, 2	32, 2	64, 2	128, 2
16, 4	32, 4	64, 4	128, 4
16, 8	32, 8	64, 8	128, 8
16, 16	32, 16	64, 16	128, 16

+2.2% Test RMSE compared to defaults

Random Forest Model

Number of Trees / Max Depth

2, 2	4, 2	8, 2	16, 2
2, 4	4, 4	8, 4	16, 4
2, 8	4, 8	8, 8	16, 8
2, 16	4, 16	8, 16	16, 16

+2.4% Test RMSE compared to defaults

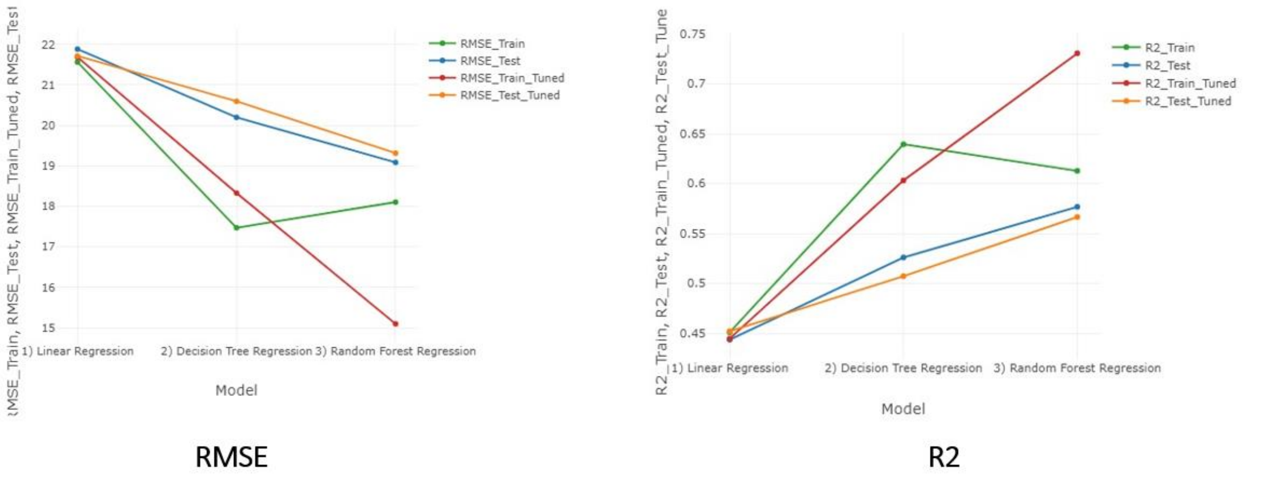
Figure 27 Model tuning hyperparamets grid search representation

In fig 27, the model hyperparamters are chosen for each and iterated over a grid of different combinations using gridsearch via 'CrossValidator' function. 10 fold cross validation was performed with the following hyperparameters –

1. Linear Regression - regularization parameter, Elastic net parameter
2. Decision Tree – Max Depth, Max Bins
3. Random Forest – Max Depth, Number of Trees

Tuning had little affect on the linear regression model, but as theory suggests, tuning had a large impact on the ensembles based learning methods; Decision Trees and Random Forest. The final results are mentioned in the fig 27.

It is evident from the graph (fig 28) that the tuned model performs hugely better on the training set and almost the same on the testing set in case of Random Forest Regression. However this is not the case for decision trees or linear regression



RMSE

R2

Figure 28 The RMSE and R2 of tuned vs untuned versions

5.2 Testing (Result)*

Metric	**Train Set**	**Test Set**	**OOT Set**
R2	60.20%	57.26%	54.83%
RMSE	18.35	19.18	20.75

All the feature transformation were applied on the OOT (out of time) dataset and Random Forest model predicted the target values. For the OOT test set, the model gave a similar performance as compared to the test and the train set. This suggests that the model was able to generalize the training data well and was not overfitting.

