COMP 551 Project 1
Predicting Reddit Comments Popularity Scores

Authors: (Group 77)
Yunxi Chen 260705209
Yan Miao 260711311
Shiqiao Zhu 260706267

Abstract

In this project, we investigated the performance of two versions of linear regression model, with closed-form approach and with gradient descent approach for predicting comment popularity on Reddit. We examined the given features (including the text features) and also two new features: "num_words" and "words_freq". In particular, "children" and "words_freq" significantly improved the performance of the model. The most important discoveries we had from this project are: for data preprocessing part, it is crucial to select good features and apply proper transformations to the data; for model training and validation part, on this specific data set, the closed-form approach outperforms gradient descent approach on both runtime and error.

Introduction

Reddit is a social news and content aggregation website with enormous amount of users and page views. Reddit users can create posts containing either links or text, for which other users can "upvote" or "downvote". In this project, our goal is to use linear regression to predict the popularity (which can possibly be affected by the number of upvotes and downvotes) of comments on Reddit. Here, we used a metric called "popularity_score" to measure the popularity of a comment.
We implemented both the closed-form (least square) linear regression and gradient descent linear regression on the data set and made comparisons between the two models in terms of prediction error and running time.
Previous work in comment popularity prediction has considered features that are closely related to the content of the post itself, which concerned mostly the sentiment analysis of the posts. However, there exist other factors that can also impose influence on the popularity of posts. In this project, we examined the effect of several features such as "children", "controversiality" and "is_root". Two additional features:"num_words" and "words_freq" are introduced. Most of the features concern less about the content of posts.
During the experiment, we had two important findings. First, data transformation can improve the prediction outcome. Take the feature "children" for example, performing data transformation by taking logarithm and square root can effectively reduce the error. Second, although gradient descent seems like a better model (with iterative update steps and tunable hyperparameters), its actual performance on our dataset was not efficient enough compared to the closed-form approach. More specifically, it took much more time than the closed-form to get nearly the same error on the training examples. However, this might not be the case when we have larger and larger datasets since it involves taking inverse in the closed-form approach.

Dataset

Size information

The total size of the dataset is 12000, which is a list of dictionaries containing the basic features of each comments: "children", "controversiality", "is_root", "popularity_score" and "text". We split the whole dataset in a proportion of 10:1:1, meaning that 10000 for training set, 1000 for validation set and 1000 for testing set.

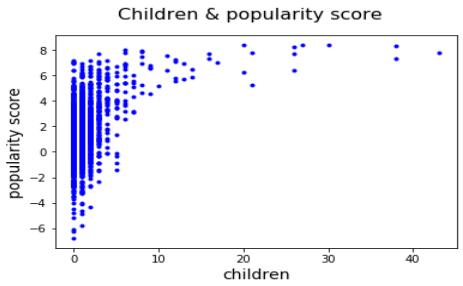
Given features

Before extracting the text features, we first modified the text to ignore capitalization and to split on the whitespace tokens. Then with the handled "text" (consisting of a

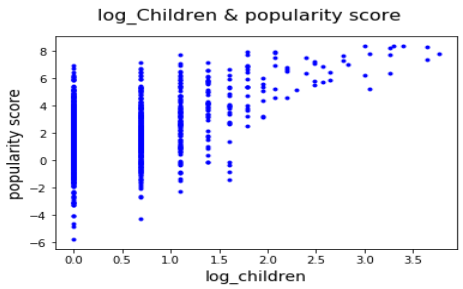
list of words) for each comment, we extracted the text features with two functions: **top_frequent_words** and **text_features**. We used **top_frequent_words** to select out the top 60 or 160 frequent words from the training set (the number of top frequent words needed can be changed by modifying the input parameter "num") and stored them in "wordList". Then, with **text_features** function, we got a list containing 60 (or 160) 10000-dimensional sublists. In each sublist, it stores the number of occurrence of each word from the "wordList" in the in each comment. This is how we constructed the text features.

Note: We applied data transformation on the feature "children" by taking logarithm and square root and included extra columns in our feature matrix. The following plots visualize the transformation.

((a))



((b))



((c))

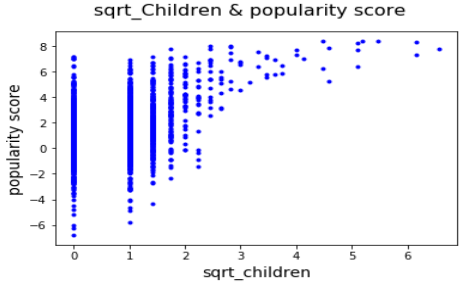


Figure 1: Data transformation

New features

- **num_words**: This counts the total number of words in each comment text.
- **words_freq**: We created this feature by: first sorting the comments according to popularity scores; then picking out the top frequent 150 words in the top 50 comments; after that, removing the first 100 words since they probably have low correlations with the popularity scores; and finally, counting the number of presence of these words in each comment.

