Using Closed-Form Linear Regression and Gradient Descent to Predict the Popularity of Reddit Comments

Noam Rabbani
McGill University, Canada
noam.rabbani@mail.mcgill.ca

Yao Jiang
McGill University, Canada
yao.jiang@mail.mcgill.ca

Xingyu Xiang
McGill University, Canada
xingyu.xiang@mail.mcgill.ca

1 ABSTRACT

Using linear models can be a simple yet effective approach in tackling prediction problems. In this paper, we investigate the closed-form and gradient descent approaches to linear regression by using them on a dataset of 12 000 Reddit comments. Our goal is to compare the two methods and to generate a model that can predict the popularity of a Reddit comment. Our results show that the closed-form approach is faster and more stable than gradient descent, but that both methods result in similar performance. We then propose a set of 167 textual and numeric features that achieve a MSE of 0.9579 on our validation data. Finally, we test our model's ability to generalize and we report poorer performance on unseen data due to overfitting. We suggest an approach help avoid this problem and we propose ideas for future work.

2 INTRODUCTION

Closed-form and gradient descent are two approaches of linear regression that can be used to tackle prediction problems. In this paper, we investigate both approaches with the goal of creating a model that is able to predict the popularity score of comments on Reddit, a famous discussion website. To do so, we study a dataset of 12 000 Reddit comments that contain numeric and text features. The comments have been anonymized and do not contain the author's usernames, even though Overdorf and Greenstadt have shown that authorship on Reddit comments can be traced from text features alone [6]. We first investigate the differences between the closed-form and the gradient descent approaches. Then, we extract original features to improve the performance of our models and finally, we test our best model on previously unseen data.

We discover that gradient descent is slower and less stable than the closed-form approach, but that they both have similar performance. We then engineer three original features that reduce the mean square error (MSE) on our validation set by 0.0259. Finally, we realize that our model is overfitting, as it reports a significantly worse error on the test set than on the validation set. This is a common pitfall of machine learning [3] and we propose a method to help with this issue.

3 RESEARCH QUESTIONS

Our study aims to investigate four research questions relating to the closed-form and gradient descent approaches to linear regression.

- **RQ1:** How does the closed-form approach compare to gradient descent in terms of runtime, stability and performance?
- **RQ2:** What performance can we achieve when considering only a set of basic features?
- **RQ3:** Can we propose new features that will further improve performance?
- **RQ4:** How well does our model generalize?

4 DATASET AND STUDY DESIGN

We were provided a data set that contains 12 000 Reddit comments and some basic features associated with them. In this section, we describe the steps taken to preprocess the dataset, to extract features, to generate our models and to test them.

DS0: Preprocess raw text of comments. Because we derive features from text, we first need to preprocess the comments. We do so by applying the python functions lower() and split() on the raw text of every comment.

DS1: Split the dataset into three parts. We then partition the data into a training, a validation and a testing

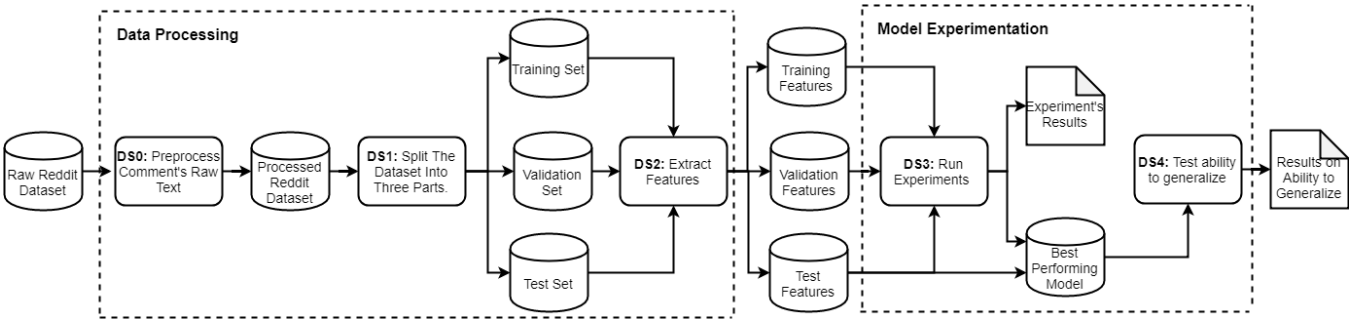

Figure 1: An overview of our approach to create and test models that predict the popularity score of Reddit comments

set. Training a model requires the most data, whereas validation and testing can be performed on smaller samples. For this reason, we use the first 10 000 entries for training, the following 1 000 for validation and the last 1 000 for testing.

