Popularity Prediction Using Linear Regression

Milad Ghanbari

1 Abstract

In this report the popularity prediction of comments was investigated based on linear regression technique. The chosen website for obtaining dataset and verification of developed model was Reddit. To achieve this goal, some useful features were extracted from the dataset. Rather than comment's content which usually determines the comment popularity, there are some other useful factors including number of replies on the comment (named children), whether the comment is in the main root or not (named is_root) and how controversial the comment is (named controversiality), most frequent word counting, etc. Furthermore, it was seen that considering the square of children and the product of controversiality and children as features can enhance the performance of estimation. Based on the idea that punctuation and stop words cannot usually determine the popularity of a comment, the inverse number of these words was considered as another feature for the model. By adding the last four mentioned features, the mean squared error (MSE) decreased by 0.025. The model parameters were obtained based on both closed-form and gradient descent approaches. Higher stability and less runtime were seen in the closed-form approach compared to the gradient descent approach (valid when the features are independent). Finally, the developed model was examined on the test data and achieved MSE was 1.2443.

2 Introduction

Popularity prediction is one of the interesting information used by marketers for obtaining most demanding topics which can be applied to their future business plans. The importance of text processing and its vital role in industries have been investigated in [1] and, additionally, some interesting text analytic subcategories such as natural language processing, information extraction and entity recognition have been addressed. Thanks to the progress of information technology and Internet, variety of social networks are available for achieving the mentioned purpose. The focus of this report is on Reddit. Reddit is a social news website created to be used for posting data either in text, images or links format which can be rated by the other users. In this report linear regression technique was investigated for popularity prediction of comments on Reddit . There have been several works related to the popularity prediction of comments on Reddit and in the following some of the important ones will be addressed. In [2], authors ignored the content of comments and found the early comments have a correlation with the popularity of the comment. Author in [3] realized that linear regression model performed 14% better compared to baseline predictor. Both SVM and linear regression approaches were investigated in [4] and it was found that SVM was slightly more desirable than linear regression approach in terms of results.

The number of 12000 data was collected from the mentioned website. Five information are embedded in each one of these data, popularity_score (indicates how popular the comment is), children (the number of replies on this comment), controversiality (reveals how controversial the comment is), is_root (indicates whether the comment is root or not) and text(the raw text of the comment). After the data were broken down into train, validation and test partitions, some features need to be extracted for constructing a model. As mentioned, we already have three measured features named children, controversiality and is_root. One of the other features which can be obtained

from the raw text is word counting. Before feature extraction, some preprocessing techniques must be performed on the raw text data for simplification. Two used preprocessing techniques were lower-casing and splitting text data based on white-space tokens. After the text data were preprocessed, the 160 most frequently occurring words over all training comments were found. These 160 words were used as features and the data matrix was filled by counting number of each obtained feature in each text. Based on intuition, we can expect both punctuation and stop words have least effect on the popularity of a comment. Consequently, the inverse number of these words can be considered as a new features for the model. An improvement in the performance of the estimation was seen by using these new features and the performance metric (MSE) dropped by 0.025. In order to obtain the parameters of the developed model, two different approaches were applied, namely closed-form and gradient descent approach. Based on the less computational complexity of the closed-form approach, less runtime was seen compared to the gradient descent approach.

3 Dataset

As mentioned, the dataset is a collection of comments, extracted from a Reddit subreddit. In the basic unprocessed form, it is consisted of 12000 comments along with some basic comment specific features. This dataset is in the form of a python list where each value is a dictionary consisted of these basic features:

- **Text:** which is the main body of the comment.
- **Children:** which indicates how many replies the comment has received.
- **Controversiality:** A Reddit based binary variable indicating how controversial the comment is.
- **is_root:** A binary variable indicating whether a comment is the main one in a discussion or not.
- **popularity_score:** This is the target variable which we will try to predict. It indicates the general popularity of the comment.

We have split the dataset into three partitions: Training set which is the first 10000 comments, Validation set which is the next 1000 comments and finnaly, the Test set which is the final 1000 comments.

We will use these partitions in training, tuning and evaluating our models. We have extracted several features in order to construct our feature vector and the pre-processed dataset. These include both text-based and the aforementioned numerical metrics of the raw dataset. Figure 1 illustrates the general correlation among the numerical features of the dataset. We use this figure to design some of the features. The extracted features are as follows:

- 160 most used words in all of the 10000 values of the dataset are extracted. Then for each comment the number of times each of these words are repeated is counted and saved.
- By general analysis of the comments and Figure 1, it is understood that *children* parameter plays an important role in the popularity of a comment as a result of which, the square of this parameter is included in the feature vector.
- The multiplication of the controversiality and children parameters are also included in the feature vector.
- The inverse multiplication of the *children* parameter and the number of punctuation marks is included in the feature vector. If the multiplication is zero, this parameter is set to zero.
- The inverse multiplication of the *children* parameter and the number of stop words is included in the feature vector. If the multiplication is zero, this parameter is set to zero.
- Finally, the raw forms of the is_root, children and controversiality parameters are included in the feature vector.

