Classification of posts on Reddit*

Pooja Naik
Graduate Student
CSE Dept
UCSD, CA, USA
panaik@ucsd.edu

Sachin A S
Graduate Student
CSE Dept
UCSD, CA, USA
sachinas@ucsd.edu

Vincent Kuri
Graduate Student
CSE Dept
UCSD, CA, USA
vkuri@ucsd.edu

ABSTRACT

Online communities such as Reddit.com have unwritten rules of conduct that are only governed by the community itself. The idea of creating and placing content to gain the most amount of attention with the least amount of effort is the goal of any user. A number of factors play a role in determining the 'likeness' of a post on Reddit.com. Multiple resubmissions of the same content in multiple subreddits can provide insightful relationships into how popular a new post about the same content is going to be in a subreddit. Our main goal is to predict the number of upvotes received on a post so we can analyse the factors affecting the prediction to use them to our advantage. Our experiment also aims to classify posts into subreddits using only textual features so we could use this technique to recommend sub-reddits to users.

Keywords

Data Science, Reddit post analysis, Multi class classification, Regression, Principal Component Analysis, Ransdom Forest, Decision Tree, Collaborative Filtering, Machine Learning, Artificial Intelligence

1. INTRODUCTION

The task of predicting the popularity of a post is especially complex because it depends on a number of factors. To further increase the difficulty, online communities in Reddit.com have the concept of sub reddit which is akin to a smaller sub community within a larger community. Since each such subreddit is unique in its own way, the unwritten rules related to posting content can vary widely between different sub reddits. Theoretically, if we have enough data about each and every subreddit, and also about each and every post, then we might be able to gain insight about the popularity of a new post accurately. But such a dataset is very hard to obtain, and some subreddits are so niche that they just do not have enough data.

As a user of an online community similar to reddit, the aim of the posting user is to gain the most number of upvotes from the community.The main goal of our paper is to predict the number of upvotes a post gets. Since upvotes is a measure of how 'liked' a post is, being able to predict upvotes can provide a key insight into the factors affecting this prediction and provides scope to influence these factors to maximize upvotes. To this effect, we have evaluated prediction for a random test set (by doing a random split on the data).

We further went on to build a recommendation model to recommend user what time of the day would be best to submit his post inorder to receive maximum upvotes. The approach was similar to item based collaborative filtering and built on a similarity matrix of post vs hour of the day with values involving scaled version of upvote count.

As an additional challenge, we tried to accurately classify posts into sub-reddits. This is particularly hard due to the skewness of the data, but we try to use only the text in the title despite the presence of other features with the aim to be able to use this in sub-reddit suggestion to users before they submit the post.

2. THE DATASET

The dataset used for this experiment was the Reddit dataset from the Stanford Network Analysis Project [1]. The dataset is made up of reddit posts that had been resubmitted multiple times with the same content. To ensure that posts had the same content, only posts with images were considered. The dataset consists of 132,307 images, which is made up of 16,736 unique images. Each image has been submitted an average of about 7 times.

The number of upvotes range from 0 to 86,707, with an average of about 1058 upvotes per post. 45 posts have 0 upvotes as compared to only 10 posts that have more than 60,000 upvotes. Fig.1 shows the distribution of upvotes in the data over 50 buckets.

There are 63,337 unique users whose posts are recorded in this dataset giving us the idea that these users post multiple times. A little more than 20,000 posts don't have users associated with them. Although the highest number of posts a user has in the data is 5608, on an average a user posts about 1-2 times, so user specific data is not very useful to us.

The number of downvotes range anywhere from 0 to 86707,

*(Produces the permission block, and copyright information). For use with SIG-ALTERNATE.CLS. Supported by ACM.