DS2: Extract features. In addition to the three basic features provided in the data set (children, controversiality, is_root), we add a bias term that is used in every one of our models. We also generate a list of the most frequently occurring words in the data set and we derive 160 features from this list. In addition, we brainstorm five original features that we believe to have the potential to improve our model's performance. We also extract the popularity score which serves as the output of our predictions. Our five original features are:

- **word_count:** number of words in a comment.
- **avg_word_length:** average length of the words in a comment.
- **avg_words_per_sent:** average number of words per sentence in a comment. We performed extra text processing to be able to extract this feature. For our study, we decide that sentences can be separated by either a dot, an exclamation mark or a question mark.
- **sentiment:** score within the range [-1,+1] that gives a sentiment analysis on a comment. A score 0 indicates neutral, +1 indicates a very positive sentiment and -1 represents a very negative sentiment. This feature is computed using the *TextBlob* library [5].
- **readability:** score that represents the readability of a comment based on the Gunning fox index [2]. For example, texts intended for wide audience should have an index of less than 12 and texts requiring near-universal understanding need an index of less than 8. We use the *TextSTAT* library [4] to compute this metric.

We also consider the following feature interactions.

- **sentiment^2:** We add this quadratic feature to limit the difference between negative and positive sentiments.
- **avg_words_per_sent*sentiment^2:** We believe that a comment that contains long sentences and that expresses strong opinions will have a high popularity score.
- **readability*sentiment^2:** We believe that a comment that is well written and that shows a strong sentiment will have a high popularity score.

DS3: Run experiments. We then run a series of four experiments to help answer our research questions. We first test the runtime, stability and performance of both the closed-form and the gradient descent methods. Then, we investigate the performance of the closed-form by varying the number features used. We first test the basic features alone, then we test the models with top-0, top-60 and top-160 words. We select our best performing model and attempt to improve it with the original features we came up with. Finally, we choose the best performing model and run it on the test set.

It's important to note that we are working with a public data set and that some ethical aspects have to be considered [8]. For instance, are we infringing on the privacy of the Reddit users by using their comments in our study? Did they agree to release this data and if they did so, was it an informed decision? We however note that the data was somewhat anonymized [6] as we are not given usernames in the data set.

5 RESULTS

In this section, we present the results of our study with respect to the four research questions.

RQ1: How does the closed-form approach compare to gradient descent in terms of runtime, stability and performance?

Results. The closed-form approach has better runtime and stability, but similar performance to the gradient descent

We begin by studying the stability and convergence of gradient descent. We look into the impact of the initial condition and the learning rate on the performance of this model. Our results show that gradient descent remains unstable if the Robbins-Monroe conditions are not satisfied. The details of our results are presented in Tab. 2 of the Appendix. We observe that if the initial value is located at a given distance from the optimal, the descent diverges even when a small learning rate is used.

Previous work shows that gradient descent is unstable [1], but that we can improve it by using an adaptive learning rate.

$$\alpha_k = \alpha_0 \frac{1}{k+1}. \quad (1)$$

Tab. 3 in the Appendix shows that with this addition, our simulations always converge. However, we do obtain a slow convergence rate if the initial condition is far from the optimal and if the initial learning rate is small. Since we limit the number of iterations to 5000 in our study, several configurations are unable to reach the convergence condition $|err(w_{k+1}) - err(w_k)| < \epsilon$.

Table 1: Performance comparison of the closed-form and gradient descent approaches

	time (*ms*)	MSE_TRAIN	MSE_VALID
Scikit Reference	8.924	1.084683	1.020327
Closed-form	3.517	1.084683	1.020327
Grad descent	42.892	1.085613	1.022377

We then compare the closed-form approach with a gradient descent configuration that uses a distance of 0.3 as the initial condition and a learning rate of $\alpha_0 = 1$. A comparison of the running time and MSE obtained by these two approaches is shown in Tab. 1. We confirm that our technical implementations are correct because they obtain the same MSE value as the SciKit-learn library [7]. It is interesting to note that our implementation of the closed-form approach runs faster than the one from SciKit. Even though the gradient descent approach avoids the calculation of the matrix inverse, the large number of iterations make it slower than the closed-form approach. Fig. 2 shows the convergence of MSE versus the number of iterations. We observe that with the current configuration, the MSE quickly drops during the first few iterations and then stabilizes until convergence.

