4th year project

Project name: Reddit popularity prediction

Author: Kevin Gillanders

Supervisor: Marija Bezbradica

Abstract:

Reddit is a social news site where content can be submitted by users as posts. It can then be voted on by other users, up or down. More popular content gets a higher score and is made more visible on the site, with content divided into subreddits, sub-forums for specialised content (e.g. Games for video game related content, aww for pictures of cute animals). This project set out to use information on individual posts and their respective subreddits to predict the eventual popularity, i.e. score, of the post. Overall, certain attributes, such as size of the subreddit the post occurs in, are relatively predictive of the score of the post and certain types of post, such as images, have higher scores in general. However, based on the attributes examined here, score determination seems to be a largely stochastic process. Further work with more complex methods may be carried out to extract more meaningful information on each post, for example the sentiment of the post. These more abstract, high level attributes of a post may prove to have a relatively large role in determining popularity, along with the more concrete attributes examined here.

Technical manual

Motivation

Reddit is a social news site which has become increasing popular and prominent as a source of internet content, scoring 8th in the Alexa ranking("Reddit.com traffic statistics"). Discussion on the site is primarily driven by what has the highest score as determined by users, as content visibility is largely determined by score.

The purpose of this project was to find what makes one post more popular than another, and to see if that popularity could be measured and predicted. Prediction was attempted by identifying key post features influencing popularity and integrating these into a predictive model. These features include elements such as submission time and whether or not there was any content associated with the post, for example an image.

The measure of popularity has wide reaching implications in human behaviour research and in advertising. The creation of viral content is fast becoming the quickest way to reach a large audience and determination of the correct starting features to adopt when posting on sites such as Reddit is poised to become a major area in marketing research.

Research

General research

A search was carried out for any previous work in the literature on popularity in Reddit. Work of this nature appears to be focussed predominantly on Twitter, with relatively little work pre-existing on Reddit. However, there is a small number of papers, and informal blogs, on the topic. On examining the work carried out in these reports, it was possible to implement some methods used and reproduce some of the results obtained.

Similar work was carried out by Randy Olson (Data driven guide). As this work had similar goals in mind, i.e. to find what drives popularity and thus give general advice on post features, it was

decided to attempt replication of these findings at a later stage of the project (discussed in 'Results').

Choosing a baseline comparison to judge the performance of the model, was based on methods used by Segall and Zamoschchin (Segall, Zamoschchin). Their control was based off of the average across their entire dataset, this was too broad. For my dataset I stuck with the average but I choose the average per subreddit, this I felt gave the baseline guess more of a chance to get the answer correct. This work focussed mainly on classification, my model on the other hand was primarily regression based. Classifying a numeric variable loses some level of information, were regression doesn't lose that precision, it also means it can be wrong a lot more. This group had more success in classifying posts than was achieved here. This could be attributed to multiple factors, such as use of different variables like title and different classifications, or potentially different bands as their methods for classification were not discussed in the paper.

On determining the variables which could be obtained, some general research was also carried out on the most appropriate statistical analyses to conduct and the best way to store and manipulate the data.

Possible test for association were investigated, Chi-squared was used for the stats test as it is a standard method to test the association between lists of categorical variables

SQLite3 was used to manipulate the database as, per the official SQLite documentation it is a lightweight library, which has all of the functionality which I require and no added extras. MySQL was briefly considered, however after reviewing the documentation a lot of the functionality was unnecessary for this project. For example, any of the server side functionality, was not required (MySQL documentation).

Continuing research on problems encountered

In addition to the initial base research performed on the existing work in this area, ongoing research was required in order to tackle some unexpected problems throughout the project. These generally related to the structure of the data, which was uncertain in the initial design stages, but also included limitations of the statistical tests used which were unknown until they were encountered. These problems will be discussed in further detail in the 'Problems solved' section, however a brief overview of the research behind the solutions used will be given here.

The problematic structure of the data first became apparent when attempting to put a class structure on the score variable (the dependent variable). There is an extreme positive skew to the data, with ~30% of scores equal to 0 or 1 and ~90% under 47, while the maximum value is over 18,000. This structure precluded the use of the standard methods to categorise a numeric variable which had been determined from initial research such as setting quartiles as class boundaries or applying the Freedman-Diaconis rule and therefore required further investigation, uncovering information about the use of logarithmic binning.

The data structure proved problematic once again at a relatively late stage, when the predictive model had been prepared and was being applied to the test data set. The high prior probability of scores at the extreme low end of the scale causes the model to adopt a sort of 'rule of majority' where only low scores are frequently predicted. This not only gave a high mean error, but is also largely useless for prediction of popularity on an intrinsic level. So, further research was required on the problem of unbalanced classes in regression (G. M. Weiss). One source (G. M. Weiss) suggested the use of undersampling, however this produces a new problem of altered prior probabilities in the training data relative to the test data. Further reading was then required into this problem and, although there are examples in the literature of methods to give a reasonable trade off between balancing classes and altering probabilities, they are complex to implement and do not account for non-binary categories, which was the case dealt with here.

Despite initially deciding on use of the chi-squared statistic to test associations between categorical variables, it became apparent on examining the p-values output from the test that there was a problem. Research on limitations of the test revealed that the chi-squared statistic is not appropriate for very large sample sizes as it tends to magnify the significance of very small associations. Further investigation provided a solution to the problem: a test of association strength on top of testing for association significance (Cramér, Harald). Cramér's V statistic is an extension of chi-squared which tests the strength of the association and which seemed suitable (Cramér, Harald)

Design

Based on the initial research carried out a general experimental design was produced to achieve the aims of the project.