CSE255 '15 San Diego, California USA

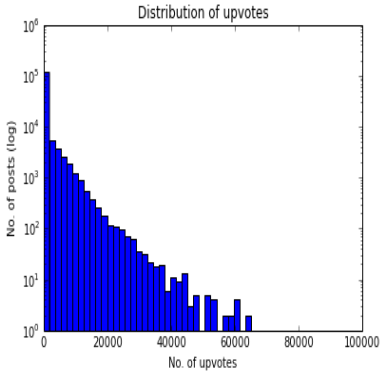

Figure 1: Histogram showing distribution of upvotes

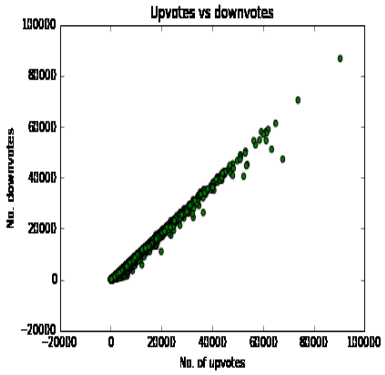

Figure 2: Scatter plot showing correlation of upvotes and downvotes

with an average of 825 downvotes per post. Surprisingly the number of posts with 0 downvotes is around 1,830 posts which shows that poeople prefer to upvote posts before they even begin downvoting posts, so it is not surprising to find only 14 posts with more than 50,000 downvotes.

The dataset also gives us the number of comments that were posted to a particular reddit post. The number of comments range anywhere between 0 to 8357 comments for the most popular one. On average 39 comments are posted per post. The low number can probably be attributed to the multiple steps involved in posting a comment as compared to downvoting or upvoting a post. Hence, it is of no surprise that 45,102 posts have 0 comments, and only 492 posts have more than 1,000 comments.

The number of unique sub-reddits are 867, and only 63 of those have more than 20 submissions, leading to a massive skew. The 63 sub-reddits account for around 129K posts while the remaining 804 sub-reddits only account for 2K posts. For the classification problem, we ignore posts from the 804 sub-reddits in the training data, considering them as misclassified in the test data. The data is so skewed that only 6 sub-reddits account for 116,253 posts and the largest group of posts, about 55k - almost half of the 116k, belong to the sub-reddit 'funny'.

3. FEATURES

The features have to be carefully selected so that they can provide us with the most insight about the new post. Each of the selected features used in our model are outlined below.

Title Length : The number of characters in the title and the number of words in the title are used as features because shorter titles are easier to read as compared to longer titles.

Hour of the day : Users are simply more active in certain hours of the day and the tendency to upvote is a loose function of that. As per our analysis, there was a weak correlation between time of the day and upvotes received so we added this information in the form of a 23-bit vector.

Automatic Readability Index of the title : ARI is a readability test that is used to gauge the understanding of a text. The output of the ARI is a number which gives the US grade level of education needed to comprehend the text. The ARI can provide insight on how the community reacts to different titles.

Downvotes : Downvotes indicates how many users have disliked a post. Downvotes, as we find out from our evaluation, turns out to be one of our most important features. Fig.2 demonstates a clear correlation between upvotes and downvotes.

Number of comments : Number of comments is very indicative of the popularity of a post and has a positive correlation with the upvotes.

Community : The community or sub-reddit the post is posted in has a large influence on the upvotes. Communities are places where like-minded people interact with posts of their interest. Good content posted in the right subreddit can go a long way. Encode the sub-reddit information as a bit-vector, using only the 63 sub-reddits with more than 20 posts. The remaining sub-reddits are all put under a slot representing miscellaneous sub-reddits.

Number of resubmissions : Users love upvotes and more users tend to resubmit popular and well-like posts in the hopes of getting more upvotes maybe in different communities or at different times.

Sentiment of the title : A strongly positive or negative title invokes polarizing reactions from people leading to many or not-so-many upvotes. We represent sentiment as a 2-bit vector where [0,0] stands for neutral, [1,0] for positive and [0,1] for negative.

Average number of upvotes in prior submission : The average number of upvotes the image received in prior submissions is indicative of how good the content of the image is, which in turn influences upvotes.

