

# COMP551-Applied Machine Learning-Project 1

Yuan Chen, 260735120  
Jeremy Xie, 260660974  
Tongyou Yang, 260669495

**Abstract**—This is a project report for the course Computer Science 551-Applied Machine Learning. In this project, we investigated the linear regression models for predicting popularity of comments of the online community named Reddit, we implemented two ways of our linear regression model, namely closed-form approach and gradient descent approach. Throughout the project, we have adopted another two features, *logarithm of number of children* and *number of children squared*, alongside with the features provided in the instruction guide to optimize our model. We have found that both of them, improved our model’s performance and the gradient-descent approach is generally slower than the closed-form approach.

## I. INTRODUCTION

The ultimate goal of the project is to develop a model that is capable of predicting the popularity of comments posted on the online community forum named Reddit through linear regression models based on features of these comments.

Here we are given a pre-processed dataset that contains the number of 12,000 comments garnered from the website. We decided to split the dataset into three major portions that are listed as follows:

- Training Set : 1st entry to 10,000th entry.
- Validation Set : 10,001st entry to 11,000th entry.
- Test Set : 11,001st entry to 12,000th entry.

We follow the instruction carefully by pre-processing the dataset first. We converted all the dataset into lowercase letters and then split them by whitespaces. We then built an array named frequency map that stores the frequencies of all the words in the dataset. The frequency map array was sorted in descending order with most frequent word at the first position and the least frequent word at the last. We only take the most frequent 160 words out of the dataset. There are several features that are involved in the first task.

- `is_root` : A boolean feature that possesses value of 1 when it is the root of another comment and 0 otherwise.

- `children` : The number of replies a comment has received.
- `controversiality` : A binary value that represents the controversiality of a comment. It is a propriety computed by Reddit on the values of 0 and 1.
- `text` : The raw text of a comment.
- `word_counts` : A feature works under the `text` feature that calculates the occurrences of the most popular words in a comment.
- `logarithm of number of children` : We took the logarithm of the number of children + 1, this is to avoid to take logarithm of zero.
- `square of number of children` : We use the square of number of children.

In the second part of the assignment, we wish to calculate a sequence of weights. Suppose we have  $n$  Reddit comments each with  $m$  features, we wish to calculate a sequence of weights such that

$$w = (w_0, \dots, w_{m-1})^T \in R^m \quad (1)$$

through two different approaches. The first approach is the close-form approach:

$$\hat{w} = (X^T X)^{-1} X^T y \quad (2)$$

And the second is the gradient descent method. Suppose we have the data matrix  $X$ , targets  $y$ , initial weight  $w_0$  and hyperparameters  $\beta, \eta_0, \epsilon$  where  $\beta$  represents the speed of decay,  $\eta_0$  associates with the initial learning rate and  $\epsilon$  is the stopping criteria, we calculate the decaying learning rate  $\alpha$  in the following fashion :

$$\alpha = \frac{\eta_0}{1 + \beta * i}$$

Our goal is to estimate the weight  $\hat{w}$  through the following algorithm:

---

### Algorithm 1 Gradient Descent Algorithm

---

- 1: do:
  - 2:    $\alpha = \frac{\eta_0}{1 + \beta * i}$
  - 3:    $w_i = w_{i-1} - 2\alpha(X^T X w_{i-1} - X^T y)$
  - 4: while  $\|w_i - w_{i-1}\|_2 > \epsilon$
-

For both of the methods, we compare the result with the target value using the measurement named Mean-Squared Error (MSE).

$$\frac{(y - Xw)^T (y - Xw)}{n} \quad (3)$$

Our most important finding is that the original features provided in the instruction guide fall short when it comes to the effort of minimizing the mean-squared error, after added two features ourselves, namely *logarithm of number of children* and *number of children squared*, the mean-squared error finally reached an acceptable level.

