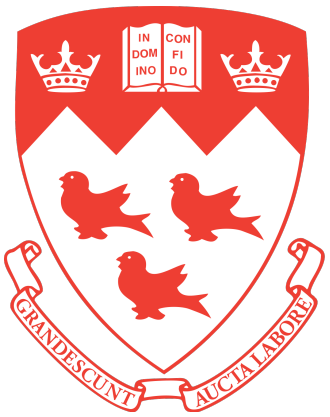

COMP 551 - Applied Machine Learning
Mini Project 1 Report

Matthew Lesko-Krleza
260692352

Sam Cleland
260675996

Wuyang Zheng
260830900

January 31, 2019

Contents

1 Abstract

In applied machine learning, linear regression is a basic method to model supervised learning prediction problems. The goal of this project is to predict the popularity of posts from Reddit, a popular social news website, using linear regression. Given a data set consisting of Reddit posts along with characteristics about the post, linear regression is implemented in two ways, through the closed-form solution and through gradient descent. Later shown in the Results section, experimentation with both these methods showed that mean square error (MSE) of the closed form solution was always less than MSE of gradient descent along with closed form running faster and it being much more stable. Some features we observed influenced popularity score significantly which include the number of replies (children), square value of the number of replies, cube value of the number of replies and number of occurrences of most frequently occurring words in the training set. There are also some features don't decrease MSE a lot which include if a post contains a question mark and exclamation point (2 separate features) which are Boolean type features, and sentiment score of these comments (using NLTK library). However, there shouldn't be a lot of features in the model to prevent overfitting. The software used in this report is Python 3.

Keywords: applied machine learning, closed-form linear regression, gradient descent linear regression, popularity prediction, Reddit

2 Introduction

Reddit is a website where users can post content and other people can view this content and react to it if they choose via an upvote or downvote system. In this task, the purpose is to design a useful model to predict how popular a post will be given a data set of Reddit posts (predicting the popularity score of the post in this case). The data set is provided as a list of data points, each data point is a dictionary which contains "popularity_score", "children", "controversiality", "is_root" and "text". This report will represent the two methods of developing these models which are the closed-form solution and gradient descent. The main body of this report is a description of the data set given, the results of many experiments with different features and models, discussion and conclusion, and statement of contribution.

Important information shown from experiments are that the closed-form solution is easy to implement and the running time is much shorter since the data set given is not very large. Also, on average it had a smaller MSE than gradient descent since the weighting coefficient of closed form model is the minimum difference between predicted score and true score as opposed to the gradient descent's gradual approximation to the minimum of the error, which in practice doesn't equal the minimum exactly.

Some factors (i.e. features) effect the popularity of the posts, so analysis on how influential these features are on the popularity score is important for successful predictions. There are several features that reduced the MSE which are "children", "children_square", "children_cube" and "160 top word count", while "question mark and exclamation point", "sentiment score" are not very related to the popularity score. "children" is the number of replies to a post and from experimentation it had a strong correlation with popularity score. When we explored the squared and cubed versions of children, "children_square" and "children_cube", resulted in a similar strong correlation. "160 top word count" is the number of occurrences of the most frequent 160 words overall comments in the training set. "question mark and exclamation point" and "sentiment score" did not significantly change the MSE indicating that popularity score has no close relation to posts.

3 Dataset

3.1 Summary

For this assignment we were given 12000 samples of posts from reddit.com. The information associated with the posts included if a post was considered controversial, if a post was the root of a thread (initial post), how many child posts it had and the sentence(s) associated with the post. This information was used in order to predict the popularity score of the post.

3.2 Training

Before training the model, the data was partitioned into a training, validation and test set. The first 10,000 samples were associated with the training set, the next 1000 with the validation set and the last 1000 for our test set.

3.3 Features

The features that were used for the model include controversial, root, the number of children post associated with that post and the number of instances of the most frequent 160 words (each word considered as its own feature). 2 new features were implemented in the model to reduce the model's mean squared error on the validation set. The first feature implemented was a square of the children feature. After analysis of the correlation of certain features to the popularity score, it was found that the children featured had the highest correlation amongst all other features. When the squared version of the children feature was added to the model, mean squared error reduced significantly on the validation set (approximately by 0.2). For our second feature we also included another variant of the children with a children cubed feature, which also led to significant improvements in MSE.