The data must be first scraped from reddit, this can be achieved using the PRAW python library.

Cleaning the data and selecting an appropriate storage system, like SQLite or just a flat text file. Once the data has been stored, the key predictors must be found, this can be done using statistical tests such as the chi-squared or Pearson's R.

Once key predictors are isolated, they must be integrated into a regression model and so experimental design must include the requirements of the model used and how they will be met. For example, before giving a model any variables they must be encoded, as regression model cannot perform operations on non-numeric variables. The best form of encoding for non-numeric variables will need to be determined. The type of regression used will be ridge regression, as it will apply a penalty to variables determined to be collinear. As highly collinear variables will be removed at source, the zero-ing out of variables applied by lasso regression models will not be required and the use of ridge regression rather than, say, ordinary least squares will provide a nuanced penalising system for collinearity for less correlated variables.

The model will be compared to a baseline which, as mentioned in 'Research', will be formed by consistently guessing the average score for the subreddit the post occurs in. It will also be validated through use of the R^2 goodness-of-fit measure and calculation of mean error.

If popularity proves to be unsuitable for prediction in this manner, more general data analysis, such as comparing the average scores of groups of posts which differ in one attribute or another, may be carried out. This would then form the basis for general advice which could be given to users with a view to creating more successful posts.

Implementation

The project is entirely implemented through Python. To scrape reddit I used Praw(Python Reddit API Wrapper), which is a wrapper for reddit's API. The SQLite3 library was used for data storage and manipulation. The sci-kit learn library was used for machine learning methods, such as ridge regression and k-nearest neighbours, as well as tests of association like Pearson's r score

The project was implemented in several stages:

Data collection

The collection of data was the first step that was implemented. The data was collected using PRAW (Python Reddit API Wrapper), it provides functionality for collecting posts between two times. After six to eight months Reddit archives the content on the site, this means users can no longer cast votes or comment on the post. Sometime just after archiving was when the data was pulled from. The reasoning behind this was that the data was not going to change and the demographics were unlikely to have changed much in the interim period.

This was performed for post specific data, subreddit data and real time data.

Differences between collection methods:

1. Posts

   As this was the first attempt at scraping data, several mistakes were made. Firstly, the data was not obtained in an easily used format, but rather dumped into a text file as it was. This caused many problems, not least the amount of time that had to be invested to make the data usable. The steps that were required are discussed further down. In addition to this, no error handling was incorporated, which did not cause any issues in actuality, but could have also been a time sink in hindsight.

2. Subreddits

   The second pass at data scraping was more successful, having learned from the first attempt. Useful information was extracted programmatically from the returned information prior to writing to a file and error handling allowed the scraping script to run unsupervised despite dropped connections etc., which were more of a problem this time around.

3. Real time

   Real time scraping of the same posts over a 36 hour time period was carried out in search of any time point that seemed of particular importance for ‘seeding’ a high score. 5000 brand new posts were taken and their scores recorded with the unique post ID, and scores recorded for the same IDs every five minutes for the 6 hour period, at which point a new batch is selected and the process repeated.

Data cleaning and storing

Initially as a test the data was scraped and placed into a text file with no appropriate cleaning applied to it, eg removal of possible delimiters from the text. This would have been a minor setback, as the data could have been re-collected. However reddit changed the algorithm that is employed to calculate the score. This meant that this data no longer existed, which meant that it had to be cleaned. This was an arduous process as the file was over 10GB and with the titles being generated by the users could contain anything.

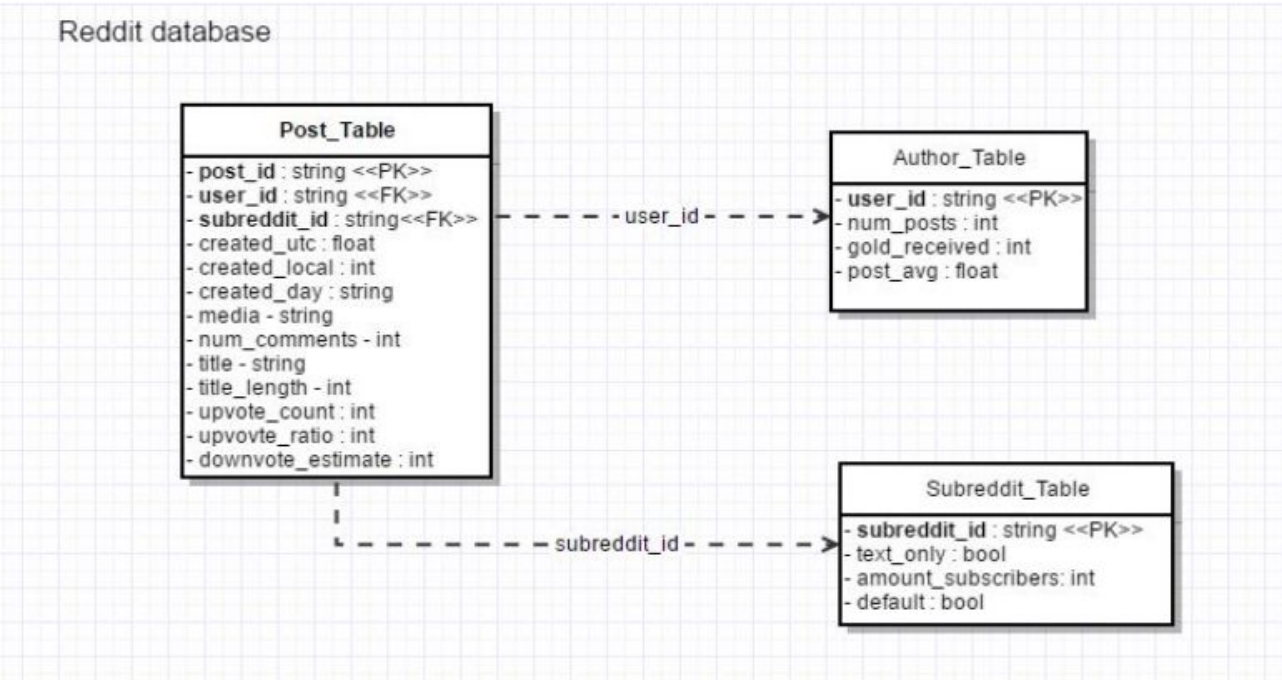
This was very challenging and required a lot of research into how regular expressions work. Eventually a regular expression was produced which could parse the entire file and break it into sections, using the lookahead assertion which is a non-consuming pattern matcher.