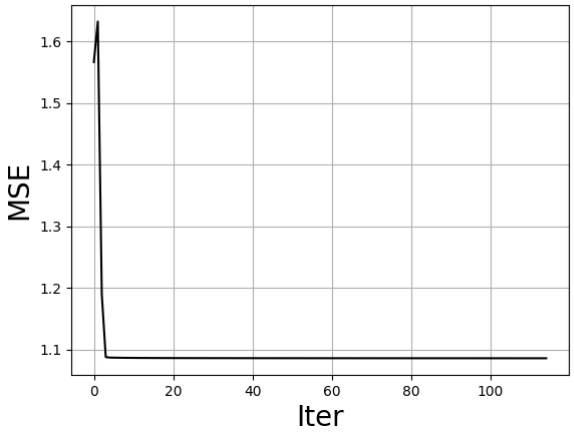

Figure 2: MSE convergence vs. iterations for the gradient descent approach

***Observation 1**: Gradient descent is slower and less stable than the closed-form, but they both achieve similar performance*

RQ2: What performance can we achieve when considering only a set of basic features?

Results. When running the closed-form model with three basic features, a bias term and 60 text features, we achieve a MSE of 1.0604 on the training set and 0.9839 on the validation set. For this experiment, we test three variations of the closed-form approach to find the best performer. The models use the same set of basic features and bias term, but vary in the number of text features. Our results indicate that the best performer is the model that uses 60 text features. We notice that the model that uses only the three basic features and a bias term is underfitting. Indeed, this

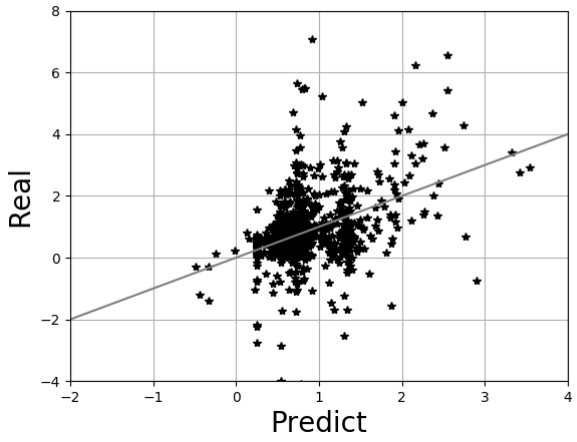

Figure 3: The predicted popularity *vs.* the real popularity

model reports the lowest performance of all because it is lacking information when making its predictions. Strangely enough, all three models report a better performance on the validation set than on the training set, so there is no clear case of overfitting at this point.

***Observation 2**: Our best performing model is the one that uses the three basic features, a bias term and 60 text features*

RQ3: Can we propose new features that will further improve performance?

Results. We find three features that, when combined, reduce the validation MSE to 0.9579 We test 16 combinations of the features proposed in Sec. 4 further improve our best-performing model. Unfortunately, most of our original features reduce the MSE by less than 0.005, which we consider too little to be significant. However, we keep three interesting features that do have a significant contribution:

- **children^2:** Improvement of 0.0213 on the validation MSE
- **nb_words_per_sent:** Improvement of 0.0039 on the validation MSE
- **avg_words_per_sent*sentiment^2:** Improvement of 0.0035 on the validation MSE

***Observation 3**: When combined, our three original features further improve our validation MSE by 0.0259*

RQ4: How well does our model generalize?

Results. Due to overfitting, the performance of our best model decreases on the test set We evaluate our newly improved model's ability to generalize by running it on the previously unseen test data. Unfortunately, our results show that the MSE goes up to 1.2524, thus indicating that our model is overfitting. Indeed, the 167 features that are used by our best performing model are more effective on the training and validation data than on the test data. However, we note that the basic model has an even higher MSE of 1.2876, which means our features still provide an improvement. Fig 3 Shows the predicted popularity score versus the real popularity score. The ideal case would be that all the points appear on the reference line $y = x$.

***Observation 4**: Our model is overfitting on the test data, but it still reduced the MSE by 0.0352 when compared to the model with no original features.*

6 DISCUSSION AND CONCLUSION

The closed-form method and gradient descent are two effective approaches of linear regression to tackle prediction problems. With our study, we make the following observations:

- The closed-form approach is faster and more stable than gradient descent, but both have similar performance.
- Our best model uses a total of 167 features to achieve a MSE of 0.9579 on our validation data.
- This model however suffers from overfitting, as the MSE goes up to 1.2524 on the test data.

