COMP 551 Mini-Project 1

Group 118
Eric leung, ID 260720788
Ingrid Feng, ID 260803777
Donovan Chiazzese ID 260742780

I. Abstract

According to wikipedia, Reddit is a social media platform for news aggregation, web content rating, and discussion. Members can submit content in the form of links, text posts, and images, which are then "upvoted" or "downvoted" by other members. In this project, we implemented and investigated the performance of linear regression model to predict the popularity of a comment on reddit.

There are multiple factors that contribute to the popularity of a comment, including the content of the comment, and also whether there are many subposts, whether the comment is a "root" (it is not a subpost to any comment), etc. Our goal is to investigate the relationship between these features to the popularity of a comment. Consequently, we would be able to predict a comment's popularity based on our linear regression model.

We found that the children squared feature improved the performance the most and that the gradient descent approach was slower than the closed-form approach.

II. Introduction

In mini-project 1, we implemented a linear regression model to test the performance of least squares and gradient descent on predicting Reddit comment popularity. A dataset with 12,000 data points were used. Among those, 10,000 data points are for the training set, 1,000 data points are for the validation set, and 1,000 data points are for the test set. Each data point is stored as a dictionary with five (key, value) pairs: text, is_root, controversiality, children, and popularity_score.

Initially, we found that the top 60 words model was reasonably fitting with the features provided in the dataset. And we then proceeded to improve upon this model by adding our own features. We used Ablative analysis, a process by which components are removed one at a time, to determine which features were the least and most helpful.

A common hurdle in this type of investigation, as also met by by Jordan Segall and Alex Zamoshchin in their Paper Predicting Reddit Post Popularity is overfitting. A typical method of reducing overfitting is increasing the size of the dataset. In their case, matlab quickly ran out of memory, but we did not have such luxuries and improvised by using ridge regression by scaling all the features from N(0,1) at the expense of some error.

After that, we carefully tried out different feature combinations, with or without the newly added features. We found that with the new features children squared, controversiality squared, number of words in the comment, and number of repeated words, the performance is the best.

III. Dataset

For the data set, since it is given as a list object, we split our data set into three parts. Training set was obtained by slicing the first 10,000 data points (ie. data[0:10000]), as for the validation set contained the next 1,000 data points, and the test set stored the rest of the data points. In addition, we implemented a function called **lowerCaseAndSplit**, which takes in a string (in this case: a comment), lower-casing all the words in the input comment and split the words into a list. And then we iterate through each datapoint, and store the features into a matrix where each row is a datapoint and each column is a feature.

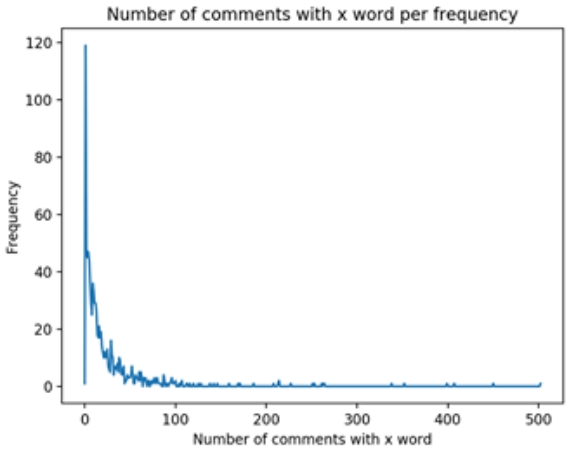
Apart from the features from the dataset, we implemented several other features:

- Length of comment in terms of letters:
  The idea is to measure the actual length of a comment, including white space, punctuation marks, etc. This feature is extracted using the **list()** and **len()** function. We thought that there should be a certain range of length that makes a comment popular. However, to our surprise, this feature actually increased the error.

- Number of repeated words:
  The idea is to measure the number of words repeated and we think that the more repetition there is, the more popular the comment is going to be. As we iterate through each function, this feature is extracted by calculating the different in length between **regular** and **unique**, which are produced by calling the function **splitNoRepitition()**, which first remove all the punctuation marks and then return two variables: **regular** (list of words with repetition), **unique** (list of words with no repetition). It turns out that this feature improved our performance in the validation set by 0.02494527.

