# CSE 190

# Professor Julian McAuley

# Assignment 2: Reddit Data

by

Forrest Merrill, A10097737

Marvin Chau, A09368617

William Werner, A09987897

# Table of Contents

# Introduction

Reddit is a massive online community where users anonymously submit content ranging from text posts to images. Users are able to immediately provide feedback on submissions through comments and a rating systems where positively received posts are given an “upvote” while negatively received posts are given a “downvote”. Popular posts are displayed on the “front page” of each sub community known as subreddits which are moderated by other users. Our project attempts to characterize and identify the features that contribute to a successful post on Reddit using the various features provided in the dataset.

Through the course of our analysis, we examine the score of a post (score = #upvotes - #downvotes) and also the approval rating of a post (approval rating = score/#total_votes) to create various predictive models. We use the number of comments of a post as well as the time posted to tune a prediction of the score. Furthermore, we examine trends in the top subreddits, and also look into the nature of deleted posts. Overall, our careful analysis of a variety of trends in the reddit data yields some interesting and useful results.

# Dataset

We are using the reddit dataset from snap.stanford.edu

URL: http://snap.stanford.edu/data/web-Reddit.html

Dataset: http://snap.stanford.edu/data/redditSubmissions.csv.gz

## Dataset Statistics

| | |
|---|---|
| Number of submissions | 132,308 |
| Number of unique images | 16,736 |
| Average number of times an image is resubmitted | 7.9 |
| Timespan | July 2008 - Jan 2013 |

## Fields

| | |
|---|---|
| #image_id | id of the image, submissions with the same id are of the same image |
| unixtime | time of the submission (unix time) |
| rawtime | raw text of the time |
| title | submission title |
| total_votes | number of upvotes + number of downvotes |
| reddit_id | id of the submission on reddit, e.g. reddit.com/14c3ls |
| number_of_upvotes | number of upvotes |
| subreddit | subreddit, e.g. reddit.com/r/pics/ |
| number_of_downvotes | number of downvotes |
| localtime | local time of the submission (unix time) |
| score | number of upvotes - number of downvotes |
| number_of_comments | number of comments the submission received |
| username | name of the user who submitted the image e.g. www.reddit.com/user/thatseffedup |

## Interesting Preliminary Findings

When we began analyzing the set of posts made to reddit, we first gathered some basic statistics regarding the dataset. This included many averages such as average scores, up/downvotes, number of comments, etc. (The raw data gathered can be seen in the chart below). Using this basic data, we intend to create a predictor that will be able to predict whether or not a post may be successful or not (success is based on the score of the post) that will utilize the other pieces of data that are available to us in the data set.