4. MODELING AND ANALYSIS

We tried the following 3 approaches to solve the aforementioned problem

- Regression analysis to predict upvotes for a new submission
  - Prediction for a randomly sampled 15% data set as test
- A collaborative filtering based approach to predict the best time to submit a post for maximum upvotes
- Multi class classification to predict the subreddit of a submission based on its votes and text content

The above features with its many values as bit vectors added up to more than 100 dimensions. In order to scale the features and keep the important projections we did a principal component analysis on the above feature set.

4.1 Principal Component analysis

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables.

The PCA analysis revealed that the top 5 components explains nearly 99% of the data. Following is the percentage of variance explained by the top 5 principal componenets

component 1	0.84
component 2	0.15
component 3	0.00089
component 4	0.000061
component 5	0.000031

Looking at the eigenvector of these 5 principal componenents we realize that the following features contribute the most

- downvotes received by the post
- number of comments received by the post
- average upvotes received by the post in prior submission

Once we had figured out the key features and principal components, we now ran different models to predict the performance.

4.2 Regression Analysis

We performed regression on the above features to predict the upvotes. We used a number of regression models and error metrics to understand and analyze the performance.

4.3 Error/Accuracy Metrics

- R^2 **Coefficient** One of the better metrics to analyze performance of a regression is the coefficient of determination or R^2 coefficient. Coefficient of determination is a number that indicates how well data fit a statistical model - sometimes simply a line or a curve. An R2 of 1 indicates that the regression line perfectly fits the data, while an R2 of 0 or negative indicates that the line does not fit the data at all.

  If $\bar{y}$ is the mean of the observed data:
  $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$
  then the variability of the data set can be measured using three sums of squares formulas:

  The total sum of squares (proportional to the variance of the data):
  $SS_{\text{tot}} = \sum_i (y_i - \bar{y})^2$,
  The regression sum of squares, also called the explained sum of squares:
  $SS_{\text{reg}} = \sum_i (f_i - \bar{y})^2$,

  The sum of squares of residuals, also called the residual sum of squares: $SS_{\text{res}} = \sum_i (y_i - f_i)^2$,

  The most general definition of the coefficient of determination then is $R^2 \equiv 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$.

- **Root Mean Square Error** We also used rmse to further analyze the performance. The root-mean-square error (RMSE) is a frequently used measure of the differences between values (sample and population values) predicted by a model or an estimator and the values actually observed.
  Since in our case, it is useful if our prediction is as close to the actual expected value, i.e the distance between prediction and true value is of significance, rmse is a good error metric to analyze the performance as well.

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{n} (\hat{y}_t - y)^2}{n}}$$

.

We further calculated the mean and standard deviation of all the upvotes and compared the standard deviation with the rmse errors to understand and verify that the variance explained is as per our expectations

4.4 Models

We implemented the following regression models alongwith the corresponding parameters for each of the models. We ran simple gridSearch to find out the best parameters for each model

- **Linear Regression** Linear regression or the method of least squares is a standard approach in regression analysis, which means that the overall solution minimizes the sum of the squares of the errors made in the results of every single equation.

- **Random Forest Regressor** Random forests is an ensemble method that operate by constructing a multitude of decision trees at training time. Each decision tree makes a prediction and the forest outputs the class that is the mode of the classes (classification) or in our case the mean prediction (regression) of the individual trees.
  The key trick in the model is in bagging or **bootstrap aggregating** which is an ensemble algorithm designed to improve the stability and accuracy of machine learning algorithms. Bootstrapping generally refers to random sampling with replacement.
  Given a standard training set D of size n, bagging generates m new training sets D_i, each of size nâĂš, by sampling from D uniformly and with replacement. By sampling with replacement, some observations may be repeated in each D_i. If nâĂš=n, then for large n the set D_i is expected to have the fraction $(1-1/e)(63.2\%)$ of the unique examples of D, the rest being duplicates. This kind of sample is known as a **bootstrap sample**.
  The m models are fitted using the above m bootstrap samples and combined by averaging the output (for regression) or voting (for classification).