This is the regular expression: ', \'(?=\w*\':)'

it matches : '', 'any_words': ''

The information in the parentheses what is not consumed by the lookahead assertion as it is the actual content which is of interest .

Once the data was cleaned it was then placed into a csv file for temporary storage. From the csv the columns could be established for the database.



The final database ended up looking very similar to what had been laid out in the functional spec, however in final database did not have a user table as this would have been very time consuming to scrape from reddit and most likely would not have added much to the actual model.

Associations between variables

Once variables had been obtained for the posts, statistical tests were carried out in order to establish any relationships between the variables. The correct test to implement depends on the type of variables to be tested. Broadly speaking, variables can be viewed as either numeric, which is self-explanatory, or categorical, where data falls into one of several labelled categories which may or may not have intrinsic order. Several variables of both types were present among the post attributes investigated.

Determining association between variables of like type was relatively straightforward. In the case of categorical to categorical comparisons, the chi-squared test was implemented, but was found to be unhelpful for the purposes required due to large sample size; for this reason, Cramér's V was implemented as well to give a measure of association strength rather than just statistical significance. Similarly, numeric to numeric associations can be determined using Pearson's R. The R value gives a measure of association strength and the method returning this value also returns a p-value, therefore covering both association strength and statistical significance.

	agg_score	agg_num_comments	agg_gilded
agg_score	null	[0.0, 0.2608871458088671]	[0.0, 0.08126919850635117
agg_num_comments	[0.0, 0.2608871458088671]	null	[0.0, 0.07843377517983488
agg_gilded	[0.0, 0.08126919850635117]	[0.0, 0.07843377517983488]	null

Problems arose in comparing variables of differing types. It was determined that converting one type and then implementing the methods used above was the best route to take, however this raises questions of which variable type can be converted to the other with the least information lost and bias introduced. Categories of categorical data could theoretically be numbered and compared to numerics using R scores, however arbitrarily numbering them may lose any relationship that exists between the variables, and numbering them according to some artificial order (such as increasing values of the numeric variable concerned) introduces some bias as it artificially introduces a correlation between the variables. Due to these difficulties, it was decided to categorise the numeric variables concerned and apply Cramér's V. However, even this was not straightforward due to data structuring. Equal bands could not be applied to categorise the data due to extreme positive skew and logarithmic binning had to be applied in place of other categorising methods.

Creating a predictive model

Having determined the best variables to include, ridge regression was implemented to integrate the predictor variables into a single predictive model. Before this could be achieved, some work had to be carried out on categorical variables to make them compatible with ridge regression. This form of modelling can only take numeric values as predictor variables. Assigning the variable a numeric values will not work as the regression model will interpret this as having an order which will skew the prediction. The correct was to encode the variables is to assign them using what is known as “one hot encoding” (see table below). This assigns the categorical variable a list of booleans, the list is n elements long, where n is the number of categories. The corresponding boolean in the list is True or “hot”.

Sample	Human	Penguin	Octopus	Alien
1	1	0	0	0
2	1	0	0	0
3	0	1	0	0
4	0	0	1	0
5	0	0	0	1
6	0	0	1	0
7	0	0	0	1

Once this pre-processing work had been carried out, creation of a ridge regression model in scikit-learn is very straightforward, providing a certain portion of the dataset as a ‘training set’.

On carrying out some validation work, using R^2 and error values for the model, it was determined that the model was relatively poor at predicting scores within the ‘test set’ with the variables given. As it was suspected that this problem had arisen from the skewed structure of the data, some filtering was carried out on the training set in order to try and correct for the skew.

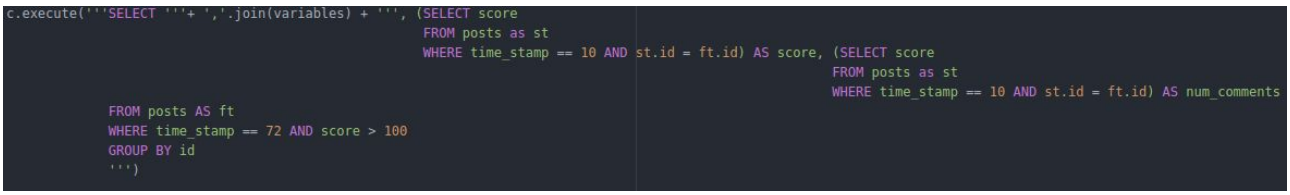
Time series data- scraping and analysis

The time series data provides a different look at the same data. The voting system on reddit works on a log scale with the first ten votes being worth a hundred times more then the next hundred, the first hundred being worth ten times more than the next and so on. This gives the first responders a disproportionate amount of control over the success of the content.