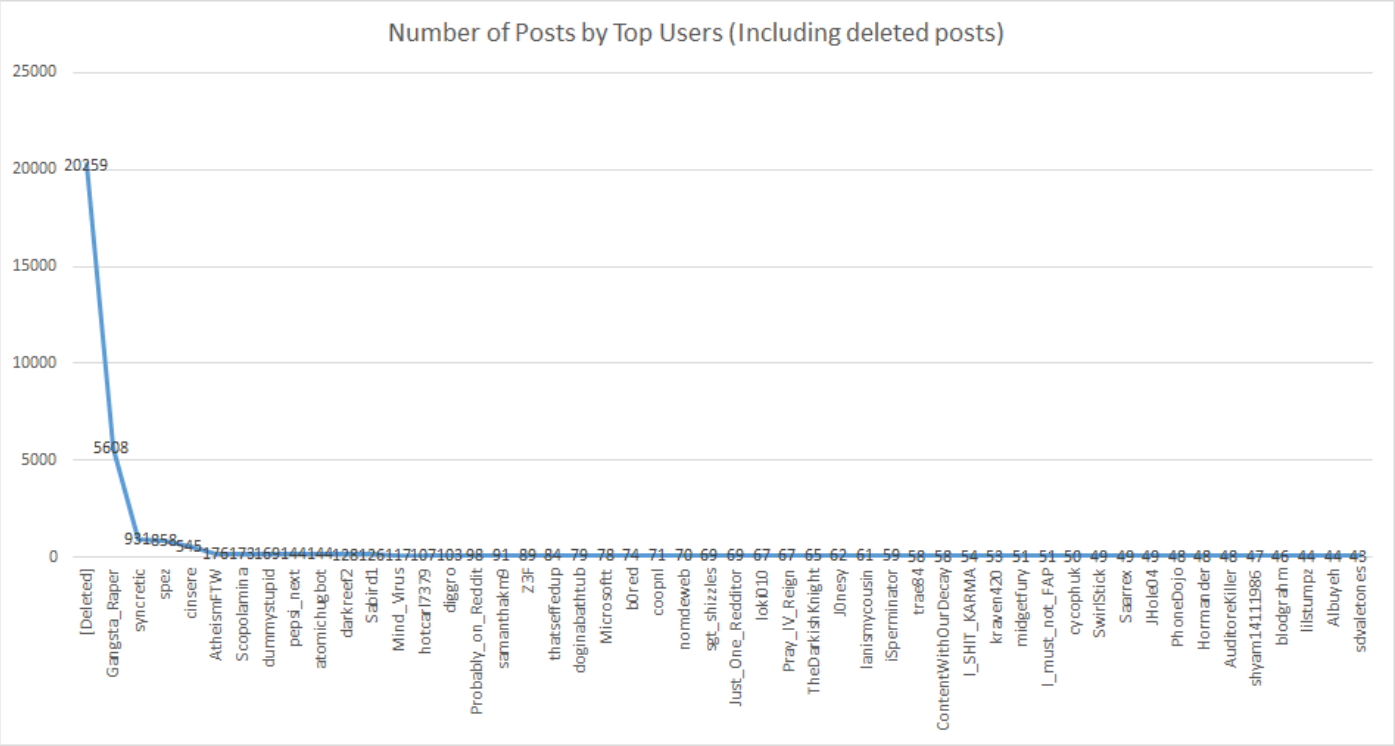
Additionally, we then decided to find the total number of users, as well as the number of posts made by each user. This led us to discover that the most active “user” turned out to be the empty string (“”). Fortunately, because we were familiar with reddit, we recognized that the only times when the username of the original poster is no longer visible on a post (or a comment) is when the user has deleted that post/comment, or when a post has been removed by moderators.

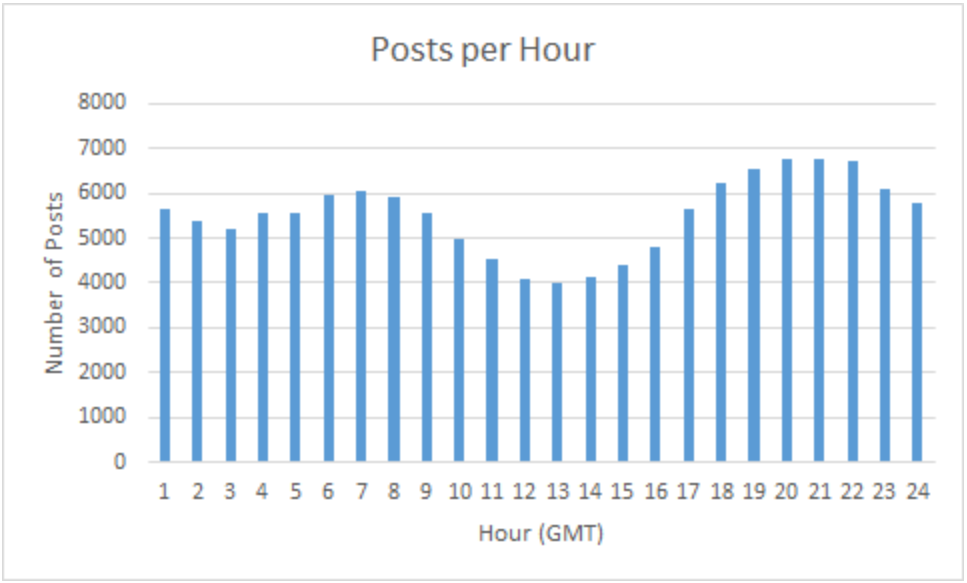
From this information, we realized that we now had 20,259 posts that had been deleted, and while we no longer had the username of the original poster, we did have valuable information such as the total score, the number of up/down votes, and the number of comments that had been left on that post. Because this information remained intact on deleted posts, we decided that we would attempt to use the data present on all posts, in order to predict whether or not a post remained active at the time that this data was gathered, or if the post had been deleted by the original poster.