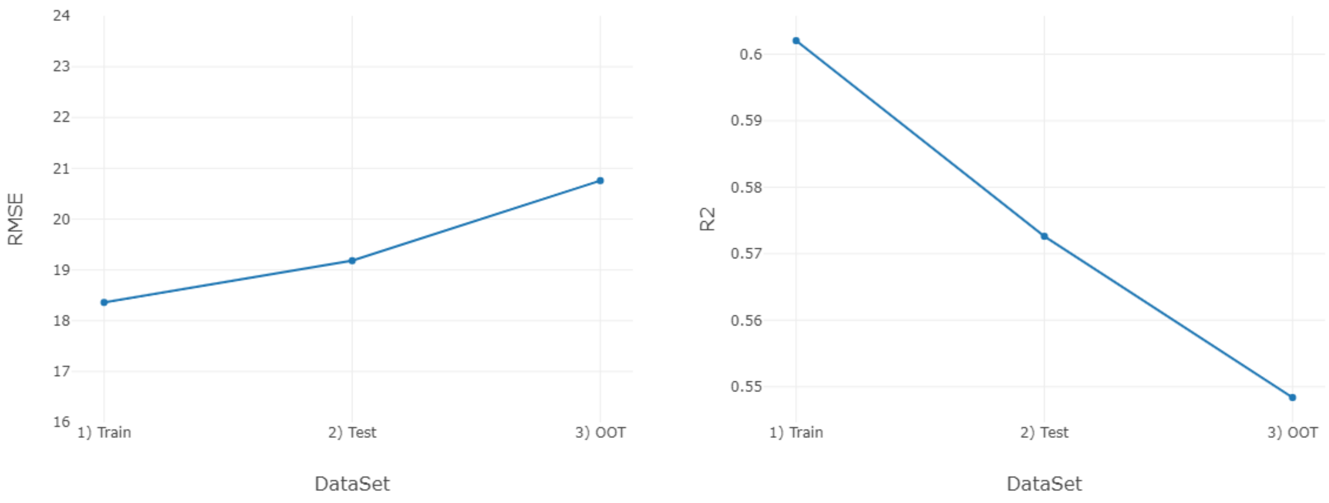


Figure 29 The performance graphs for all the data sets (train/test/oot) for random forest

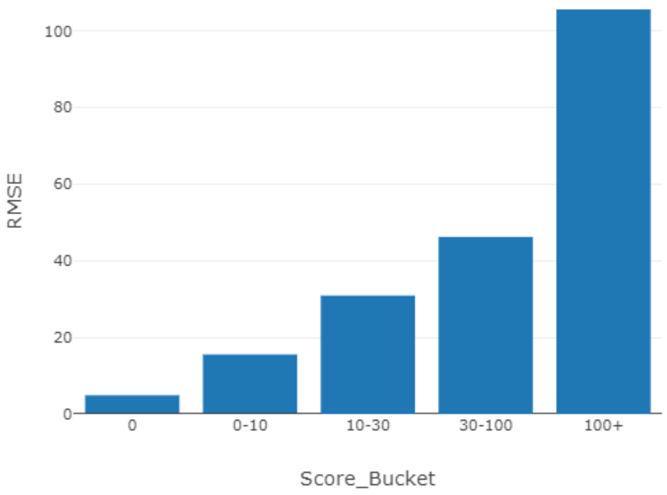


Figure 30 Score buckets for category wise evaluation

Moreover, according to fig. 30, similar trends are shown in the model testing on OOT test set as was seen during the training, where the scores with lower values are predicted quite well and the RMSE increases as we move towards prediction of higher scores.

References

1. https://spark.apache.org/docs/latest/api/python/pyspark.ml.html?highlight=word2vec
2. https://spark.apache.org/docs/latest/mllib-feature-extraction.html
3. http://www.sthda.com/english/articles/38-regression-model-validation/158-regression-model-accuracy-metrics-r-square-aic-bic-cp-and-more/
4. https://stackoverflow.com/questions/2979174/how-do-i-compute-the-approximate-entropy-of-a-bit-string
5. https://spacy.io/universe/project/spacy-sentiws
6. https://spacy.io/usage/linguistic-features