In conclusion, the feature vector without our features has 164 elements. These include 160 top words, is_root, children and controversiality metrics and a final bias term. This dimension increases to 168 when adding our designed features.

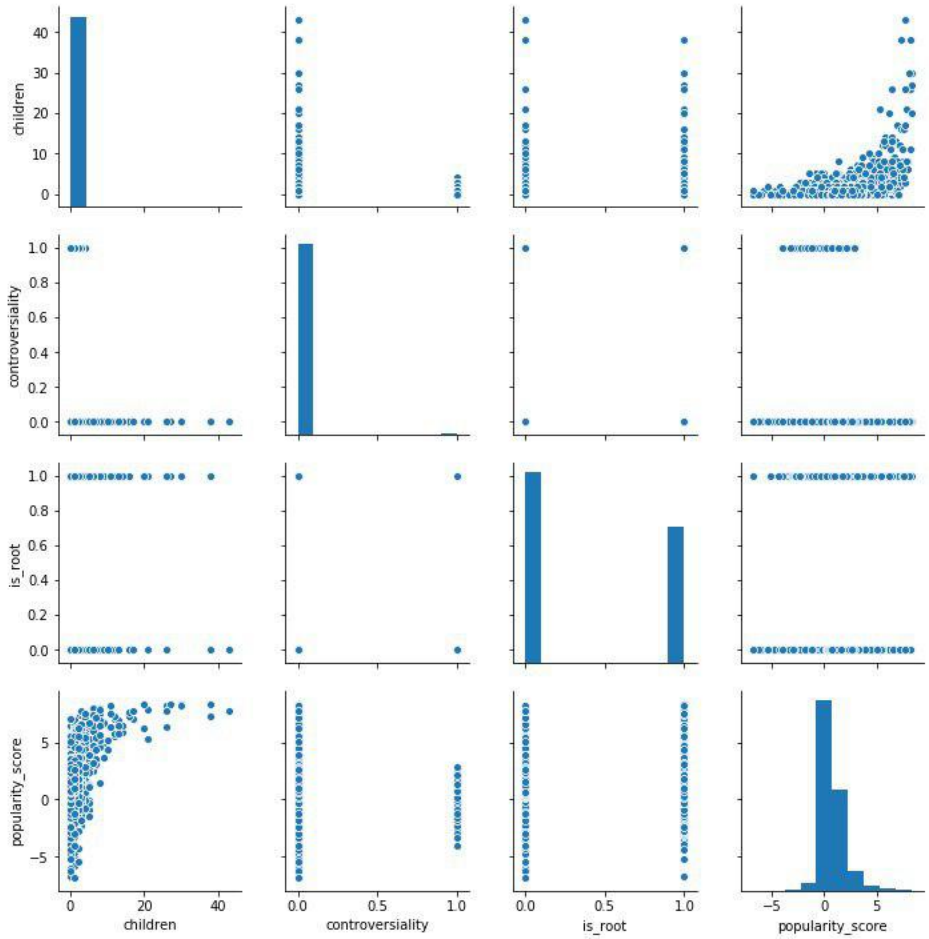

Figure 1: Correlation between numerical features of the dataset

4 Results

4.1 Runtime and Stability

Before implementing the new features we will test and run experiments on the models using only the 160-top-words feature. In Table 1, runtimes of the closed form and the gradient descent algorithm are illustrated. Both algorithms are tested using a simple feature vector consisting only of the numerical metrics of the dataset i.e. "children", "is_root" and "controversiality". Multiple choices of the learning rate are used for the gradient descent algorithm in order to display its effect on runtime. Moreover, based on empirical results, we have chosen the values of other gradient descent parameters as follows:

$$||w_i - w_{i-1}||_2 > \epsilon; \epsilon = 0.00001$$
$$\alpha_i = \frac{\eta_0}{1+\beta}, \quad \alpha = LearningRate, \quad \eta = initial\ Learning\ rate = 0.000001\ and\ \beta = Controller\ for\ the\ speed\ of\ decay = 0.000001$$

As can be seen in Table 1, the closed form has the lowest run time due to the lower computational complexity. As for the gradient descent, by lowering the learning rate the runtime increases. Using a fixed input matrix, the closed form of the linear regression produces a single result. However, because of the hyperparameters, the gradient descent algorithm produces various results. The goal is to tune the gradient descent model so as to get results which are the closest to those of the closed form. The stability of gradient descent approach highly depends on its learning rate and for high learning rate the approach becomes unstable.

Algorithm	Runtime(s)
Closed Form	0.000226
Gradient Descent (A)	0.264
Gradient Descent (B)	1.164
Gradient Descent (C)	3.0955

Table 1: Illustration of the runtimes of both algorithms. A: Learning rate = 0.00001, B: Learning rate = 0.000001, C: Learning rate = 0.0000001

4.2 Model Comparison

In this section we will compare the results of the closed form approach using several groups of features. Table 2 illustrates the mean squared errors of this algorithm based on the training and validation sets.