Our suggestion for future work is to implement k-fold cross validation to our methodology. We believe that our extensive testing resulted in overfitting on the validation set. Cross validation could help in solving this problem by averaging the prediction error over k folds. Finally, it would be interesting to see if our model is able to generalize to comments on different platforms, such as StackOverflow or YouTube.

7 STATEMENT OF CONTRIBUTIONS

The workload was split evenly between Yao and Noam. Xingyu attended some of the team meetings.

APPENDIX

Table 2: Steps of convergence for different initial condition and learning rate using fixed learning rate

Dist/ Rate	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0.3	337	244	193	160	div	div	div	div
0.6	637	394	293	235	div	div	div	div
0.9	813	482	351	279	div	div	div	div
1.2	937	544	392	310	div	div	div	div
1.5	1034	592	424	334	div	div	div	div
1.8	1113	632	451	354	div	div	div	div
2.1	1180	665	473	370	div	div	div	div
2.4	1237	694	492	385	div	div	div	div

Table 3: Steps of convergence for different initial condition and learning rate using adaptive learning rate

Dist/ Rate	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0.3	2106	580	313	237	204	180	160	142
0.6	4999	1419	834	669	567	485	418	366
0.9	4999	2478	1546	1249	1040	874	746	650
1.2	4999	3741	2430	1955	1605	1336	1136	991
1.5	4999	4999	3469	2773	2253	1863	1582	1386
1.8	4999	4999	4654	3694	2978	2452	2083	1833
2.1	4999	4999	4999	4711	3773	3099	2636	2331
2.4	4999	4999	4999	4999	4636	3802	3240	2880

Table 4: MSE error for different initial condition and learning rate using adaptive learning rate

Dist/ α_0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0.3	0.007	0.003	0.002	0.002	0.001	0.001	0.001	0.001
0.6	0.021	0.009	0.007	0.006	0.005	0.004	0.004	0.004
0.9	0.048	0.019	0.014	0.012	0.01	0.009	0.008	0.008
1.2	0.085	0.032	0.024	0.02	0.017	0.016	0.014	0.014
1.5	0.132	0.048	0.036	0.03	0.026	0.024	0.022	0.021
1.8	0.191	0.069	0.05	0.042	0.037	0.033	0.031	0.029
2.1	0.259	0.094	0.068	0.055	0.049	0.045	0.042	0.04
2.4	0.339	0.123	0.089	0.072	0.063	0.057	0.054	0.051

Table 5: Tests of linear regression using different features and their MSE on the training, validation and test sets

Original features	MSE_TRAIN	MSE_VALID	MSE_TEST
children^2	1.013781	0.962639	1.259064
word_count	1.06041	0.983454	-
avg_word_length	1.060404	0.9837	-
nb_words_per_sent	1.058671	0.980057	1.280437
sentiment	1.060332	0.985833	-
readability	1.060317	0.98339	-
nb_words_per_sent *sentiment^2	1.05994	0.980313	1.287555
word_count^2	1.059746	0.985085	-
avg_word_length^2	1.060416	0.984037	-
nb_words_per_sent^2	1.059551	0.982399	-
sentiment^2	1.060392	0.982733	-
readability^2	1.0596	0.984423	-
children^2, nb_words_per_sent	1.012257	0.95952	-
children^2, nb_words_per_sent, nb_words_per_sent *sentiment^2	1.012136	0.958006	1.252383

REFERENCES

[1] Vitaly Bushaev. 2019. Improving the way we work with learning rate. (Jan. 2019). https://techburst.io/improving-the-way-we-work-with-learning-rate-5e99554f163b

[2] Robert Gunning. 1952. The technique of clear writing. (1952).

[3] Douglas M Hawkins. 2004. The problem of overfitting. *Journal of chemical information and computer sciences* 44, 1 (2004), 1–12.

[4] Matthias Hüning. 2005. TextStat Simple text analysis tool. *Dutch Linguistics, Free University of Berlin, Berlin* (2005).

[5] Steven Loria, P Keen, M Honnibal, R Yankovsky, D Karesh, E Dempsey, et al. 2014. Textblob: simplified text processing. *Secondary TextBlob: Simplified Text Processing* (2014).

[6] Rebekah Overdorf and Rachel Greenstadt. 2016. Blogs, twitter feeds, and reddit comments: Cross-domain authorship attribution. *Proceedings on Privacy Enhancing Technologies* 2016, 3 (2016), 155–171.

[7] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[8] Jessica Vitak, Katie Shilton, and Zahra Ashktorab. 2016. Beyond the Belmont principles: Ethical challenges, practices, and beliefs in the online data research community. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*. ACM, 941–953.