JudgIt- A Reddit Popularity Predictor

Brian Lange
Northwestern University
715 Lincoln St.
Evanston, IL 60201
+1 (513) 254-6751
brianlange@
u.northwestern.edu

Wesley Sun
Northwestern University
720 Clark St Apt 1N
Evanston, IL 60201
+1 (707) 485-4787
Wesleysun2013@
u.northwestern.edu

Sheng Wu
Northwestern University
1856 Orrington Ave. Rm. 206
Evanston, IL 60201
+1 (405) 762-2905
Shengwu2014@
u.northwestern.edu

ABSTRACT

Online content aggregators like Reddit.com drive millions of pageviews, but out of thousands of stories submitted to them each day, only a select margin make the front page and gain exposure to the community. We attempt to learn the various importance of different features of a story at submission time in order to build a future score predictor for new stories, which content submitters could use to optimize the success of their submissions.

Categories and Subject Descriptors

I.2.6 **[Artificial Intelligence]**: Learning – *concept learning, knowledge acquisition, parameter learning.*

General Terms

Algorithms, Measurement, Experimentation

Keywords

Internet communities, content aggregators, popularity prediction, online learning

1. INTRODUCTION

Content aggregation is a huge business on the internet. Reddit alone gets 15 million unique visitors and 150 million pageviews per month. On Reddit, users post text or links to webpages that are voted on by other users, resulting in a score that determines how many future visitors will see the post. Successful Reddit posts generate tens of thousands of hits to a website, resulting in valuable ad revenue. Websites like Gawker for which web traffic is a main source of revenue want to know what to post, when to post it, and what words to use in a title in order to maximize how many people see it

2. PROJECT GOALS

We set out to predict the score of a Reddit post data from the Reddit API. Existing work in this area like [1] has used regression on early scores to predict future score, but we wanted to attempt building a score predictor using only data available about a story at post time. With this project, we wanted to explore several questions:

- Can we accurately predict scores using only data at post time from the Reddit API?
- To maximize score, what aspects of a post are the most important?
- What effect will different k-values have on the effectiveness of our learner?

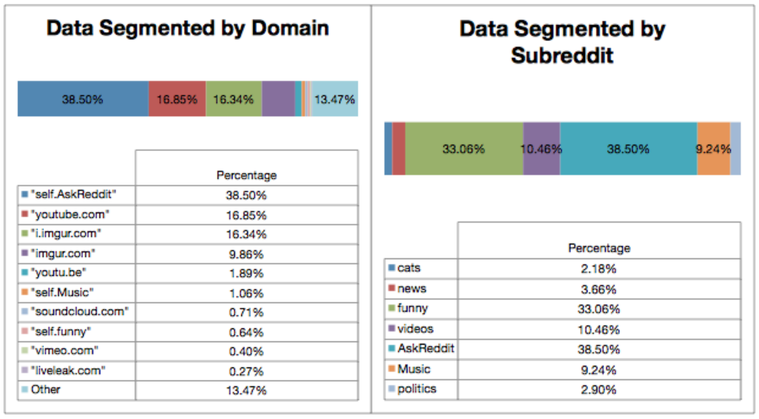
3. DATA

Data analyzed included all new posts from November 20 to December 7, 2013 on the following subreddits. Subreddits are communities within Reddit with a specific topic of focus.

- /r/AskReddit - a question and answer section of Reddit
- /r/News - general domestic and world news stories
- /r/Music - posts of songs and discussion about music in general
- /r/Videos - online videos of many types
- /r/Politics - political news stories and discussion
- /r/Funny - humorous stories and photos
- /r/Cats - mostly photographs of cats

These subreddits were selected because according to stattit.com, they are some of the highest traffic subreddits in terms of both submissions and voting, and they are fairly distinct in subject matter. /r/Cats is included as an example of a lower-traffic subreddit.

Data was collected from the Reddit API using an hourly scraper. 159,695 posts were scraped in total, and the average score (up votes – down votes) after 24 hours for those posts was 38. Additional features of the data used are visible in Figure 1.