By using this ensemble method which bootstraps on the dataset, we correct for the overfit of individual decision trees.
The parameters we used for our Random Forests are as follows

```
n_estimators=50, criterion='mse', max_depth=None,
min_samples_split=2, min_samples_leaf=1,
min_weight_fraction_leaf=0.0, n_jobs=1,
max_leaf_nodes=None, bootstrap=True,
oob_score=False, random_state=None, verbose=0,
warm_start=False, max_features='auto',
```

We chose mse as the loss function as the distance between predicted value and true value is of significance to us. We tried with different estimators and max_depth and concluded on the aforementioned values based on the model's performance and time consumed in running the model

- **Gradient Boosting** Gradient Boosting is yet another ensemble model which is a combination of many weak learning models. A weak learner is defined to be a predictor which is only slightly correlated with the true results (it can label examples better than random guessing). In contrast, a strong learner is a predictor that is arbitrarily well-correlated with the true results. gradient boosting combines weak learners into a single strong learner, in an iterative fashion. For a given loss function it runs the model with a weak learner, using a pre defined loss function, calculates the loss with this learner and adds a new estimator to the learner such that the new loss/cost function is lss than the revius one In this way it keeps on adding the estimator until there is no further improvement in the loss. The estimator to be added is calculated based on the previous weak learner predictions and the true values. The parameters we used for our gradient boosting are as follows

```
number of estimators=250, learning_rate=0.1,
loss function = least squares, presort='auto',
n_estimators=100, subsample=1.0, max_depth=3,
min_samples_split=2, min_samples_leaf=1,
min_weight_fraction_leaf=0.0, verbose=0,
max_features=None, init=None, max_leaf_nodes=None,
warm_start=False, random_state=None,
```

We tried with different estimators and max_depth and concluded on the aforementioned values based on the model's performance and time consumed

- **Collaborative Filtering** Collaborative filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the collaborative filtering approach is that if a person A has the same opinion as a person B on an issue, A is more likely to have B's opinion on a different issue x than to have the opinion on x of a person chosen randomly We frame the problem of predicting the upvotes for a submission as a collaborative filtering based recommendation problem wherein for a given post we recommend which is the best time to post based on the performances of other posts at different times. For time we take a granularity of an hour. We then built a similarity matrix of posts vs hour of the day when it was submitted for the training data. For our problem, we used an item based collaborative filtering approach and trained an SVD model and computed the sparse matrix of image/posts vs our of the day, the values in the matrix corresponding to upvotes received. Based on this we could predict the best time to submit a post. The same can be extrapolated to predict the best subreddit for an image submission. However, we excluded this from the analysis since the dataset had a very skewed distribution of subredddits.
In order to compute performance of this approach, we used a randomly split 20% of the data as test data and for this dataset, predicted the upvote count for a given post at the corresponding hour of submission. The error then was computed as the RMSE error of the prediction's true value and using the true array of values of prediction and true values, we could compute corresponding $r^2 aswell$

4.5 Upvotes prediction

Table 1 shows the performance of the 4 models described on the given dataset. As can be seen, the 2 key features of downvote count and comment count contribute the most to the overall performance of the model further verifying our PCA analysis wherein the first 2 projections/components explain 99% of the total variance. One other feature which contributes the most to the model performance is the average number of upvotes the post received in prior submissions. We had also included many key features around the time of submission and title of the post. But, there is little correlation between these features and the upvotes received.

Further in the dataset considered, upvotes has a **standard deviation of 3504** and therefore, we end up with RMSE values in similar ranges for the corresponding regression analysis. A better fit with an coefficient of determination of 1 has an RMSE of 186.
We further note that the performance of the 2 ensemble models as well as collabrative filtering is very similar as far as upvotes prediction is concerned.