## Exploratory Analysis

| | |
|---|---|
| Total number of users | 30592 |
| Total number of posts | 132308 |
| Average number of votes | 798.326488194 |
| Average number of upvotes | 448.047548145 |
| Average number of downvotes | 350.278940049 |
| Average score | 97.7686080963 |
| Average number of comments | 16.6030852254 |

| | |
|---|---|
| Average posting time | 1340036295 (06/18/2012 @ 4:18pm) |
| Average title length | 2 |
| Number of deleted posts | 20259 |



# Predictive Task

Our idea for a useful predictive task is to predict what posts will have the highest scores. Score = (total_upvotes - total_downvotes). After some initial lookups and comparisons on the data, we realized that a potentially useful ratio to calculate would be the approval rating of a post. The approval rating of a post is defined as follows:

```
Approval Rating = ((total_upvotes - total_downvotes) / total_votes)
           OR   = Score/total_votes
```

This rating gives us a number between -1 and 1, with -1 indicating that 100% of users downvoted the post and positive 1 indicating that 100% of users upvoted the post. Now, there are some concerns with the approval rating. For example, if a post gets exactly one upvote, then they will have a 100% approval rating, but this does not mean that the post is popular. However, if a post gets a lot of upvotes (ie, 500) but also gets significantly more downvotes (ie, 2000), then the post is rather unpopular. We would like to examine the usefulness of trying to predict a post's number of upvotes vs the post's score vs the post's approval rating. To analyze the data and make our predictions, we split the data in half for a training and test set each of length 66154.

First, we examine how the approval rating can be used to predict the score. We calculate the average approval rating of a post:

avgApprovalRating = 0.254094561002523 (over training data)

This indicates that when examining all posts, the average post receives more upvotes than downvotes (ie a positive score). A benefit of using the approval rating to predict score is the following: the approval ratio of each post is weighted to be a number between -1 and 1. This prevents outliers with huge amounts of upvotes from drastically skewing the data. The tradeoff is that posts with very few votes have more influence on the data.

We can now start predicting data. We devised our own method for calculating error (this method may well already exist, but we didn't know what to call it). We calculate the percentage error for each prediction and average all of these errors together. For example, if a post has approval rating 0.5 and we predict 0.25, the percentage error for that post is (0.5-0.25)/2 where 2 is the size of the scale (the scale is -1 to 1). This would give us an error of 0.125, or 12.5%. For our first comparison, we compare the true approval rating values of the data against the average approval rating. Using our error calculation schema, the average percent error over the test data is:

avgPercentError = 0.12885537753224496

(using training data's avgApprovalRating over the test data)