We initially had ideas for other features rather than using the children cubed feature. Two of these brainstormed features were boolean features (similar to controversial and root) which are set when the sentence has an exclamation point or question mark symbol (when a sentence has an exclamation point, set this feature to 1. Same procedure for question mark). Both of these features led to minimal reduction in MSE and had to be discarded. We also tried to use sentiment analysis to see if there was a correlation between how the negative or positive tone of a sentence was correlated with its popularity score, but it ended up not working well because the algorithm leaves out the context of the post. A post could be very negative, perhaps someone complaining about something, and can be very well received from other people that agree or not well received from people that disagree. The sentiment could be very negative in this case but the understanding of the post is missing with sentiment analysis. From this observation, we then tried a variation of a sentiment analysis in which we didn't care about if a post was necessarily negative or positive but the magnitude of the emotion. This did end up improving the model slightly, but the main problem with this approach is that you can have neutral sentiment sentences that are very popular or unpopular from the context of the post. Because of these drawbacks we ended up leaving this feature out of the final model. The results of these experiments can be seen in Table 5 in the appendices.

3.4 Ethics

There were no real ethical concerns with the data set we were given as there was no sensitive information associated with people in the posts. However, if we had sensitive information there would have to be consent the person associated with that data in order to proceed. Without the consent, there could be massive legal and financial implications. An example portraying this includes the Facebook-Cambridge Analytica data scandal where sensitive information from Facebook users was used without consent by the users. This information was then used by Cambridge Analytica to increase the popularity of certain political candidates. This had a severely negative impact on Facebook, plummeting their stock by 100 billion dollars within days of the news breaking out along with more issues with the government over security. It is important to understand not only the data used but also that the implications of using the data, especially if the source of the data is not from oneself nor anonymously posted.

4 Results

4.1 Comparison between Gradient Descent and Closed-Form Approaches

The experiments done for this particular task, which involved popularity score prediction, only used the three simple features provided: children, is_root, and controversiality.

4.1.1 Runtime and Performance

As seen within our Jupyter notebook, the closed-form solution took 0.000958 seconds to complete weight estimate computation on the whole training dataset and had a mean squared error of 1.020327 on the validation set. Whereas, the gradient descent approach was tested with 5 different sets of parameters. We can compare the first trial in Table 1 with the rest of the trials to highlight the effects of changing certain parameters. For brevity purposes, we'll only consider the final trial that used the parameters that improved performance across previous trials.

In the final trial, the performance was of an MSE of 1.037743, however, it also had the longest train time of 1.361360 seconds. This highlights the fact that with gradient descent, it may take a significantly longer time to compute optimal weight estimates unless we somehow knew of and used the 'goldilocks' step size or something near such a value.

To summarize, given our dataset, the closed-form solution computed its weight estimates faster and performed better than its gradient descent counterpart. The closed-form solution had a runtime of 0.000958 seconds and a mean squared error of 1.020327 whereas the best performing gradient descent model had a runtime of 1.361360 seconds and an MSE of 1.037743. However, in theory, when using larger datasets (in at least orders of 100'000s), gradient descent should preferably be used because the closed-form solution would take far too long to compute.

4.1.2 Stability

Considering that the closed-form approach doesn't have a set of parameters, it will always yield the same weight estimates when given the same input training data and training labels. We therefore tested the closed-form approach's stability by having it compute weight estimates over different partitions of the training dataset. The results can be seen in Table 2. For reference, we've also added the case where the whole training set was used.

It can be noted that for the majority of the trials, the closed-form solution approach's performance is around 1.022, except in the particular case where it reached 1.049841 in the 5th trial. This outlier could be explained by the fact that the training dataset doesn't have an equal distribution of popularity scores across its entirety.