Features	Training	Validation
No Text Features	1.0846	1.0203
60 Top Words	1.0604	0.9839
160 Top Words	1.0477	0.9950

Table 2: Illustrating the mean squared errors of the closed form algorithm on several groups of features.

By observing the table, it can be seen that by adding more features to the feature vector, the mean squared error decreases which indicates a better learning on the training set and the validation set. It is noteworthy to mention that the results of the gradient descent algorithm are almost the same as the above values and the difference is negligible.

4.3 Our Features

In this section we will implement our features with both methods using multiple combination of models. Table 3 illustrates the results of this implementation on the training set and the validation set.

Features	Training Set (Closed Form)	Validation Set (Closed Form)	Training Set (Gradient Descent)	Validation Set (Gradient Descent)
60 Top Words	1.0604	0.9839	1.0604	0.990
60 Top Words + Ours	1.0018	**0.9589**	1.0018	**0.9601**
160 Top Words	1.0477	0.9950	1.0477	0.9942
160 Top Words + Ours	**0.9898**	0.9673	**0.9898**	0.9623

Table 3: Illustration of the MSE of multiple combination of features based on the Validation and Training set using both closed form and gradient descent methods.

Table 3 illustrates how these models perform using these combination of features. On the training set, models using 160 top words along with our designed features outperform others which indicates the effect of these new features. However, MSEs on the validation set indicate that the model with 60 features along with our features outperform others in both closed form and gradient descent methods. Moreover, by comparing the general results of the validation set and the training set, a small amount of overfitting is observed. Based on this table, we pick the model with 60 top words and our designed features since based on the validation set, the MSE is lower compared to those of others. We observe a 0.025 improvement using our features compared to the best model without these features, which is the one with 60 top words.
Finally, we evaluate our best model on our test set using both closed form and gradient descent methods. The results are illustrated in Table 4.

Features	MSE of Test Set
Closed Form	1.2443
Gradient Descent	1.2479

Table 4: Results of our model applied to the test set using both methods

By comparing the result of the test set and the validation set, overfitting is observed in the model.

5 Discussion and Conclusion

Based on our results, the extra features other than word count has resulted in a decrease in the mean squared error and improving the final accuracy. Also, another important feature is 'word counting' which takes the occurrence of top most frequent words into account. Having said that, the key idea behind word counting feature is that *non-content* words, such as stopwords and punctuations, which can't appropriately influence the popularity score of the text, should be trimmed. Thus, the inverse number of these words is considered as another feature which reduces MSE of the model .
We have implemented two different approaches for the linear regression algorithm. The closed-form results in a unique answer while the gradient descent algorithm, having multiple hyperparameters, leads to various results under different conditions. The learning rate, the effect of which is analyzed in previous section, affects the runtime and stability of the algorithm. Finally, we conclude that multiple combinations of features have a significant effect on the performance of the model, thus displaying the importance of feature engineering. However, to extract features from the text, there are other potential concerns that can possibly affect the model. The following are some suggestions for future works:

- We need to take intentionally misspelled words into account and remove them, if any. For example, "veryyyy" and "very" refer to the same word where the repeated characters should be eliminated from the text.
  Also Word stems should also be considered. All variants of a word can be reduced to a single term (e.g. $\{go, went, gone\} \rightarrow \{go\}$). Finally, the word counting feature can be more challenging by saying that other than pruning the *non-content* words (e.g. common recurring stop words such as "the" or "and"), very rare words that occur less than a certain threshold (e.g. less than 20 times in 10000 words) can also be deleted from the text. However, recent studies [5] revealed that improper word deletion only based on word frequencies may adversely affect the feature extraction process, because words with less frequency of occurrence often contain more information.

- In order to extract the desired features more efficiently, thus increasing the final accuracy, more complex models can be implemented.

References

[1] A. Moreno and T. Redondo, "Text Analytics: the convergence of Big Data and Artificial Intelligence", 2016, International Journal of Interactive Multimedia and Artificial Intelligence Vol. 3.

[2] A. Terentiev and A. Tempest, "Predicting Reddit Post Popularity Via Initial Commentary", 2014, Stanford library.

[3] H. Kang, "Predicting Reddit /r/ relationships Post Popularity", 2016, (https://pdfs.semanticscholar.org/fcd9/3f5bcc6544a1be44c181bed412ab7c506abf.pdf)

[4] J. Segall and A. Zamoshchin, H. Sahay and L. Wang, "Predicting Reddit Post Popularity", 2012, Stanford library.

[5] Liang, Hong and Sun, Xiao and Sun, Yunlei and Gao, Yuan, "Text feature extraction based on deep learning: a review", EURASIP journal on wireless communications and networking, 2017.

Paninski, Liam, "Estimation of entropy and mutual information", Neural computation, 2003, MIT Press.