Are your Reddit posts popular?

COMP 551

Xavier Sumba (260900337) Ameya Bhope (260849407)
Jaume Miñano (260874194)

January 2019

Abstract

The objective of the project was to explore the performance of the linear regression model for predicting the comment popularity on the social networking website Reddit. Higher the popularity of the comment more prominently it is featured. The target of the mini-project was to find the relation between certain features such as controversiality, whether the comment has replies to it, or a number of text features such as the most common words, the sentiment of the comment to the popularity of the comment. For this we implemented linear regression in its closed form solution, with gradient descent and decay, with gradient descent and momentum, to find that the closed form solution almost always gives the best performing model, in the quickest runtime. We achieved a MSE of 0.948 in the validation set and 1.256 in the test set.

1 Introduction

In this project, we compare linear regression models for predicting comment popularity on Reddit. We found that working with real data is challenging due the heterogenous and statistically erratic nature of data.

Previous research has treated popularity prediction. For example, in [2], the author treats popularity prediction as a text classification problem by identifying topics and then classifying them. He uses Latent Dirichlet Allocation and TF-IDF. We have been inspired by this work and we have applied the TF-IDF to build a new feature. Next, [3] also worked on the prediction of Reddit Post Popularity. The focus of their research was to analyze the post's content as well as other factors. Their work included features such as the title of the post, the subreddit of the post, and the time of day the post was created.

In our work, the features on which the prediction of comment popularity was to be done based on the number of replies for each comment, controversiality, differentiation of post and comments and the raw text. Taking these into consideration we derive many features such as the powers of the feature children, or the sentimental analysis, most frequently used words in the comment, length of the comment for the text feature.

We attempted to predict the comment popularity by implementing the linear regression model. The linear regression model was implemented by using the closed form solution and also the gradient descent method. The gradient descent method was implemented in two ways — decaying learning rate and with momentum for faster upgrading of the learning rate. As expected we found that the closed-form solution gives us the least validation dataset error and tuning the hyperparameters accurately, we could get the weights such that the error for the gradient descent closely approximated that of the closed-form solution.

In this report, we describe the dataset and its features in detail, and explore some insights about the data in section 2. Next, we summarize the model selection and give results in section 3. Section 4 summarize our key takeaways from this project and conclusions.

2 Dataset

The dataset used for this project is a curated dataset from Reddit. For instance, Reddit is organized in communities, also called subreddits, where users can create and comment in posts; the subreddit used is *r/AskReddit*. The dataset contains $12,000$ instances which were split in the following order $10,000$, $1,000$, and $1,000$ for training, validation, and testing respectively. The target variable is the *popularity_score*, which is a measurement of how popular a post is. The table 1 illustrates all the features [1].

Id	Feature	Description
F1*	children	number of comments a post have received.
F2*	controversiality	indicates whether a post is controversial or not
F3*	is_root	true if it is the original post otherwise is a comment
F4*	text	text of the comment
F5	tfidf	weight of words according to its importance
F6	feeling	select only words that belong to a dictionary of emotions
F7	$children^2$	F1 (children) squared
F8	len_text	number of characters
F9	len_sentence	number of sentences
F10	sentiment_{neg,neu,pos}	negative, neutral, and positive score of text (F4)

Table 1: Features

By looking at the correlation between the main features, we observed that the children feature has a high correlation with the popularity of the comment[2]. Hence, model a combination of the variable *children* were made. Many features were derived from the raw text of the comments. In one of the features, we tried to find the most important words in a comment by TFIDF. Another feature was developed with a comparison of the text with the feelings dictionary. In another feature, we computed the length of the characters in the comment. Similarly, a feature which showed the number of sentences in the text feature. In another class of features, we performed a sentimental analysis of the raw text, with one being the negative, neutral or positive score.

While working with the real world messy data, it is important to note the ethical implications. The privacy of the user should not be breached in any circumstance. Also, in such a platform controversial, unchecked statements might be considered as facts resulting in provoking certain individuals.

3 Results

In this section, we make an analyses of the gradient descent approach in terms of runtime, stability, and performance and compare it the closed-form solution. In addition, we summarize the models on which we have obtained the best results. Finally, we report our results by executing the two top models in the testing set.

[1] Features with a '*' are the ones that come by default in the dataset while the other ones are generated.

[2] Some graphs and analyses were avoided because of the limitation of space, but they can be found in the Jupyter notebook provided.

3.1 Analyses of gradient descent and closed-form solution approaches

Since linear models are convex, when the model reach convergence, the optimizer will reach a maximum. Specifically, in the case of the closed-form approach, we will get an exact solution. However, if the data is not i.i.d., as in some real world applications, the problem won't be solvable because the matrix is not invertible. This can be shown in the reddit dataset when the number of bag of words is greater than 160. This problem can be solved by using gradient descent.

In this work, we are dealing with a small dataset ($10,000$ instances) in which the matrix inversion and multiplication is computable. But this might not be the case for bigger datasets. Some variations of gradient descent [1] (i.e. stochastic gradient descent) can scale to big datasets.