We can inferentially deduce that when trained on significantly smaller portions of a dataset, the closed-form solution approach still remains stable. In contrast with the closed-form approach, the gradient descent approach can be unstable, as seen in the case with an initial step size of 0.000001 where its MSE is of 2.346012. If a machine learning designer isn't careful with parameter initialization, performance can significantly worsen, train time can significantly increase without any improvement on performance, or in some cases when the initial step size is way off, the gradient descent process will never converge (as we've seen first hand when testing our gradient descent implementation).

4.2 MSE Performance on No Text Features, Top-60-Words, and Top-160-Words

For this task, we used the closed-form solution approach. The results can be seen in Table 3.

It can be seen that as more top words are selected as features, the model's performance improves slightly. Upon further analysis of the correlation heatmap in Figure 1, we could see that the top-word features are very weakly correlated to popularity score. The most correlated top-word feature is only correlated to popularity score by a factor of 0.07. This can strongly suggest that as we're including more top-word features to our dataset that we're unnecessarily increasing the complexity of our model and the dimensionality of our feature space, thereby overfitting our model. In the case of the model without any text features, we can see that its MSE is of 1.020327 and lacks any strongly correlated or a large number of mildly correlated features with popularity score, as seen in Figure 1.

4.3 New Features Performance Improvement

The two new features are the following: 'Children Squared' and 'Children Cubed.' The two features were tested alongside the Top-60-Words, is_root, children, and controversiality features. Three new trials were run. One with the first new feature, another with the second new feature, and finally with both new features. The model's performance with none of the new features is added to the list of trials for comparison. The runtime and performance results of the experiment can be seen within Table 4.

The children squared and children cubed features separately lead to noticeable decreases of 0.027540 and 0.027171 in MSE respectively. However, as we've seen in Figure 2, children squared and children cubed are strongly correlated to one another, therefore, using both of the features doesn't lead to a significant improvement in MSE when compared to using only one of them. Finally, when the model with children squared and top 60 words is tested on the test set, it has an MSE of 1.258713. This clearly shows that our model is overfitting. It has a significantly lower validation MSE when compared to the test MSE.

5 Discussion and Conclusion

There are several key takeaways for the differences in performance, runtime, and stability between the closed-form solution approach and the gradient descent approach. A machine learning designer must experiment and test for different sets of parameters for gradient descent because their impacts on performance and runtime can vary greatly from one another, making it an unstable model. Depending on the size and number of features fed as input for gradient descent, a designer needs to tune the various parameters. Closed-for solution is stable, and performs identically when given the same inputs. Given that the amount of data given to us is small, the closed-form solution ran quicker than gradient descent.

When it comes to using text features and coming up with new features, we must make sure to not unnecessarily increase the dimensionality of our feature space, as it can lead to overfitting and over complicated models as we've seen in the case of adding the top-160-words features. We've also seen that we really need to 'think outside of the box' to engineer and select new features that could help develop higher performing models.

6 Statement of Contributions

Matthew worked on the processing and encoding of data, worked on the gradient descent implementation, experimented with the closed form solution and gradient descent models, merged code, and wrote the results, discussion, and conclusion sections. Sam worked on gradient descent implementation, experimentation with new features, revised abstract and introduction section and wrote dataset section. Wuyang worked on the closed-form implementation and experimentation with new features, and write the abstraction and introduction sections.

7 Appendices

Initial Step Size	Decay Factor	Error Threshold	Train Time (Seconds)	Performance (MSE)
0.001	10	0.0001	0.008976	1.055071
0.00001	10	0.0001	0.006981	2.346012
0.001	5	0.0001	0.019947	1.045079
0.001	10	0.000001	0.513627	1.048162
0.001	5	0.000001	1.361360	1.037743

Table 1: Gradient Descent Parameter Variation Experiment Results

Slice of Training Dataset (Indices)	Train Time (Seconds)	Performance (MSE)
0 to 10000	0.000958	1.020327
0 to 2499	0.000000	1.021670
2500 to 4999	0.000998	1.023137
5000 to 7499	0.000998	1.020025
7500 to 10000	0.000000	1.049841
0 to 4999	0.000993	1.020826
5000 to 10000	0.000997	1.023606