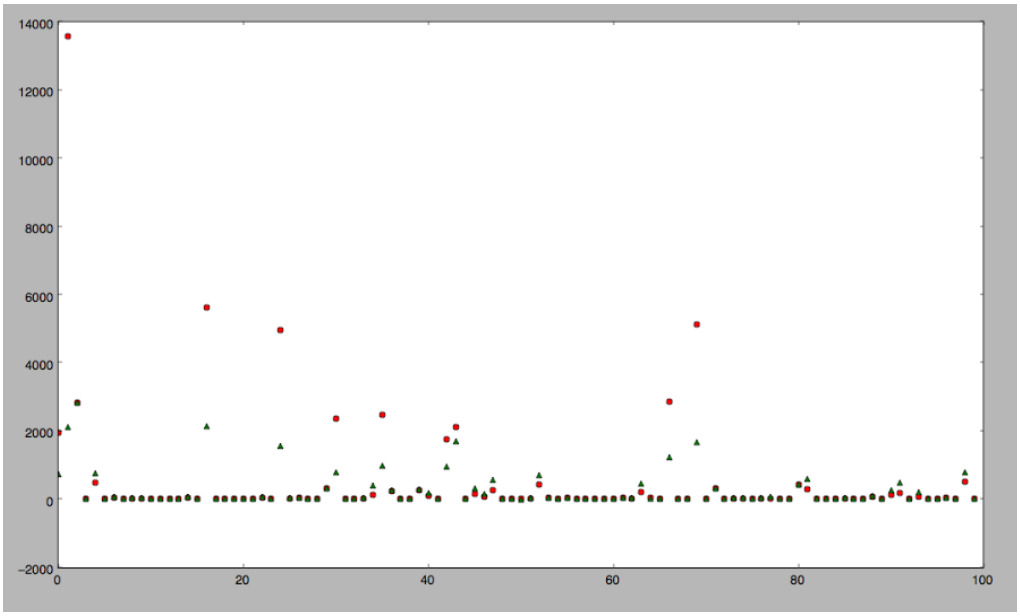
This means that on average, this model predicts the approval rating with 0.87104742967 accuracy. As it turns out, always predicting the average is a pretty decent model for determining the approval rating. We also tried calculating similar baselines using values some test values in place of the average approval rating. These values and rates are as follows:

| Predicted Rating | Average Percent Error |
|---|---|
| 0.254094561002523 (avgApprovalRating) | 0.12885537753224496 |
| 1 | 0.3729313234293037 |
| 0 | 0.17359649799454788 |
| -1 | 0.6270686765706814 |

Let us take this one step further. We can use the predicted approval rating multiplied by the total number of votes to predict a post's score. For these predictions we will use the mean squared error, as the percent error function won't yield conclusive results on score data.

MSE = 1417230.6401335977 (using the simple predictor against the test data)

When examining the data, we can graph our predictions vs the real values. Here are the first 100 predictions with the corresponding values (red is prediction value, green is actual value):




From the chart, we can see that our predictions are less and less accurate the more votes a post has. To address this issue, we must build a better predictor. We turn to a model similar to the one in homework 3:

approval rating = score/total_votes = α + β1(feature1) + β2(feature2)

We first try with the following features:

approval rating = score/total_votes = α + β1(number_of_comments) + β2(unixtime).

We can then use the approval rating to compare with our percentage error rate. We can also use the same model to predict the score and evaluate a new MSE.

alpha = 4.4075070698103804

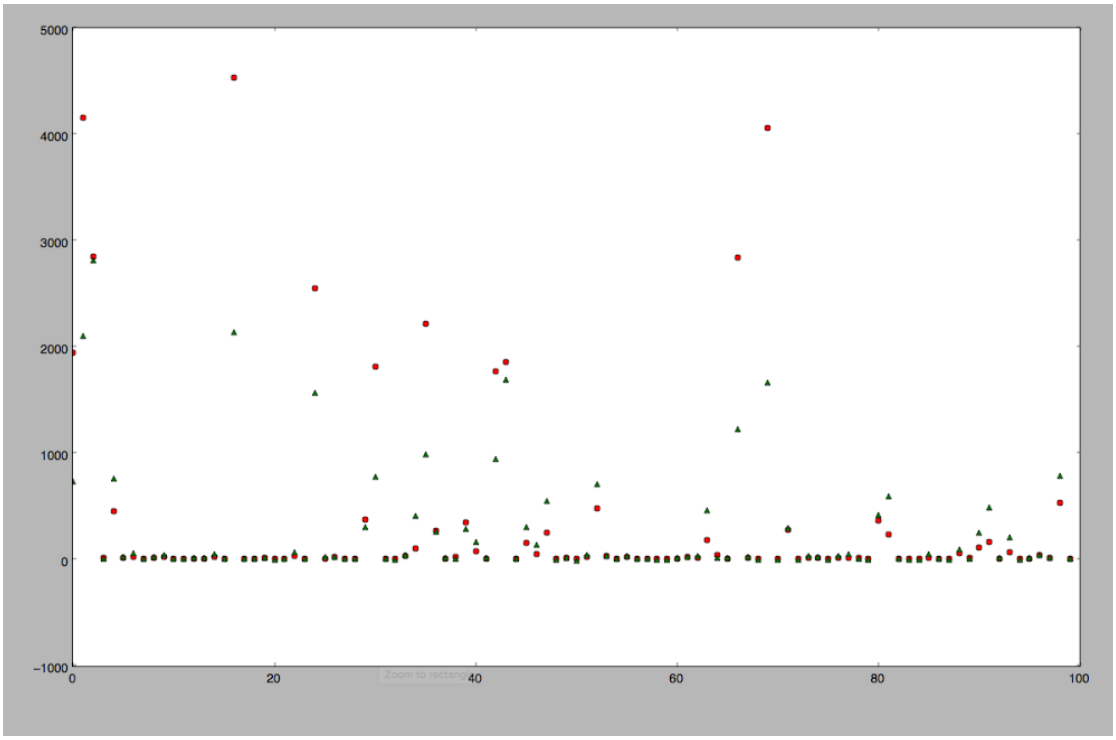
β1(number_of_comments) = -0.00022725949626023617

