COMP 551: Mini Project 1

Group 43
Charles McCluskey, Salman Refayet and Grégoire Moreau

January 2019

1 Abstract

The focus of this project was to practice using linear regression and to investigate its effectiveness to predict the popularity of comments on Reddit. We used a dataset of 12000 independent, out of context comments with a specific set of auxiliary information that we used as primary features. To improve the accuracy of prediction, we experimented with a wide variety of potential features and eventually settled on the log of the number of children, the square of the number of children (both modifications of an existing feature), and a check on the length of the comment. We discovered that those features improved the quality of our model, and we found that the gradient descent approach was slower than the closed-form approach.

2 Introduction

The primary goal of this project was to get hands-on practice with linear regression and to examine how effective it would be at predicting the popularity of comments on reddit. We started with the use of base features such as whether the comment was controversial or not (binary), is a root or not (binary), how many replies it has (integer value), and how frequently it uses 0, 60, or 160 of the 160 most frequently used words in our dataset. For our feature selection, we first tried to search for links to either YouTube or image hosting sites, and then tried to do simple natural language processing using the nltk Python package. We abandoned this idea as the potential new features didn't really improve our model, and settled in favor of new features based on transformations of existing ones, which proved significantly more effective at improving the final model.

3 Dataset

The dataset was a collection of 12000 comments seemingly randomly selected from the "AskReddit" subreddit on the reddit.com, each comment having the following information associated with it: its controversiality, how many replies it had, whether it was a root comment, a popularity score, and of course the text itself. We split the data as follows: the first 10000 comments acted as the training set, the following 1000 were used as the validation set, and the last 1000 were used as the testing set. Aside from the base required features of using the raw data we were provided and comparing the texts with the 160 most frequently used words of the entire dataset, we also included the following features in our model: whether the comment possessed 21 or more characters (binary), the log of the number of children, and the number of children squared. As far as ethical implications go, one point that our team found the most concerning is how a dataset such as this, though perhaps a more politically charged one, could be used in the making of bots to artificially influence online discussions and polls, especially since the original posters of those comments would be totally unaware of such an endeavour.

4 Results

4.1 Runtime comparison between the gradient descent and closed-form solution

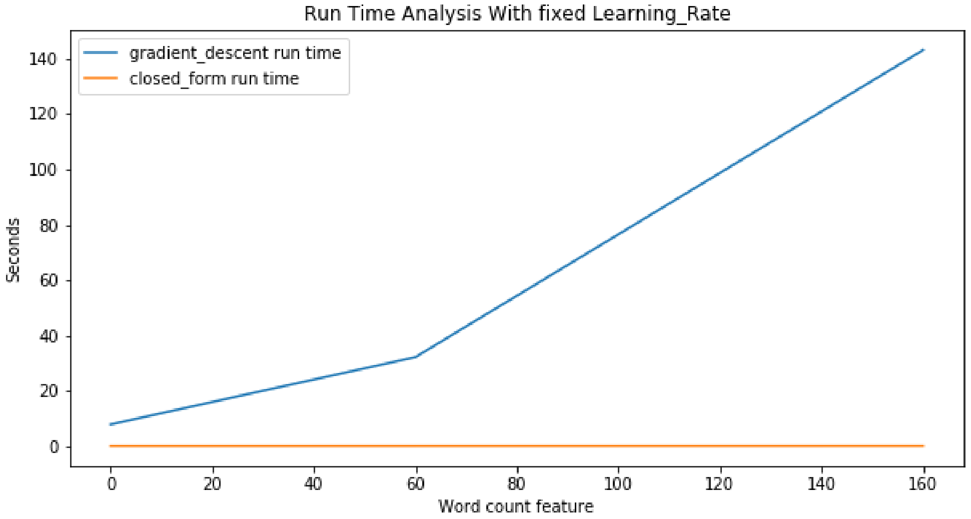


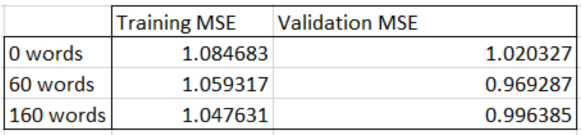
Figure 1 : MSE's of the base model with different word counts

Here we compare the run time of gradient descent and closed form linear regression. We pick the learning rate, alpha = 0.003 as it is the rate at which the algorithm converges for all the number of word features 0, 60 and 160. For a given alpha, gradient descent is slower across the entire word feature list and significantly so when the number of words used is 160. The algorithm takes 50,000 iterations to reach an MSE of 1.086265, very close to the closed–form MSE. It takes 30,000 and again 50,000 iterations to reach MSE 1.069323 and 1.056293 for 60 and 160 word features respectively. Therefore, closed form regression is more efficient in this case and of course, optimal. However, with a fixed or higher alplha, gradient descent performs equally fast when it runs with only the three basic features. Closed form can be computationally heavy if n is very large.

