# Comp 551 Project 1

Hasnain Mamdani

Paniz Bertsch

Jean Djamen

January 31, 2019

## 1 Abstract

In this project, we investigated the performance of linear regression models for predicting popularity of comments on Reddit. Using gradient descent and the closed-form solution, we calculated the weight matrix. In doing so, we were able compare their performance and execution time using a variety of features. We used closed-form solution approach over gradient descend for evaluating various experiments in this project because of its performance in terms of run-time and accuracy. Comparing models with top-60 words vs top-160-words, we observed a small reduction in root-mean-square error with top-60 words. Therefore, we chose top-160 words in our model as it resulted in better prediction with negligible increase in execution time.

Additionally, we added a second pre-process stage to our data; it composed of 3 extra interaction features. Our final model (using SciKit TF-IDF feature extraction and our 3 new features) was used to make prediction on testing data and was found to have our original prediction.

## 2 Introduction

There have been various studies published on the content based popularity prediction. Yu et al. (2011) converts message to a word vector using bag-of-word representation and uses SVM and Naive Bayes to predict message popularity [1]. Tartar et al. (2011) predicts the expected popularity of articles based on the number of comments during a short observational period using linear regression [2]. Our goal in this project is to implement a classic linear regression model to predict popularity of Reddit comments. The dataset

used in this project consists of Reddit comments, their popularity score, the number of replies they have, the controversy generated by the comment, and whether or not the comment is a direct reply to a post.

To improve our model, we made use of the python library SciKit to get rid of unimportant, yet reoccurring words. This done, our next step was to create new features that allowed us to make better predictions. We compare the root mean square and run times of several models. We applied both the closed form solution and gradient descent and observed that the closed form solution was superior in both metrics. However, we observed that with smaller learning rate, gradient descent for weight matrix converged faster.

## 3 Dataset

Our data consist of 12000 data points, out of which we used 10000 data points for training, 1000 for validation and 1000 for testing. The first pre-process step was to use the python library Counter to get the 160 most popular words in our training data and set these as our features. Just looking at these, we saw that many of them were insignificant (e.g. "the", "a", "is") so we decided to use another pre-process method. This one used SciKit TF-IDF. In a large text corpus, some words carry very little meaningful information about the actual contents of the document but are present i abundance. If we were to feed the count of words directly to a classifier, those very frequent terms would shadow the frequencies of rarer yet more interesting terms. Tf means term-frequency while tf–idf means term-frequency times inverse document-frequency[3]. This already allows us to get better results compared to the naive method of ranking word by frequency of appearance. We then added 3 new features.

The first one, was an interaction term between the length of a text and the number of children the comment has. It stands to reason that a lengthy comment might incite a reaction from viewers or turn some readers who are not interested in reading a lot off. The way we separate these two types of comments, is to have a measure of how responsive the viewers are to a particular comment size.

The second feature is an interaction term between is_root and the number of children a comment has. This allows us to weigh root comments with more responses differently than root comments without. It makes intuitive sense that popularity of a root comment is linked to the number of replies it gets.

Similarly, the last feature is an interaction feature between is_root and the controversiality of a comment. The reasoning behind this is that controversial comments tend that spark controversial debate should be weight MUCH more differently than those that merely participate.

# 4 Results

We found that the run-time performance of linear regression with closed-form solution is better than gradient descent. Also, when using gradient descend approach, the run-time performance was noticed to be better with smaller learning rate. We also experimented with different initial values of weights and found the resulting models very comparable. Based on the RMSE of training, validation and testing datasets, we do not notice any evidence of underfitting or overfitting. The exact numbers are presented below.

| Time execution analysis | |
|---|---|
| Model | Execution time in seconds |
| Closed-form | 0.0097210 |
| Gradient Descent | 0.255780 |

Table 1: Comparison of execution times in seconds of closed-form solution and gradients descend. For gradient descend, $\epsilon = 0.2$ and $\beta = 100$

.

| Gradient Descent with different learning rates | | |
|---|---|---|
| $\beta$ | Execution Time | Root-Mean-Square Error |
| 50 | 0.254984 | 1.309606 |
| 100 | 0.133815 | 1.309656 |
| 150 | 0.092497 | 1.309705 |
| 200 | 0.072997 | 1.309755 |

Table 2: Comparison of execution times in seconds for different decaying rates. $\epsilon = 0.2$

.

| Gradient Descent with different weight initializations | | | |
|---|---|---|---|
| Initial value of all weights | No. of iterations until convergence | Training RMSE | Valid RMSE |
| 0.00 | 4462 | 1.7860598 | 1.7060090 |
| 0.01 | 2823 | 1.6889734 | 1.6058662 |
| 0.05 | 6909 | 1.9697627 | 1.9634538 |

Table 3: Comparison of performance of gradient descend with different initialization of weight vector

.

| Performance analysis of various models | | | | |
|---|---|---|---|---|
| Model | Training Set | Validaton Set | Testing Set | Runtime |
| Closed-Form Model No text feature | 1.1906784 | 1.1906784 | | 0.0003988 |
| Closed-Form Model Using top-60 words | 1.0298982 | 0.991667 | | 0.0076839 |
| Closed-Form Model Using top-160 words | 1.0236094 | 0.9975316 | | 0.0247149 |
| Closed-Form Model (Tfidf + 3 new features) | 1.0702291 | 0.9294961 | 1.0636469 | 0.0097210 |

Table 4: Comparison of RMSEs for models with different features. $\epsilon = 0.2$ and $\beta = 100$

.

# 5 Discussion and Conclusion

In summary, using the closed form solution, we were able to get the test RMSE of 1.0636469.

It would be very interesting to analyse the popularity of a comment if we can experiment with more data. For example, if we have a time range for when the scrubbing of these comments took place and we could find out what was popular in a specific time frame. Since Reddit is such a widely used application this, we could associate a popular theme of comments to a specific time range of the internet. Furthermore, if we obtain data ranging multiple years, we could trace the journey of the general human minds throughout the years and, maybe eventually, we would be able to estimate the next step in a way similar to Szabo and Huberman's manupulation of the comments on the webste Digg [4]. In their research, they were able to accurately predict the popularity of a given comment. This of course leads to a slew of ethical discussions as people often get unsettled when "computers" predict things they deem personal.

# 6 Statement of Contributions

**Jean Djamen -** Task 1: Pre-processing data, developing new features

**Hasnain Mamdani -** Task 2: Implementing gradient descent and closed form solution. Experimenting with initialization of different weight vectors

**Paniz Bertsch -** Task 3: Run experiments, Tfidf feature extraction

# 7 References

*[1] B. Yu, M. Chen, L. Kwok, Toward predicting popularity of social marketing messages, Lecture Notes in Computer Science 6589 (2011) 317–324.*

*[2] A. Tatar, P. Antoniadis and M. Dias de Amorim, "Predicting the popularity of online articlesbased on user comments," in Proc. of WIMS'11, Norway, May 2011.*

*[3] https://scikit-learn.org/stable/modules/feature$_e$xtraction.html*

*[4]G. Szabo and B. A. Huberman. Predicting the popularity of online content. ArXiv e-prints, Nov 2008.*