- Number of components:
  A component is a series of words with no punctuation marks. For example, in "Hello World! Nice to meet you," there are two components. The intuition is that the more components there are, the less likely that a comment is popular. We implemented **splitIntoComponents()** to form components into a list for each component and then we use the **len()** function to measure the number of components. However, this feature also increased the error.

- Number of words per comment:
  We count the number of words, not including punctuation marks, using the **len()** function. And then this feature is implemented as 1/number of words + 1(to avoid dividing by 0). Our intuition was along the same logic as finding the most frequent words, we tend to write similarly as others, we'd call this human bias. This intuition was correct and plotting the frequency of the length of a comment from 0 to the longest comment in our dataset, we'd likely to get an inverse function. This feature actually improved the performance in the validation set by 0.01322389.



- Children squared:
  A technique we saw in class was adding a feature as a function of another feature. Our intuition was that the relationship between popularity and the more replies (children) a comment had was not linear, but possibly squared. We investigated this by adding a feature which was the squared value of the children of that comment.

- Controversiality squared:
  Same as the children squared, we added this feature since we think that the relationship between the popularity and the controversiality may not be linear.

Some of the ethical concern we find is that when working with a public social media dataset of this variety, we might be exposed to some users' personal information, such as their cell phone number, personal emails or addresses, and we shall not collect these kinds of data since it is not ethical to store one's private information.

IV. Results

On average, the runtime of the Least Squares solution was approx. 100-1000x quicker than Gradient Descent solution and remained this proportion while the complexity grew as new features were added, specifically with our chosen hyperparameters. We noticed that if we reduced the learning rate we would sacrifice some error in exchange for a faster runTime. Given time was not an issue, we decided on an alpha of 1e-6 and epsilon of 1e-5 and got results precisely close to those of the closed form solution. They are neatly printed side by side as the output in the code.

TABLE I

NO TEXT FEATURES

	Training Set	Validation Set	Test Set
MSE	1.08468307	1.01950474	1.26727408
RMSE	0.66666809	0.64891124	0.71598755
MAE	0.45431822	2.80080974	1.37993142

TABLE II

TOP 60 TEXT FEATURES

	Training Set	Validation Set	Test Set
MSE	1.06042914	0.91996816	1.16579676
RMSE	0.66428632	0.63882311	0.70659955
MAE	0.47355486	2.78070384	1.32020565

TABLE III

TOP 160 TEXT FEATURES

	Training Set	Validation Set	Test Set
MSE	1.04777632	0.77858701	1.05171032
RMSE	0.66378418	0.6051315	0.67607077
MAE	0.46543812	3.22425161	1.35824873

Moreover, we compared a model with no text features, a model that uses top 60 words, and a model that uses the full 160 words using the closed-form approach. We found that there is underfitting for the model with no text features and there is overfitting for the model with 160 words.

In our best performance model, we used four extra features: children squared, controversiality squared, number of words per comment, and number of repeated words.

TABLE IV

BEST PERFORMANCE MODEL

	Training Set	Validation Set	Test Set
MSE	1.01342523	0.8804665	1.15810591
RMSE	0.65547734	0.62734432	0.70643119
MAE	0.42330435	3.33143631	1.31473791

V. DISCUSSION AND CONCLUSION

Throughout the project, we have successfully demonstrated and utilized the Linear Regression Algorithm (Least-Square, Gradient descent). Moreover, all three of us has had our first exposure to Machine Learning. In conclusion, it was educational experience for all of us, and we really enjoyed brainstorming for new features that can be applied to our model as we have come up with four more bonus features at the end.

STATEMENT OF CONTRIBUTIONS

Task 1 was done mostly by Eric, Task 2 was done mostly by Donovan, and Task 3 was done mostly by Ingrid. And we completed the write-up together.