The collaborative filtering model takes into account only time and upvotes and therefore has values similar to regression performance for the cases of no downvotes and comments feature

Moreover, when evaluation is done after removing all the text features, we see only a marginal drop in RMSE and an even smaller drop in R^2. For example, the R^2 given by Random Forest Regressor still stays at 1 for random test set evaluation and the RMSE given by the Gradient Boosting Regressor is at 181.15 compared to 171.49 with text features leading us to the conclusion that the title in fact may have very little to do with a user's decision to upvote the post. We demonstrate this in Fig.2 by plotting random 60 points in the test set and our predictions for them without using text features.

4.6 Additional Analysis: Subreddit Prediction

We performed multi-class classification to predict the subreddit for a post. This is an interesting problem as it gives us the opportunity to recommend appropriate sub-reddits for

Table 1: Random Test Set Evaluation

Evaluation/Model	PCA + Linear Regression	Random Forest	Gradient Boosting	collaborative filtering
R^2	0.9912	1	0.9970	-
R^2 w/o downvotes, comments	-0.24611	-0.3700	-0.2425	0.034
RMSE	293.84	186.90	171.49	-
RMSE w/o downvotes, comments	3534.59	3702.76	3529.58	3655

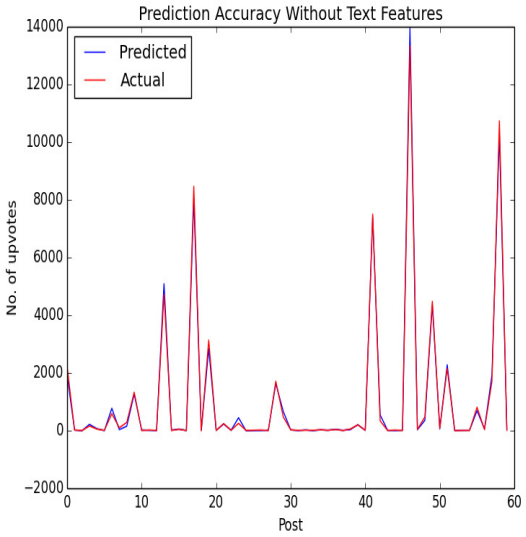

Figure 3: Plot of 60 random points in the test set and our predictions for them without using text features

a post. We used the **average word vectors of the image captions** as features since they can be extracted before post submission. We use the 50-dimensional GloVe vectors [2] pre-trained on Wikipedia 2014 and Gigaword 5 datasets to capture the context of the words in the image captions. Image captions are typically 5-10 words long, with no more than 5 non stop-words on average. Therefore, we consider it appropriate to simply use the average word vector as a feature. The classification is performed by a **random forest classifier** with 50 estimators. Since the data is extremely sparse, we only use sub-reddits that have at least 20 posts. This limits the number of samples to 129627 and the number of classes (sub-reddits) to 63. Table 2 provides a confusion matrix of the above classification analysis.

4.7 Evaluation

With the setting described above, the classifier barely achieves an accuracy of 42%. However, it is easy to understand why the model performs so poorly. The obvious reason is that the data is extremely skewed. The two most popular sub-reddits account for more than 61% of the data (*funny*: 41% and *pics*: 19%). Table 3 shows the confusion matrix for the top 4 subreddits. Clearly, the model is overwhelmingly predicting the top 2 sub-reddits.

However, the problem isn't merely data skew. The data skew is actually caused by a significant re-submission of posts in the top sub-reddits. To verify this, we map each sub-reddit into a image ID space. This means every sub-

Table 2: Sub-reddit prediction confusion matrix. Fraction of *row* predicted as *column*.

	funny	pics	WTF	gifs
funny	0.630	0.141	0.069	0.07
pics	0.632	0.129	0.076	0.074
WTF	0.626	0.142	0.072	0.073
gifs	0.648	0.134	0.069	0.067