Collecting the time series data gave the ability for a continuous overview of the score and as such, the score at the first five minutes could be added to the model and

Sample code

Pulls data from the time series database, it takes the score and number of comments at the first hour, this data will then be fed into the regression model.



```
c.execute('''SELECT '''+ ','.join(variables) + ''', (SELECT score
                                                     FROM posts as st
                                                     WHERE time_stamp == 10 AND st.id = ft.id) AS score, (SELECT score
                                                                                                          FROM posts as st
                                                                                                          WHERE time_stamp == 10 AND st.id = ft.id) AS num_comments
          FROM posts AS ft
          WHERE time_stamp == 72 AND score > 100
          GROUP BY id
          ''')
```

Filters content by amount of times a particular subreddit has appeared in the dataset

```
def filterOnFrequentSubreddits(c, variables, count):
    # Fitering the data based on the number of
    # times the subreddit appeared in the database
    scores = []
    setData = []
    baseline = []
    c.execute('''SELECT '''+ ','.join(variables) +'''
                 FROM posts as ft JOIN subreddits AS st ON st.subreddit_id == ft.subreddit_id
                 WHERE ft.subreddit IN (SELECT subreddit
                                        FROM posts
                                        GROUP BY 1
                                        HAVING COUNT(subreddit) > 10)
                 LIMIT 200000
                 ''')
    insertData(c, setData, scores, baseline)
    return shuffleData(setData, scores, baseline, count)
```

The chi-squared/Cramér's V algorithm, implemented myself due to lack of support in python libraries

```
for x in col:
    for y in row:
        c.execute('''SELECT count(*)
                     FROM ''' + tableName + '''
                     WHERE ''' + var1 + ''' = ? AND ''' + var2 + ''' = ?;''', (x[0], y[0],))
        # Calculate corresponding expected value
        expected.append((x[1] / total) * y[1])
        for rows in c.fetchall():
            # Chi value given by:
            # sum(((observed[i]-expected[i])^2)/expected[i])
            chi += math.pow((rows[0] - expected[len(expected) - 1]), 2) / expected[len(expected) - 1]
            observed.append(rows[0])
print('chi : {} '.format(chi))
print('total : {}    len(rows)-1 : {}   len(col)-1 :  {}'.format(total, str(len(row) - 1), str(len(col) - 1)))
# Chi-square is very senstive to large sample size, carmer V(based on chi)
# gives the stength of the association between two variables
cramerV = math.sqrt((chi / total) / min(len(row) - 1, len(col) - 1))
print('Cramer V : {}'.format(str(cramerV)))
```

Problems solved

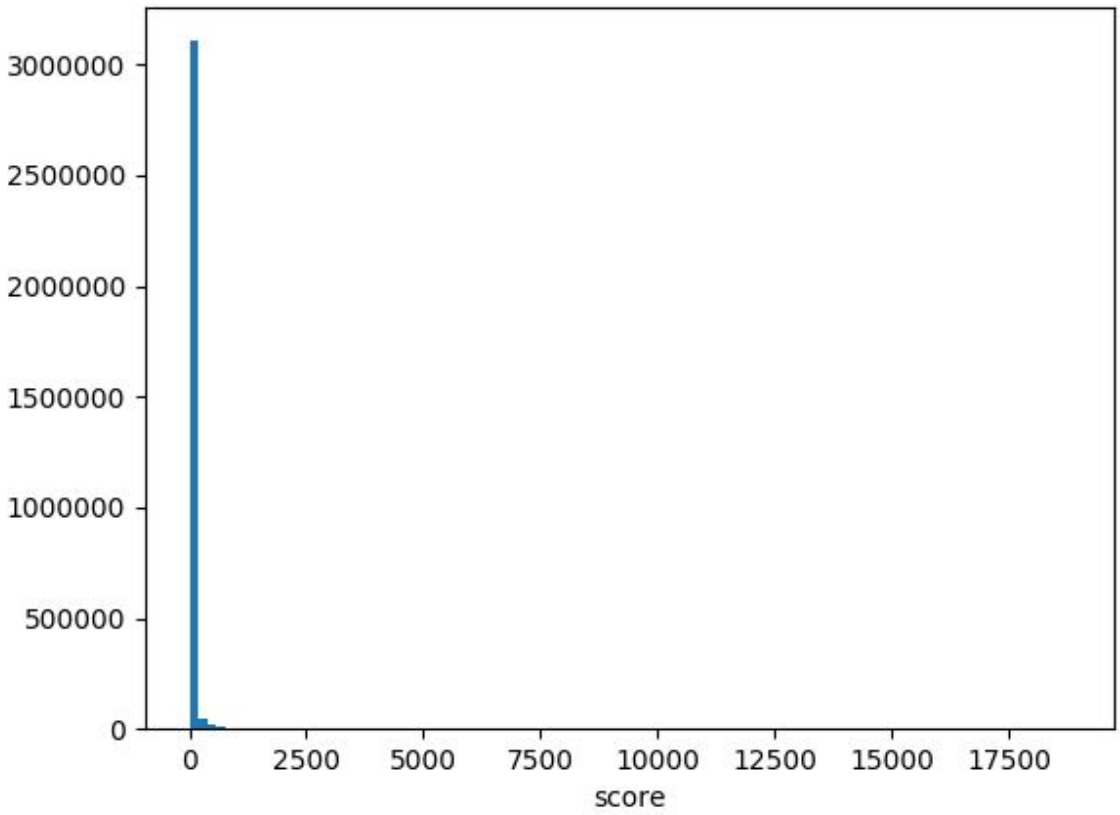
The first big problem which was solved over the course of this project was the cleaning of the initial batch of data. This data was meant to be used as a test to see how it would be stored, but reddit changed the algorithm which they use on their site. This meant that this data no longer existed.
Researching regular expressions and thinking on the issue required two days of work.
The lesson however was learned and some cleaning was performed with subsequent data, such as the removal of delimiters from the text, as well as the inclusion of error handling.