## II. DATASET

The given dataset was initially pre-processed but still there are some work left for us to do, and one of them is to extract the text features from the dataset. In order to do that, we conducted the following functions in our code for the text feature extraction purpose, they are listed as follows:

- 1) We created a `build_freq_map` function to build the frequency map. By traversing through each word of the text, we update the global frequency map at the end of the function.
- 2) Function `pre_process` is used to preprocess the data set. It accepts a data set and number of top words as parameters. For each comment in the data set, it first processes the field “`is_root`” to a binary value then processes the “`text`” field by converting the string to lower case and separating the string by space. After iterating through all the comments in the data set, frequency map is built and sorted by the value in the descending order. `Build_freq_map` function is called to build the frequency map for first 10000 words. Also, a list called “`most_frequent_word`” containing the number of most frequent word is constructed based on the sorted frequency\_map. Then in each comment, we want to create an extra field called “`w_counts`”, which contains a vector of “`number of top words`” dimension where each entry, `w_counts[i]`, is equal to the number of times word `w_i` occurs in that comment. We do this by iterating through the processed “`text`” field in each comment and gradually updating the “`w_counts`” vector. Finally, we can return the processed data set.

In addition to the text features extracted above, we decided to use two more features that might be of value to us.

- 1) We decided to use another feature named **logarithm of number of children** as we believe this is quite useful when it comes to predict the popularity of comments. From the graph of *Number of Children vs Popularity* we can see that the number of children has a non trivial relationship with popularity, however this relationship is not linear. We try to fix this by applying a logarithmic transformation on the data. Since the number of children can be zero, we add 1 to avoid applying logarithm to zero.

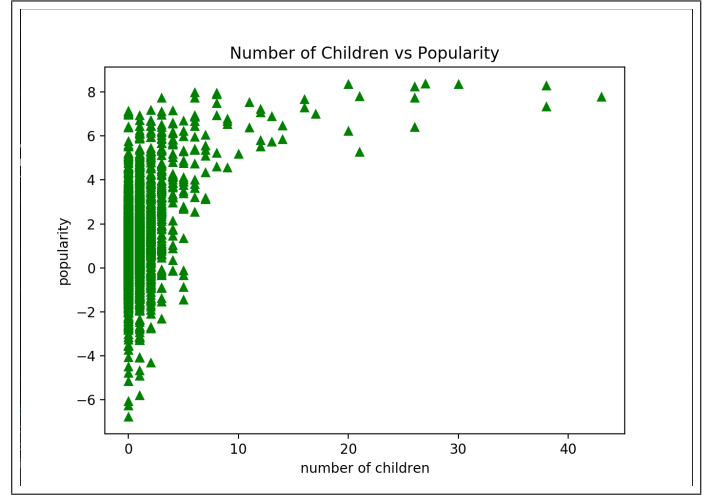


Fig. 1. Number of Children vs Popularity

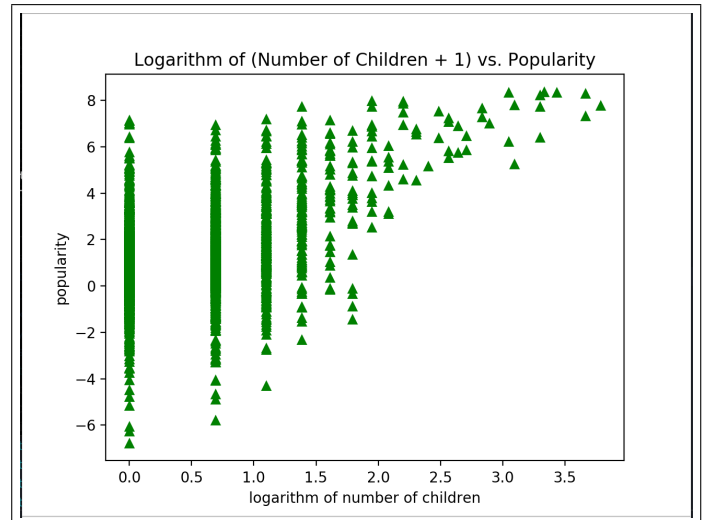


Fig. 2.  $\log(\text{Number of Children} + 1)$  vs Popularity

- 2) For the second feature we have decided to use is **Number of Children Squared**. From the graph *Number of Children Squared vs Popularity* we can tell that two variables have a positive

relationship.