In terms of stability, closed form regression gives optimal weights directly which makes the model more stable in terms of accuracy. However, a very important aspect of stability is being able to approximate weights when there is a not unique solution, i.e. $X^T X$ is non-invertible. Such cases may arise due to multi-collinearity of features and then closed form approach fails.

4.2 Tests on the model with no added features

The table in figure 1 shows the training and validation mean-squared errors for a model using only the base features, a model using the base features and information about the 60 most used words in the database and a model that uses the base features and the full 160 most used words.



	Training MSE	Validation MSE
0 words	1.084683	1.020327
60 words	1.059317	0.969287
160 words	1.047631	0.996385

Figure 1 : MSE's of the base model with different word counts

The following bar chart in figure 2 presents the same information in a more visual way.

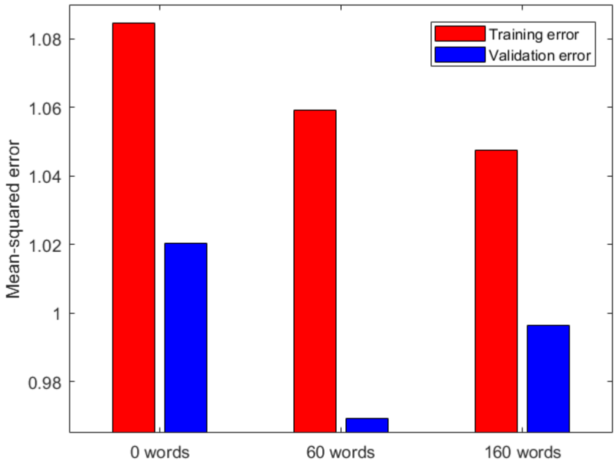


Figure 2 : MSE's of the base model with different word counts

It is made evident by this graph that the model performs better on the validation set when it only uses the 60 most used words. The reasons explaining this will be detailed in the discussion. The next experiments have thus been conducted using only the 60 most used words.

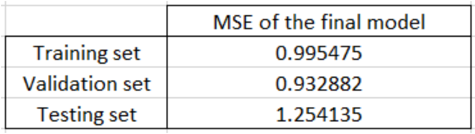
4.3 Tests on the added features

As explained in the introduction, the first meaningful feature we added was the natural logarithm of the number of children of the comment, or 0 if the comment had no children.

Associating this new feature with our simple model with the 3 base features and the 60 most used features gave us a new model with a mean-squared error on the validation set of 0.938403. This represents a decrease of 0.0309 of the mean-squared error compared to the model without that new feature.
The second feature we added was the square of the number of children. Adding this feature to the base model gave us a mean-squared error on the validation set of 0.949556. This represents a 0.0197 decrease of the MSE compared to the base model.
The last feature we added to the model is a check that returns 1 if the comment is longer than 21 characters and 0 otherwise. Adding this feature to the base model gave us a mean-squared error on the validation set of 0.964568. This represents a decrease of 0.0047 of the MSE, still compared to the base model.

4.4 Tests on the final model

The final model thus includes the 3 base features (is_root, controversiality and children), the number of appearances of each of the 60 most used words in the dataset, and the 3 new features described above. The following table describes its performance using the MSE for the 3 sets.



	MSE of the final model
Training set	0.995475
Validation set	0.932882
Testing set	1.254135

Figure 3: MSE's of the final model on the different sets

The validation error with the 3 new features combined represents a decrease of 0.0403 of the MSE compared to the model without those features. However, we can see that the error on the testing set is huge compared to the errors that we have seen before.

5 Discussion and Conclusion

As shown in the results, the model gives a larger error when the features include the word count of the 160 words that are the most used in the dataset instead of only the 60 most used words. This error happens because the model is overfitting due to the large number of the features, especially since the training set is the one that has the biggest impact on the list of most used words. We might want to remove words like "the", "it", etc. that don't provide significant information. Detection of the features' impact can be further analyzed by computing the standard errors and other statistical properies of the features and their interactions, making this another point of consideration for future experiments.

It's also important to notice the large MSE that the model creates with his predictions on the testing set. This happens because, we kept the features that gave the best decrease of the MSE on the validation set. The final model might thus be overfitting the validation set, which explains why it's not performing so well on the testing set. One possible solution to that problem is using cross-validation for our future ML projects.
We can see that our added features don't increase the model's quality by a huge amount. This is probably because two of them (log of the children and square of the children) are correlated with one of the base features (children), which means they don't provide a lot of new information to the model, while the third one is pretty arbitrary and doesn't provide a lot of information about the comment's content.

As explained in the introduction, we experimented with a lot of different text features without getting much results. we used the nltk Python package to count the number of words of different types nouns, adjectives etc and tried to see if that would be a useful feature. Unfortunately, those features didn't have a significant impact on the validation error. We believe that a comprehensive study on Reddit's comment along with the associated important features will improve the predictability of the model

6 Statement of Contributions

Charles handled the text processing part and found ideas of potential new features. Salman wrote the gradient descent function and compared its performance to the closed-form solution. Grégoire wrote the closed-form function and the functions implementing the new features.