The Chi squared function in scipy didn't provide functionality required for this data, as that function assumed that the data was normally distributed, which it was not. As such an extension to the method had to be implemented, to allow for non-normally distributed data to be given to the method.

The Chi-squared test itself brought its own issues, as it is wholly unsuitable for getting any meaningful information in a dataset of this size. Once the population size being tested reaches a size of around 20,000, the test becomes overly sensitive to very small associations, with the sheer number of items inflating the significance of such associations. In order to make the test usable, a further extension was required to the standard function in the form of Cramér's V, which gives strength of association and is not sensitive to sample size.

The heavy skew in the data provided a continuing problem throughout the project. The first issue that was encountered was attempting to categorise the score. Over one third of the data was

either zero or one, with ninety percent being under 50 (see figure below). This presented issues with finding a meaningful way to categorise the data, and also in attempting to implement the regression model. In the case of categorising, a form of logarithmic binning provided a relatively satisfactory solution. However, in attempting to use this dataset to train and test a predictive model, the problem of skew in the data was never satisfactorily 'solved.' This is due to the ill effects of having large modifications to the training set in regression. For this reason, methods such as undersampling could possibly have been more trouble than they were worth as they alter the prior probabilities that are expected by the model. This was investigated to see if there might exist a method that would give a sensible balance between correcting skew and altering priors, however, as mentioned in the 'Research' section, no suitable method was uncovered.



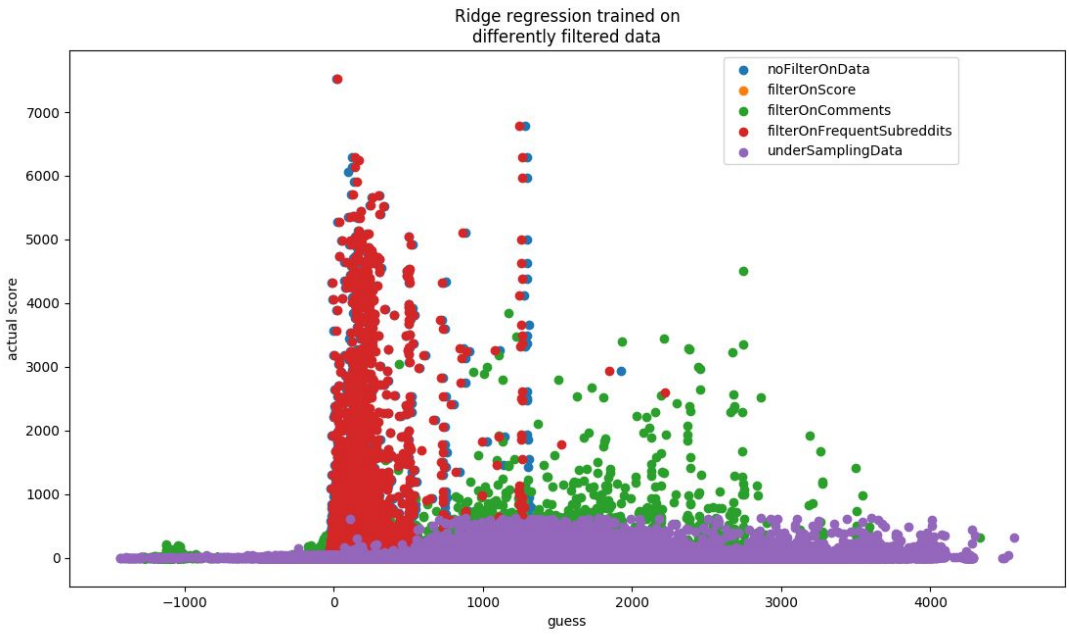
Results

Validation

Validating a statistical model is a very important step. The R^2 score is number which indicates the proportion of variance in the dependant variable which is explained by the predictor variables.