β2(unixtime) = -3.0930555938496387e-09

Average percent error = 0.12675492859292412 (not a significant decrease from baseline)

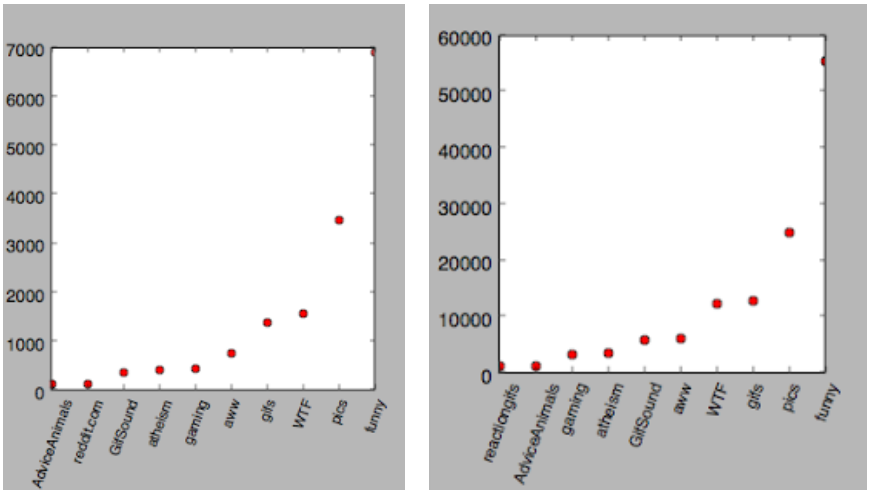
MSE = 1192994.7982745632 (down by 200,000! Significant decrease!)

**In the graph below, pay special attention to the y-axis scale as compared to the previous predictor's graph scale**




This graph once again examines the first 100 predicted vs actual values. Examine the scale, before our worst prediction was in the 13,000 to 14,000 range, now it is under 5,000! These results conclude that our trained predictor is much better suited for handling outlying data. Before, our predictor was very close for the average data but was very sporadic for posts with large scores. The new predictor is better, but is unfortunately not close to perfect. Our new model suggests that more comments is actually not a good thing for achieving a high score. Perhaps more controversial posts spark flame wars and the post's score reflects that attribute? Also, posts with larger unixtime values tend to have a lower score.

## Additional analytics




The two images above demonstrate the popularity of an image given the subthread. The graph on the right displays the 10 subreddits with the most posts, including duplicate image id posts. The image on the left displays the counts of each image id just once, and is only grouped with the subreddit under which it received the highest score. This indicates that popular subreddits yield the highest scores for duplicate posts. To find this information, we first find the maximum score for each image id in the data and append the corresponding subreddit. These subreddits sport the top scores for each unique image id, knocking other subreddits with less successful duplicate posts off the list.