	Percentage
🟦 "self.AskReddit"	38.50%
🟥 "youtube.com"	16.85%
🟩 "i.imgur.com"	16.34%
🟪 "imgur.com"	9.86%
🟦 "youtu.be"	1.89%
🟧 "self.Music"	1.06%
🟦 "soundcloud.com"	0.71%
🟥 "self.funny"	0.64%
🟩 "vimeo.com"	0.40%
🟪 "liveleak.com"	0.27%
🟦 Other	13.47%

	Percentage
🟦 cats	2.18%
🟥 news	3.66%
🟩 funny	33.06%
🟪 videos	10.46%
🟦 AskReddit	38.50%
🟧 Music	9.24%
🟦 politics	2.90%

Figure 1. Domain and Subreddit information about collected data

4. APPROACH

Based on our observations of popular Reddit posts, the vast majority of new stories become popular within 24 hours of submission. Knowing this, we sought to predict the score of a given Reddit submission after 24 hours using only data available when the story is first posted, detailed in section 4.2.

4.1 Regressor

The regressor used is a k-nearest neighbor (or k-NN) regressor implemented in Python. To predict popularity of a given Reddit post, the regressor starts by finding the k “nearest” posts in the training data to that post, based on a distance measure defined as a weighted linear sum of the metrics listed in section 4.3. After finding those k-nearest posts, it finds the mean score of them and returns that as the predicted score for the post.

4.2 Features

Our regressor considers the following features of a Reddit post, for the given reasons:

- Day of week posted: the bulk of users may only be online on certain days, so the day of week posted would affect visibility
- Hour of day posted: the bulk of users may only be online at certain times, so the time of day posted would affect visibility
- Top-level domain of the linked content: users may be more inclined to click submissions from sources they are familiar with
- Submission title: how descriptive or interesting a title is may determine how much attention it receives
- Submission author: reputable users may garner more attention
- Subreddit: different Subreddits have different community sizes, thus affecting how many votes it could have

4.3 Distance Metrics

To find distance between posts using the features above, we employed the following metrics:

- discrete metrics (d(a,b) = 0 if they are the same, d(a,b) = 1 if they are different) for subreddit
- string edit distance for author and domain
- number of days/hours between a and b (whichever is shorter with wraparound) for day of the week/hour of day
- Hamming distance between bag of word vectors for submission titles. For example, given two titles we can create a dictionary vector where each position in the vector represents a word that occurs at least once in the two titles, and each title can be described as a binary vector of the same length, where a 1 indicates that the word for that position is in the title and a 0 indicates that it is not. The number of positions for which those vectors differ is the distance between them.

4.4 Learning Weights

$$d = w_0 * day + w_1 * hour + w_2 * domain + w_3 * title + w_4 * author + w_5 * subreddit \qquad (1)$$

We sought to optimize our regressor by adjusting each of the weights in the linear sum of distance measures (1) used by our k-nearest-neighbor score predictor. Each weight could be one of an infinite number of values, so we narrowed the search space in a number of ways. We reduced the data to half its original size by random sample, and sampled 1/30th of those 79,847 posts for testing. We also assumed independence between them so we could vary each weight one at a time. Finally, we chose a set of discrete values (0, .25, .5, 1, 2, 4) to test over. We recorded mean squared error values for each weight at each discrete value, holding the other weights at unit value 1. We also tested the effect of different sizes of k by holding all weights at 1 and testing k values 1 through 5 on the same test set.

Overall this covers a very small percentage of possible values for the distance measure and makes potentially false assumptions that each of the metrics are independent, but it gives us an idea of which parts of the distance equation are important or irrelevant, and the general effect of increasing k.

5. RESULTS

After varying each weight among 6 values each as described in section 4.4, we chose the values that produced the lowest mean squared error values (see Figure 2) to produce the following optimized distance metric:

$$d = 1 * day + 0 * hour + 4 * domain + 0 * title + .25 * author + 0 * subreddit \qquad (2)$$

Or, with the 0 terms for hour, title, and subreddit removed:

$$d = 1 * day + 4 * domain + .25 * author \qquad (3)$$

We also found that increasing k on a distance metric with all weights set at 1 decreased mean squared error (see Figure 3), so we chose k=5 for our k value.

weight \ feature	day of week	hour of day	domain	title	author	subreddit
0	186378.06	143137.37	176738.8	144036.2	143055.1	160573.9
0.25	188932.34	151709.55	160588.8	155482.9	140176.6	164400.5
0.5	186176.76	165353.29	167411.3	188867	154373.4	168150
1	168343.65	168343.66	168343.7	168343.7	168343.7	168343.7
2	179420.49	155170.45	165988	189748.5	168101.3	175443.2
4	187420.54	145324.33	156614.4	205726.2	143598.9	166247