However, gradient descent has its own drawbacks. For instance, it is susceptible to hyperparameter tuning. The selection of hyperparameters can affect the solution in terms of time and precision. Figure 1 illustrates a comparison using features *F1, F2, and F3.* Gradient descent is executed with a momentum rate of 0.9 that seemed to work well for this combination of features. The graph on the left shows the time to compute using gradient descent and closed-form approach. We can notice that a really small learning rate will take too long compared with the time that the closed-form takes, and the appropriate selection of the learning rate can lead to a runtime as close as the one provided by the closed-form. In the same way, the graph on the right shows how close are the approximate weights computed with gradient descent compared with the exact solution provided by the closed-form approach. For instance, in this model, the learning rates of 0.1 and 0.01 give weights that are near to the weights of the closed-form with a decimal precision of $1e-7$. So, the results of the closed form will be unique, but the result of the gradient descent approach will vary according to the hyperparameters selected.

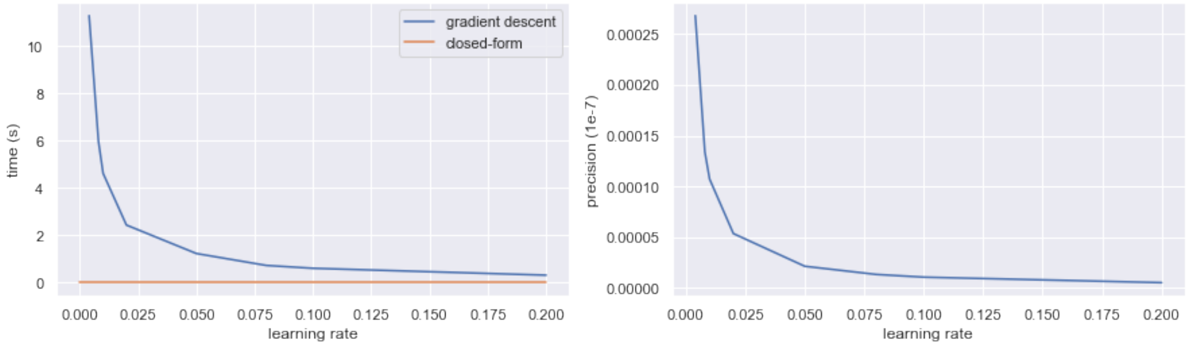

Figure 1: (Left) time complexity. (Right) difference of weights.

3.2 Model selection and testing

We executed a variety of models varying the hyperparameters (i.e. learning rate, momentum rate, weight decay, and initialization)[3], and combining different sets of features from table 1. The closed-form approach is usually faster and gives an exact solution. In addition, as discussed in section 3.1, the results of gradient descent vary according to the selection of hyperparameters. We have found that momentum generally converges faster than weight decay. And models don't seem to be affected about the initialization (zeros or random) carried out. We trained the models with a tolerance (ϵ) of $1e-5$ and $1e-7$.

[3] Results of all experiments can be found in *results\all_data.csv*

Table 2 shows the top 8 models[4]. In general, the closed-form approach gives the lowest error in the validation set. We obtain a MSE of 0.949 in the validation set by combining the two extra features. Obtaining an improvement of 0.04. We can notice that each extra feature gives an improvement of 0.02.

Features	MSE Train	MSE Validation	Method
F1,F2,F3*	1.084683071	1.020326685	Closed-form
F1,F2,F3, F4 (top 60)*	1.060229826	0.98318597	Closed-form
F1,F2,F3, F4 (top 160)*	1.046832912	0.9895357	Closed-form
F1,F2,F3,F4 (top 62),F8	1.058810088	0.969466625	Closed-form
F1,F2,F3,F4 (top 57),F7	1.013686756	0.961392588	Closed-form
F1,F2,F3,F4 (top 57),F7,F8	1.013596497	0.962265642	Closed-form
F1,F2,F3,F4 (top 60),F7,F8	**1.012119260**	**0.948822589**	Closed-form
F1,F2,F3,F4 (top 62),F7,F8	1.013339017	0.962771099	Closed-form

Table 2: Top 8 models executed with the closed-form approach.

We selected the top-two models of table 2 to execute with the test set. We notice that the test error of our best model does not perform as well as the second best model. However, both models might be slightly overfitting since the test error is greater than the validation error. We obtained a MSE of 1.267798 and 1.256074 for best model and second best model respectively.

4 Discussion and Conclusion

Our results indicate that the model might be overfitting. To combat this as a future improvement we can implement a regularized regression algorithm (L2 or L1). Another improvement we could do is dimensionality reduction, which takes care of the multi-collinearity, removes redundant features thus improving the performance of the model. Finally, we find that the variance of the data is high and hence implementing a cross validation strategy might help avoid the overfitting and test the stability of the model.

In this project, we analyze gradient descent and the closed form solution for gradient descent and build a model for the prediction of popularity in a subreddit, achieving good results in the validation set.

5 Statement of Contributions

We all three contributed to the project quite equally. The main framework of the linear regression was written by Xavier. And we all contributed to the pre-processing part and execution of experiments as well in the report.

References

[1] Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT*. Springer, 2010.

[2] Rohlin, T. Popularity prediction of reddit texts.

[3] Segall, J., and Zamoshchin, A. Predicting reddit post popularity.

[4] The models with a '*' are the three models required in this project.