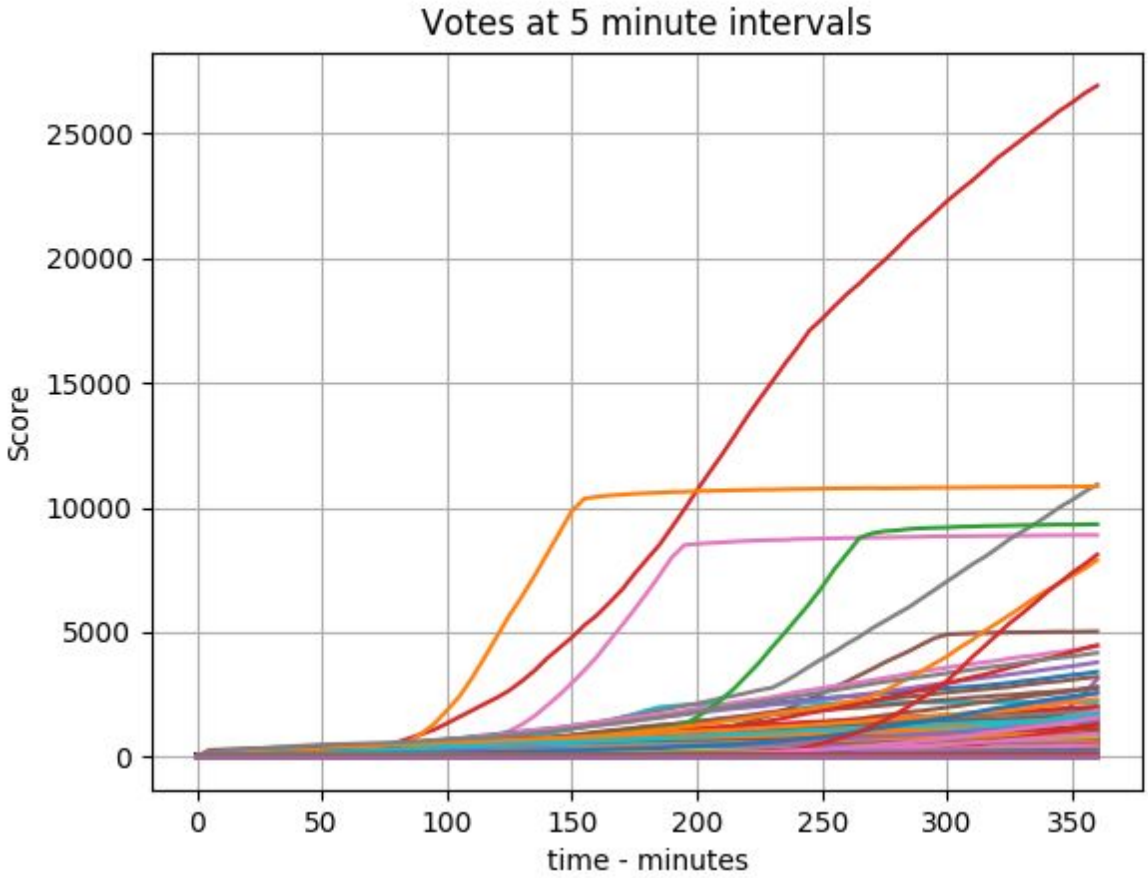
Type of filter	Base R^2 score	Base Mean Error	R^2 score	Mean Error
noFilterOnData	0.1079623602	49.7817064	0.1081409259	55.77876
filterOnScore	-12.702858298	26.5140655	-732.3491379832	262.46123
filterOnComments	-0.1806522309	32.6012143	-59.6236679268	371.06919
filterOnFrequentSubreddits	0.1139174628	47.1311575	0.113487356	53.31597
underSamplingData	-1.9834754998	28.645595	-858.9046176564	887.84349

As we can see from the table above the R^2 score is best in both noFilter and in filterOnFrequentSubreddits, this is because the training data is very similar to the actual test data. When more complex filtering starts the model starts to get worse as it starts to deviate from the actual data. From the below graph, the same data can be visualised more clearly. It can be seen that the the purple, which represents a model trained on under-sampled data, tends to guess higher values with greater frequency than any of the others, this comes from affecting the prior probabilities by including scores with an equal frequency, this makes it a lot more likely to get higher scores correctly.



Time series data

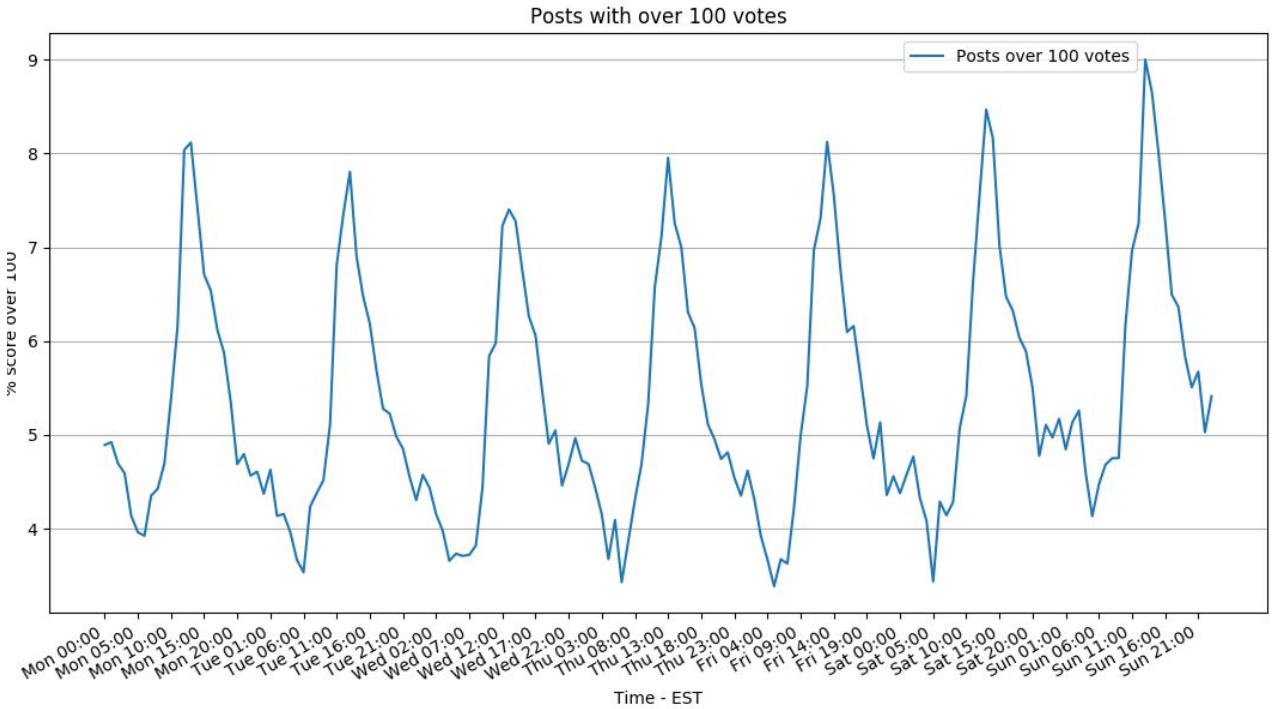
The below figure illustrates the results of the time series scraping of scores for 5000 posts. This scraping was carried out in order to investigate if there may be a 'seeding' time for popular posts eg. if a post gains 10 votes in the first half hour is it guaranteed to achieve a higher score. The data obtained do not seem to suggest that this is the case, however. All in all, seven posts break 5000 points, but they do not seem to have a common point of origin for when they began to gain traction, or even follow the same pattern of a life-cycle. The highest and second highest scoring posts both seem to begin climbing at a similar time, but the highest continues to climb hours after the second highest has peaked and levelled off. These results seem to suggest there is no set path to achieving an eventual high score, although a larger data set and tracking over a longer period of time may be required to be certain.