As a contrast to our predictive task, we want to look at whether or not a post will be removed. A post has been removed if the username no longer shows up on the post, as we have tested on reddit. To examine what has caused a post to be removed, we again look at the approval rating as defined above. To examine this approval rating as an indicator for whether or not a post has been deleted, we split the data into two sets: existing posts and deleted posts. We then calculate the average approval rating over each of these sets. The results will be used as baselines and are as follows:

- Non-deleted posts' average approval rating = 28.42721199966877%
- Deleted posts' average approval rating = 0.08732370345079953%

These results indicate that there is a significant difference in the score (total_upvotes - total_downvotes) of deleted posts as opposed to their non-deleted counterparts. To predict whether or not a post is deleted, we need to ask ourselves a few questions:

1. What is it that makes a user want to remove a post?
2. If the user didn't remove the post, was the post inappropriate or flagged as spam?
3. Some removed posts have high approval ratings - why are these posts removed and is there a better indicator to predict their removal?

These questions provide a basis for further predictive analysis for future projects.

# Related Work

Our group is analyzing an existing dataset provided by SNAP (Stanford Network Analysis Project). The dataset provided (redditSubmissions.csv.gz) explores the online communities of Reddit which has become a vital source of information and entertainment in today's social media. Similar to their Reddit dataset, SNAP has provided a dataset for Flickr, a popular photo sharing website. In their research paper, "Image Labeling on a Network: Using Social-Network for Image Classification", Julian McAuley and Jure Leskovec discuss their findings on image retrieval/classification and community development through the analysis of tags.

Himabindu Lakkaraju, Julian McAuley, and Jure Leskovec continued to analyze the development of online communities through their analysis of Reddit and the trends dictating submission success in their research paper, "What's in a name? Understanding the Interplay between Titles, Content, and Communities in Social Media". Lakkaraju, McAuley, and Leskovec developed numerous models and utilized the Jaccard Similarity in order to study the dataset. The influence of submission content, submission title, selected subreddit, and submission time was documented in their statistical model. The community model evaluated the influence of the previously listed factors on resubmissions and its impact on overall success. The language model and topic model were used to analyze the influence a title had on submission success. Lakkaraju, McAuley, and Leskovec associated each word/title with a topic developed using the supervised LDA framework. A title possessed a topic distribution which took the form of a stochastic vector where words unique to each community were identified as either generic, community specific, or content specific. Each word/title was given a linking parameter which identified whether the word is positive, negative, or neutral. Lastly, Lakkaraju, McAuley, and Leskovec implemented the Jaccard Similarity to compare the titles of resubmitted content taking their models into account.

Through their research Lakkaraju, McAuley, and Leskovec concluded that resubmissions are less likely to be popular than the original submission, submissions made to more popular subreddits are more likely to become popular however face more competition, and the timing of submissions play a role in the popularity of a submission. Submission titles also play a key role in the potential success of a submission. Successful titles should be relevant to the target subreddit, unique compared to previous submissions, and an

appropriate length. Using the same data, our group attempted to predict submission/resubmission success using the average approval rating. In addition, to classifying successful posts, our group found interests in deleted posts. We noticed that deleted posts had a lower average approval rating. We trained a function where biases were assigned for the time of the submission and amount of comments. While optimizing our predictor we noticed that the time of a submission's had a greater impact on its approval rating compared to the amount of comments it possessed. This finding aligned with Lakkaraju, McAuley, and Leskovec analysis of the dataset.

# Conclusion

From our models and analysis above, our results and conclusions are clear. When analyzing the reddit data, posts with duplicate image ids can either be incredibly popular and successful or slide by unnoticed by the majority of users. Our model most notably combines the number of comments and the time posted to try and predict a post's score. When posting an image to reddit, a variety of factors come into play. The title, the time submitted, the subreddit thread in which the post was submitted and more influence the popularity of any given post. While no single feature can accurately predict a successful post, a combination of features can help to predict a post's success. From our analysis, it seems that sticking to the most popular subreddits is the easiest way to see success. We hope that our analysis of this data provides some useful insight on the mechanics of success on reddit.