Table 2: Closed-Form Solution Training Partition Dataset Variation Experiment Results

Features Selected	Train Time (Seconds)	Performance (MSE)
No Text Features	0.000998	1.020327
With Top-60-Words	0.001996	0.983186
With Top-160-Words	0.005984	0.989536

Table 3: Closed-Form Solution Text Feature Experiment Results

New Features Added	Train Time (seconds)	Performance (MSE)	Decrease in MSE
None	0.001996	0.983186	0
Children Squared	0.002963	0.961996	0.027540
Children Cubed	0.001996	0.962365	0.027171
Children Squared and Cubed	0.001995	0.962446	0.027090

Table 4: Closed-Form Solution New Features Experiment Results

New Features Added	Train Time (seconds)	Performance (MSE)	Decrease in MSE
None	0.001996	0.983186	0
Exclamation	0.005510	0.983233	-0.000047
Question Mark	0.005198	0.983233	-0.000047
Sentiment	0.004723	0.983195	-0.000009

Table 5: Closed-Form Solution Brainstormed Features Experiment Results

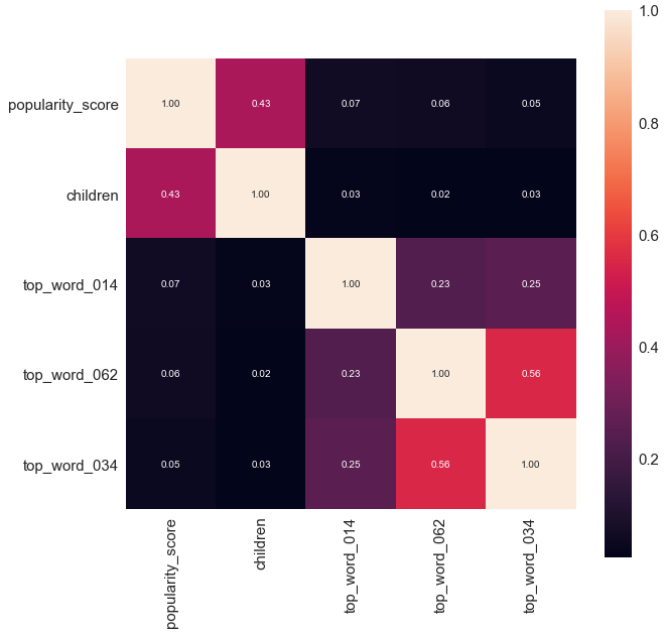

Figure 1: Correlation Heatmap of Popularity Score and Top 5 Correlated Features

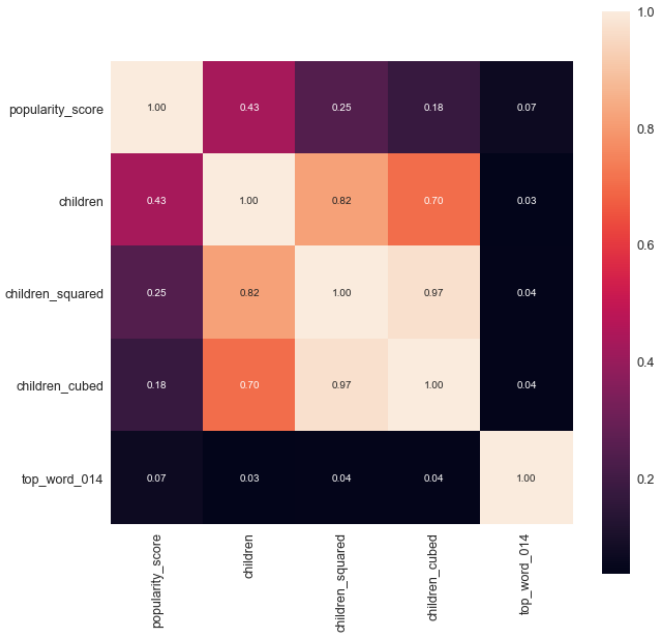

Figure 2: Correlation Heatmap of Popularity Score with New Features