General Advice to Users

Although the model produced was sparingly predictive, some general insights can be gleaned from examining the data. Although many of these insights have been determined previously, this project does provide support for previous observations through replication and extension of the results obtained.

Work carried out by Dr. Randy Olson showed that the time most likely to result in a post with a score over 100 was posting it at 9AM EST (Data driven guide). He also concluded that "your post stands the best chance of becoming successful if you're sharing some form of visual media: images, GIFs, or videos (in that order)". Reading through his work the similarities between our two datasets, it became that I could attempt to replicate his results . Which I was successfully able to do.



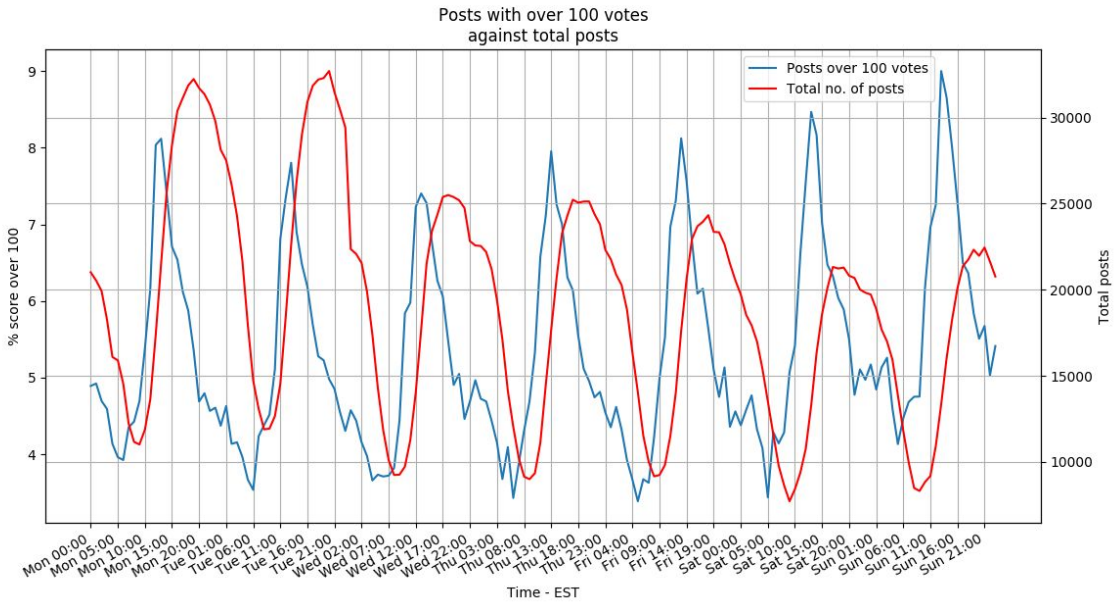
My data set also confirms Olsen's result that images give the highest score:

Rank	Media	Average score	Score sum	Total posted
1	image	125.26	61,342,491	489,704
2	mature_image	103.41	10,056,053	97,242
3	mature_video	65.37	204,606	3,130
4	social_media	61.52	2,677,773	43,530
5	article	30.77	20,783,385	675,514

6	video	26.24	6,001,513	228,749
7	mature	21.73	763,131	35,117
8	mature_social_media	19.67	363,038	18,452
9	none	12.14	18,903,964	1,556,853
10	mature_none	10.22	583,013	57,065

A post with an image attached has far and away the best chance of getting a high score. Posts with no media attached have the lowest average, despite being ranked second in terms of number of posts.

Olson found the pattern but didn't explain why 9AM EST is the best time to post.