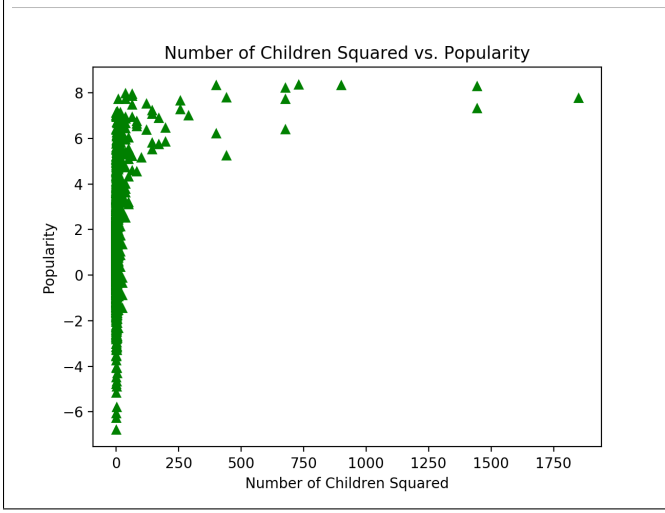


Fig. 3. Number of Children Squared vs Popularity

**NOTE:** For the comparison on the data perspective, please check that in the **RESULTS** section.

#### A. ETHICAL CONCERN

There are some ethical concerns that attracted our attention during our processing time of the dataset. These concerns not only could happen on Reddit, but also a highly possible matter on other sites.

One major issue is the foul language we have found, the infamous f\*\*\* word returned a relatively high frequency than many other words. Another issue is that when we take the comments and to build a model to predict the popularity of comments, we did not acquire consent from the users who posted them, especially considering some comments might contain important information of the users online. The confidentiality of users' profile info is an extremely sensitive topic in today's world and this could create a big obstacle to machine learning researchers and developers as we rely heavily on data from the real world.

### III. RESULTS

We have conducted the following experiments for part three of the assignment.

#### A. RUNTIME COMPARISON USING 3 SIMPLE FEATURES

In this experiment we ignored the text features and only used the 3 simple features which are: number of children, controversy and is\_root.

The runtime of our closed form method is 0:00:00.000853 minutes and our gradient method returns with 0:00:00.209691 minutes. It is safe to draw a conclusion that the runtime of our gradient descent method is relatively slower than our closed-form method. Beware that the gradient descent method returns with various speeds but some gradient steps may not converge, we can only take those results that converges eventually as part of our experiment. All of them were computed using only the three simple features.

For the gradient descent method, we tried different learning rates by varying  $\eta$  and  $\beta$ . We tried a combination of  $\eta$  of values 0.0001,  $1e^{-05}$ , and  $1e^{-06}$ , and  $\beta$  of values 0.001, 0.0001 and  $1e^{-05}$ . The initialization is randomly determined. Some of them did not converge. Among those who did,  $\eta = 1e^{-05}$  and  $\beta = 1e^{-06}$  gave the best runtime and an MSE on the training set of 1.0846830751328929 and an MSE on the validation set of 1.0203291980027545.