Legend	
205726.2	Maximum value
192616.28	
179506.36	
166396.44	
153286.52	
140176.6	Minimum Value

Color/number corresponds to mean squared error values in points^2

Figure 2. Results of weight learning calculations

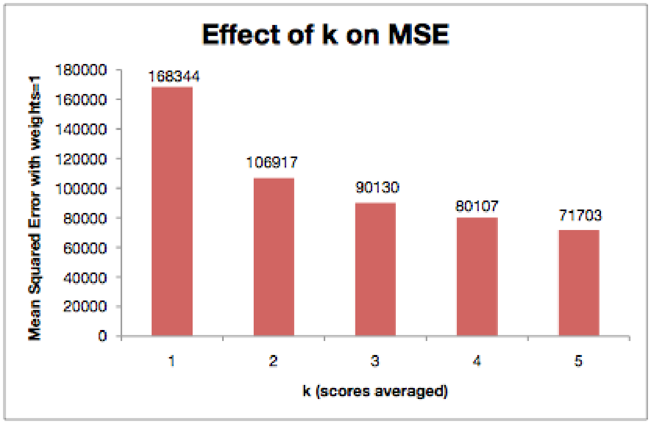


Figure 3. Tested k levels vs. MSE

Using 30-fold cross validation on the 50% of data not used during the weight learning stage, we tested the regressor with this distance measure and compared it with a baseline prior of using the average score of all stories (38) as the prediction for each story.

Table 1. Results of Cross-validation and paired data statistics

	k-NN Regressor MSE	Baseline MSE	Δ_{MSE}
Mean	70305.91601	61862.24378	8443.67223
St. Dev	15723.0188	15938.68843	2563.50759

Since on each fold of our cross validation we tested both the regressor and the baseline, we are able to employ a paired-sample t-test. Our null hypothesis H_0 is $\Delta_{MSE} = 0$, where Δ_{MSE} = the regressor mean squared error (a.k.a MSE) on a set of data - the baseline MSE on the same data. The 95% confidence interval we found for Δ_{MSE} is [7486.55, 9400.79], which means that we must reject our null hypothesis in favor of the alternative hypothesis $\Delta_{MSE} > 0$. In plainer terms, our regressor consistently performed poorer than the baseline on the same testing data.

6. DISCUSSION

The results of our weight learning were counterintuitive, claiming that hour of day posted had no effect on a story's eventual score, and that neither did the title or subreddit posted in, despite the fact that the subreddits we chose have widely varying areas of focus. Not having sufficient time to cross-validate these findings across multiple varying test sets, we find it hard to unequivocally accept these weights as optimal. We believe with a better approach (described in more detail in the next section) and more computation time that a regressor which performs better than baseline is possible, but our results are not an example of such a regressor.

7. IDEAS FOR IMPROVEMENT

7.1 Features

One feature not taken into account in our analysis was uniqueness of content. Posts of content which has already been posted on a subreddit may be (but aren't always) intentionally voted down by the community, and other studies of Reddit like [1] have found that on average a photo on Reddit has 10–12 reposts.

We also did not attempt to scrape and process the actual content linked to in submissions, which would provide much more unstructured text data which our regressor could take into account using a document similarity metric like cosine similarity on bag-of-words representations of the content.

Finally, we only compared users based on the actual text of their usernames- we could scrape their profiles to gain more information about a user's past submissions and general influence, and create a better metric for user similarity than the similarity of their names.

7.2 Learning Weights

Given the limited calculation time we had for this project, we heavily limited the search space of possible weights, assuming independence of each of the combined distance metrics and only trying 6 different values for each. We were also unable to do any sort of cross validation in our testing of weights, so we don't have a measure of confidence that allows us to firmly choose one weight over another. Finally, we only tested k-values 1 through 5; based on the results we saw, increasing k further may yield better results. Given more computation time and a more efficient implementation of k-NN, we could try additional weights, additional k values, and cross validation to potentially develop a better weighted combination that improves the predictive power of our algorithm.

8. ACKNOWLEDGMENTS

Our thanks to Bryan Pardo for his feedback on this project, advice on avant-garde jazz saxophonists, and his instruction in general.

9. REFERENCES

[1] Katyaini Himabindu Lakkaraju. Demystifying content popularity on Reddit. 2012. CS224W:Social and Information Network Analysis Group Project. Stanford University, Stanford, CA. http://www.stanford.edu/class/cs224w/upload/cs224w-025-final.v01.pdf