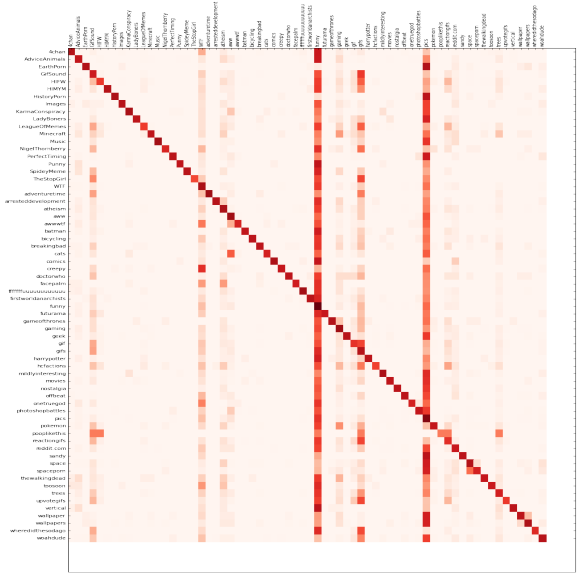

Figure 4: Heatmap of common Image IDs

reddit is mapped to a binary vector of size 16732 (number of unique images in dataset). The distance between two sub-reddits is simply the size of the intersection of their image space vectors. Figure 4 shows the heatmap of common images across sub-reddits. The two solid vertical red lines on *funny* and *pics* shows that posts shared in other sub-reddits are overwhelmingly re-shared in these categories.

Due to this inherent similarity of sub-reddits, we considered it more appropriate to predict the *cluster* of sub-reddits to which a post belongs, where a *cluster* is defined as the **K** most similar sub-reddits in the shared image space. The trivial case is $K = 1$ where we must identify the exact sub-reddit. Figure 5 shows the dramatic increase in accuracy with the size of the cluster.

5. CONCLUSIONS

For the aforementioned results, we can conclude that a regression analysis to predict upvotes for the given dataset has an rmse which matches the standard deviation of the dataset. We achieve a coefficient of determination of nearly

Table 3: Cluster prediction accuracy for various values of K.

K	Accuracy
1	0.42
2	0.897
3	0.997
4	1.0
5	1.0

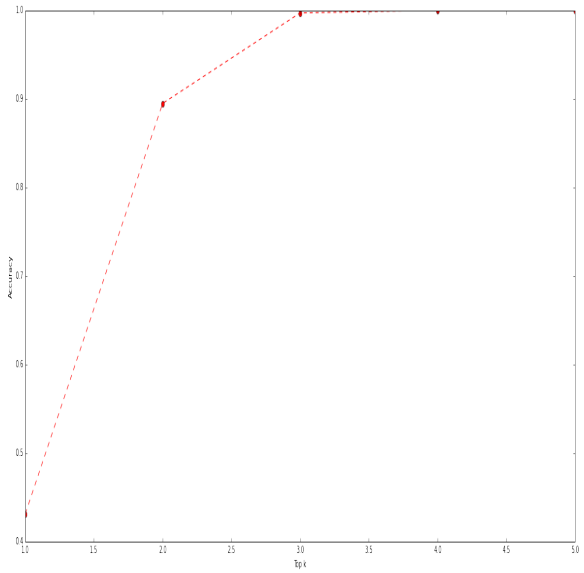

Figure 5: Cluster prediction accuracy for various values of K.

1 while predicting upvotes and therefore can consider the model as a good measure to predict the upvotes for new posts.

While for the multilabel classification, our conclusion is that predicting the exact sub-reddit is not a fruitful exercise for two reasons. First, many posts are re-submitted to a very limited set of popular sub-reddits (e.g. *EmmaWatson -> Celebs*). A model that always predicts these sub-reddits will always be trivially correct. Secondly, the number of sub-reddits in the entire data-set is large (867). Multi-class classifiers do not scale very well to a large number of classes. A better approach would be to map the sub-reddits into a feature space (say, average word vectors) and recommend the K-nearest neighbors in this space.
Further the collaborative filtering based model provides us with a handy quick tool to predict the best times and sub-reddit for the user to submit his post into.

6. REFERENCES

[1] J. L. H. Lakkaraju, J. J. McAuley. What's in a name? understanding the interplay between titles, content, and communities in social media. In *ICWSM*, pages 1532–1543, 2013.

[2] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543, 2014.