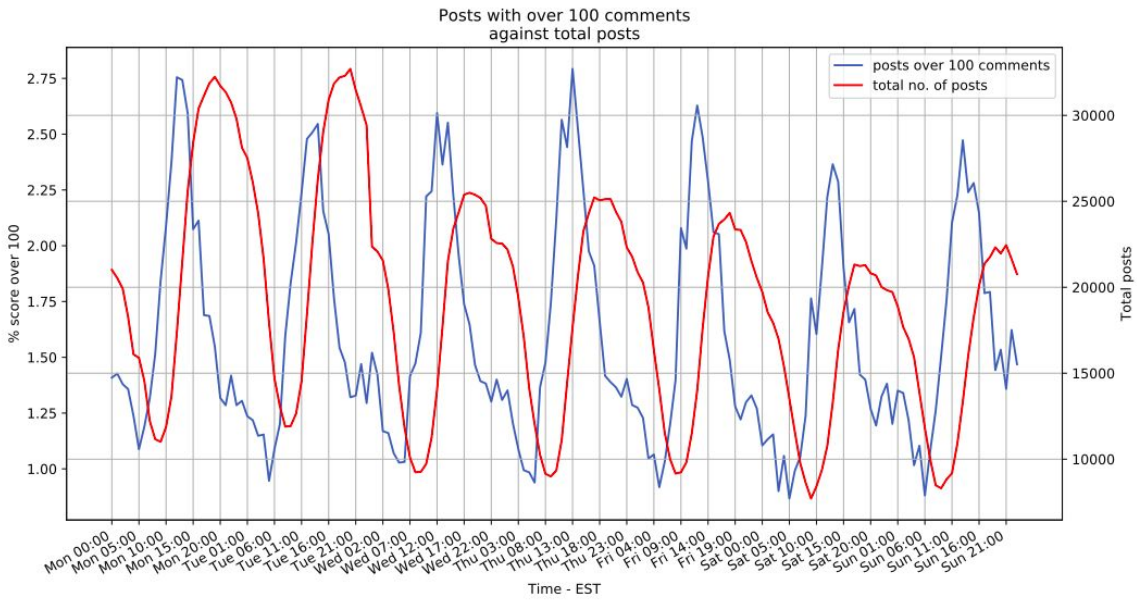


The amount of content posted whenever a user posts something I believe is the driving force behind whether the post will be successful or not.

So it is a combination between a lack of competition and when the main user base comes online, this provides the perfect environment for a post to succeed.

Our datasets both follow the same cycle, with the weekends, particularly Sunday being the most fruitful time to post something. There is also a significant drop off in the amount of content being posted on the weekend, this adds some credence to the theory that a lack of competition leads to more chance at success

I also decided to plot the number of comments a post receives in the same way. As the number of comments could also be used as a proxy for popularity



The number of comments follows a very similar cyclical pattern to the previous graph, however this graph has a significant drop off on the weekends. Commenting on a post is an intrinsically more involved activity and most likely people just have better things to do on the weekend than sit on reddit posting comments. It however follows the same window, like the previous graph, where a lack of competition provides the best window of opportunity.

Where to post: high subscriber count subs

Future work

Improvements to data set

Some improvements could be made to the dataset which could potentially improve the predictions obtained from the model. For example, in the case of undersampling, essentially all of the higher scoring posts were included in the training set in order to give a large enough training set while also having a relatively even sampling of the various score levels. This caused these larger values to be more or less absent in the test data, which most likely adversely affected the model's performance. In a much larger data set, undersampling might be a more viable way to

abrogate the effect of skew in the data as the test set would be better equipped to fairly test the model.

There is also a possibility that the large number of small/unpopular subreddits included may have introduced too much noise to the dataset i.e. too many low scoring posts that may only be low scoring because of the low exposure of the subreddit in which they were posted. To test this, the analysis could be repeated using the most popular subreddits, those with the largest subscriber base. There is some evidence that this approach may be worthwhile from the work already carried out in that number of subscribers is the most predictive variable of score of those looked at and filtering the training data so that it only includes the top subreddits marginally improves the model's performance over using the raw data. Potentially, score determination may be less vulnerable to stochastic effects in larger subreddits due to the size of the audience viewing the content.

The inclusion of extended time series data may also shed some light on the random nature of this problem. The time window in which real time scores were recorded was likely not long enough to cover the full 'life cycle' of long-lived post as not all posts recorded had ceased increasing in score at the time data collection was halted. Although this data improvement may not necessarily improve the model's ability to predict popularity, it might reveal some insights into how score increases over time, possibly allowing some greater understanding of random fluctuations that would give an opportunity to define the limits of prediction in the case of post scores.

Deeper content analysis

TFIDF on the titles and articles which are posted, this could be used to see if there are any hot button words or phrases which are disproportionality more popular than others. This could allow for more targeted content to the users of reddit, increasing chances of gaining traction

Sentiment analysis on the titles/articles posted could be used alongside TFIDF, to see if a particular emotion could be elicited which in combination with frequently used words could possibly yield a stronger result.

Perhaps some sort of image classification techniques could be employed on images on reddit, this could be used to establish a popularity pattern, and establish when some images do better than others. This coul have a subreddit specific focus, on r/pics (general purpose picture subreddit) this might get rid of a specific subreddit bias, like in r/aww(where cute animal pictures are the only pictures allowed)

Creation of predictive tools

A web front end could be hooked up to this model, where a user inputs what content they want to post and the model could return the optimal time along with their predicted score. This could be developed with an aim of helping advertisers focus their content and target it more directly at their potential customers.

Reference List

Reddit.com traffic statistics. (2017, May 24). Retrieved from http://www.alexa.com/siteinfo/reddit.com

Data driven guide. (2015, January 11). Retrieved from http://www.randalolson.com/2015/01/11/a-data-driven-guide-to-creating-successful-reddit-posts-redux/

Jordan Segall and Alex Zamoshchin *PREDICTING REDDIT POST POPULARITY* http://cs229.stanford.edu/proj2012/ZamoshchinSegall-PredictingRedditPostPopularity.pdf

When to use SQLite. (2017, May 25). Retrieved from http://www.sqlite.org/whentouse.html

MySQL documentation. (2017, May 25). Retrieved from https://dev.mysql.com/doc/connector-python/en/

Weiss, G.M. *Mining with Rarity: A Unifying Framework.* Sigkidd explorations, vol. 6, pp 7-19.

Cramér, Harald. 1946. *Mathematical Methods of Statistics. Princeton: Princeton University Press, page 282 (Chapter 21. The two-dimensional case).*