#### B. CHOOSING LEARNING RATE FOR MODEL WITH 165 FEATURES

We chose the learning rate for the gradient descent method by testing different  $\eta$ ,  $\beta$  and  $\epsilon$ . The hyperparameters need to be small, thus we let  $\eta$  to be smaller than 0.0001,  $\beta$  smaller than 0.001 and  $\epsilon$  smaller than 0.0001. We tried various combinations of the parameters by decreasing the values by a factor of 10 every time we tried. Some did not converge. Among the ones that converged, the two best MSE on the training set were: 1.0021048436610491, 1.0001340964213374, and were giving by  $\eta = 1e^{-08}$ ,  $\beta = 1e^{-06}$ ,  $\epsilon = 1e^{-06}$  and  $\eta = 1e^{-08}$ ,  $\beta = 1e^{-07}$ ,  $\epsilon = 1e^{-06}$  respectively. We decided to choose the first one since it gave us a better validation MSE of 0.9695724564844914. Thus we used these parameters for the following sections.

#### C. MSE COMPARISON

**NOTE:**The 3 simple features are children, controversy and is\_root.

1) **TRAINING SET:** The Mean-Squared Errors of the training set returned from the closed-form method are listed as follows:

- Closed form with NO TEXT FEATURES : 1.0846830709157256
- Closed form with top-60 words + 3 simple features: 1.060429141685383
- Closed form with 160 words + 3 simple features: 1.0477763217987113

2) *VALIDATION SET*: The Mean-Squared Errors of the validation set returned from the closed-form method are listed as follows:

- Closed form with NO TEXT FEATURES : 1.020326684843145
- Closed form with top-60 words + 3 simple features: 0.9839397297217666
- Closed form with 160 words + 3 simple features: 0.9950693970669267

From the data listed above, we can tell that the model of 160 words has the possibility of overfitting since the MSE of the validation set using this model appears to have a increase after descent. The safest method for us is to use the top 60 words model in order to prevent unnecessary mistakes from 160 words model.

#### D. COMPARISON WITH NEW FEATURES

**NOTE:** This part we use the model of top-60 words, and 3 simple features.

By adding the feature:  $\log(\text{children} + 1)$  we have the following results:

- *TRAINING SET* :  
Closed form MSE : 1.0399834822204392
- *VALIDATION SET* :  
Closed form MSE : 0.9747734373618556

Comparing to the previous section's result of 163 features, adding this feature decreases the training MSE by 0.02044565946 and the validation MSE by 0.00916629235. Both MSE are improved significantly, which means that this feature is helpful in predicting the popularity.

By adding the feature: number of children squared we have the following results:

- *TRAINING SET* :  
Closed form MSE : 1.0137811352156898
- *VALIDATION SET* :  
Closed form MSE : 0.9626394443118133

Comparing to the previous section's result of 163 features, adding this feature decreases the training MSE by 0.04664800646 and the validation MSE by 0.02130028541. Again, both MSE are improved significantly, which means that this feature is also helpful in predicting the popularity.

Incorporating both features:

- *TRAINING SET* :  
Closed form MSE : 1.0096198072343652
- *VALIDATION SET* :  
Closed form MSE : 0.9601250768334687
- *TEST SET* :  
Closed form MSE : 1.2815894543441326

After incorporated our two new features, we can tell that both of the closed-form training set MSE and validation set MSE become significantly smaller than the corresponding model with no new features. Our test set performance MSE is within an acceptable range. If we run gradient descent method with small changes in the learning rate, namely the hyperparameters, it might lead to vary different changes whereas the closed-form solution always returns the same value.

#### IV. DISCUSSION AND CONCLUSION

Since this is our first ever machine learning project, we have learnt a lot throughout the process of constructing a machine learning model, for example :

- The dataset must be split into three major portions, namely training, validation and test set such that test set should never be used until the model is complete.
- The hyperparameters for the gradient descent algorithm should be small enough for the model to converge.
- Gradient descent algorithm generally possesses the potential of improving the performance of our model, it usually returns better result than the closed-form approach. However, our model begs to differ.

For future improvement, we should have a slightly larger dataset that has been preprocessed so there will be less outliers that might affect the model.

#### CONTRIBUTION STATEMENT

- Yuan Chen : Task 3, code improvement, final report proofreader
- Jeremy Xie : Task 2, final report first author
- Tongyou Yang : Task 1, code improvement, final report proofreader