Ethical concerns

When working with a public social media dataset, it is hard to tell whether the data itself involves certain private information of the users. For example, such dataset may contain phone numbers, names and even home addresses. Although the information is entirely public, accessing and working with it for research purpose without informing the users can still be seen as an invasion of privacy.

Linear Regression Models

- **Closed-form approach**:

$$\hat{W} = (X^T X)^{-1} X^T y$$

- **Gradient descent approach**:

$$\begin{aligned}
&\mathrm{i} = 1 \\
&\text{do} \\
&\quad \alpha = \tfrac{\eta_0}{1+\beta i}; \\
&\mathrm{w}_i = \mathbf{w}_{i-1} - 2\alpha \left(\mathbf{X}^\top \mathbf{X} \mathbf{w}_{i-1} - \mathbf{X}^\top \mathbf{y}\right); \\
&\text{while } \|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2 > \epsilon;
\end{aligned}$$

Results

Runtime & Error comparison

Note: 1. In gradient descent, we scaled the learning rate by 1/n, where n = #training points. 2. We carried out this first part of the experiment using only **children**, **controversiality** and **is_root**.

	closed-form	gradient descent (η_0=3e-05, β=1e-07, ϵ=1e-06)
runtime (s)	<0.016	around 40
error	1.023	1.089

Clearly, closed-form approach outperforms gradient descent approach on both runtime and error since to achieve almost the same error on training set, using gradient descent takes much more time than using closed-form.

Stability Test of Gradient Descent

We tested the stability of gradient descent by making slight changes to the hyperparameters (i.e. the learning rate) and calculated the resulting MSE on the training set. It is shown in the following table (with ϵ = 1e-06):

Table 1: Gradient Descent MSE

	η_0=8e-06	η_0=3e-05	η_0=8e-05
β=5e-08	1.120	1.089	1.081
β=1e-07	1.120	1.089	1.081
β=5e-07	1.121	1.089	1.082

From the MSE values in the table, we found that the gradient descent approach gives slightly different solutions, depending on the learning rate α, where $\alpha = \frac{\eta_0}{1+\beta i}$. Compared to closed-form approach, gradient descent approach has more factors that can affect the final value of MSE, which indicates that it is relatively unstable.

Training error plot of gradient descent

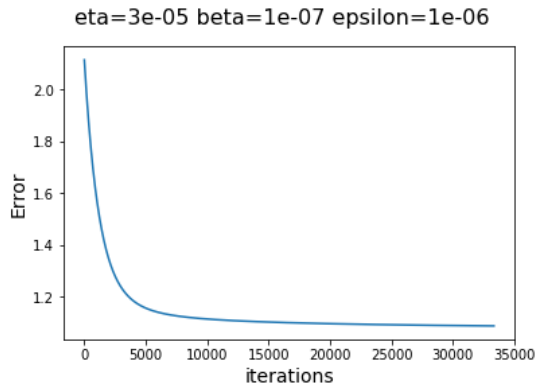


Figure 2: Training error

Choice of model

From the previous results and analysis, we found that the closed-form linear regression gave us a better result within a shorter amount of time. Plus, its performance was of high stability. So we carried out the following experiments using the closed-form approach.

Examine text features

With closed-form approach, we compared three situations using text features: no text features, top-60 words and top-160 words. We made the comparisons by calculating train and validation mean-squared errors. The results present as follows:

Table 2: Text features MSE with closed-form

	no text features	top-60	top-160
train	1.023	1.000	0.989
validation	0.987	0.950	0.959

As we can see from above, both of the training and validation errors goes down after including top-60 frequent words as text features. However, when we change the top-60 words to top-160 words, the training error keeps decreasing whereas the validation error goes up. This indicates that using the top-160 words will lead to overfitting.

Examine new features

Based on what we have discovered above, we added the top-60 words to our features and continued our experiment.
For the two new features, **num_words** and **words_freq**, the following analysis shows how they help to improve performance on the validation set.
Note: Here, we had applied data transformation to **num_words** by taking the logarithm and reciprocal.

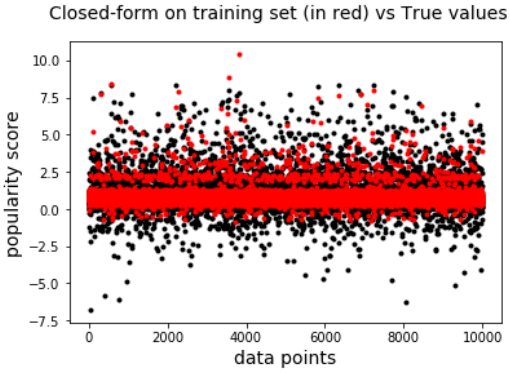
Table 3: Validation errors with new features

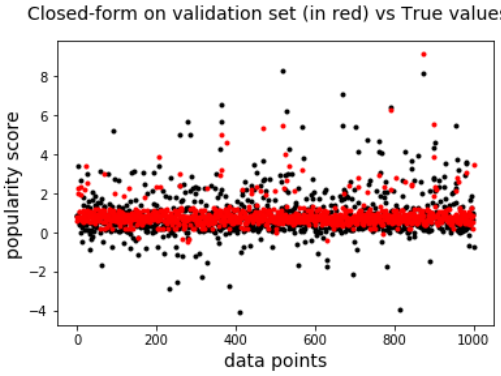
	Errors
without new features	0.950
with num_words	0.947
with words_freq	0.934
with both new features	0.932

With **num_words** and **words_freq**, the validation errors decreased by roughly 0.003 and 0.016 respectively. When we made use of these two new features together, the validation error decreased by roughtly 0.018, which apparently improved the performance of our model.

Put all together

Now we have our best-performing model – a closed-form linear regression model using "children", "controversiality", "is_root", "num_words" "words_freq" and frequencies of top 60 words as features. The following presents its performance on train, validation, and test dataset.





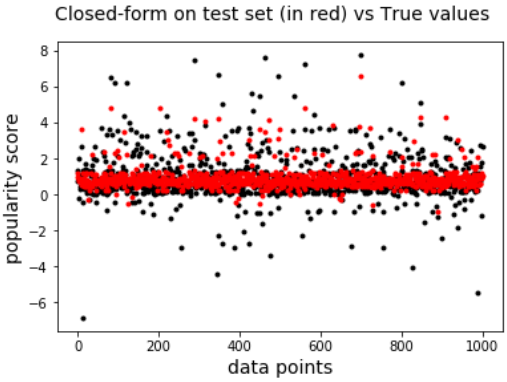


Table 4: Performance of final closed-form model

	Errors
train	0.997
validation	0.932
test	1.236

Discussion and Conclusion

In this project, we have compared the performance of two versions of linear regression, closed-form approach and gradient descent approach on predicting the popularity score of the Reddit comments. This project provided us a great hands-on experience of building a model from beginning and improving it until it can perform well on prediction. Now, we are able to sketch out a procedure for working on a machine learning project based on what we have done. This involves: data preprocessing, model training, model validation and model testing. Data preprocessing focuses on the features. In our case, encoding the boolean feature "is_root" as 0 and 1, converting the comment text into separate words and summarizing out the top-frequent words are great examples. We may also encounter categorical features in the future, which require more advanced encoding operations, But the basic idea here is simply converting the data of all kinds into numerical values that can be taken as input to our models. Model training and validation can be seen as a whole part. The goal is to select out the good features and tune for the better combinations of hyperparameters in the model. Examples would be choosing among "no text features" or "top-60 frequent words" or "top-160 frequent words" and finding proper values of η_0, β and ϵ in our case. Finally, once we have our best-performing model, we need to test it on the test set. This step makes sure that our model is free from overfitting both train and validation set. That is to say, when given another unseen dataset, it can still make good prediction. Also, if we have multiple models and want to pick out the one that performs the best on a specific problem, the final step will provide us valuable information.

Directions for future investigation may include: gathering larger datasets, enriching the varieties of features and improving validation method. Larger datasets will help generalize our model. That is, while we have received 1.236 as test error on the given test set (with size 1000), it is possible that we will get an error that is much greater than 1.236 on an unseen dataset for we may have the model overfitted on our small dataset. Larger datasets include more cases and thus, avoid models from overfitting. Next, we could introduce more features that are relevant to the content of comments such as measurements of sentiment of each comment. Other features such as the authors, the dates and the titles of the comments are useful as well. Finally, if it is hard to search for new data, performing k-fold validation is often a good choice. In this case, we will use the whole dataset for training instead of saving part of it for validation and training only on the rest. This will help to generalize our model as well.

Statement of Contributions

Yunxi Chen: task 1&2, report proof-reading
Yan Miao: task 1&2, report writing
Shiqiao Zhu: task 3, report writing