Predicting Issues Important to Citizens

Abhishek Ray
A53049953
a2ray@eng.ucsd.edu

***Abstract*— We consider the problem of identifying the issues which are most important to the citizens by predicting which issue would receive the highest number of votes. For this project, we use both the description of the issue as submitted by the users along with the metadata associated with the issue, to make our prediction.**

I. Introduction

311 is a platform where citizens can submit issues to the city administration or the government. Once an issue has been posted, other citizens can vote and make comments on the issue so that the government officials have a degree of awareness regarding which issues are the most important to the citizens.

This particular dataset was hosted as part of a Kaggle challenge[1] whose main goal was to be able to quantify and predict how people react to specific 311 issues. In this project, we focused on being able to predict what makes a particular issue urgent and which issues the citizens care about the most by analyzing different factors associated with the data such as the description of the issue, the location from where the issue was created, and the time the issue was created amongst others.

We hope to able to come up with a model which can predict the number of votes a particular issue will receive from other citizens based on the votes received by similar issues in the past. We used a regression based model based on both text and metadata associated with the data and come up with a predictive model which can help predict which issues are urgent.

However, since the Kaggle competition had already finished by the time we started working on the project, we weren't able to compare our result neither against the official test set nor against the leaderboard.

II. Related Work

The dataset was available as part of a Kaggle challenge. One of the winning teams published a report based on their model and results[2]. Their final model was a weighted average of a linear regression model and a segmented ensemble of tree-based gradient boosting regression models and linear regression models. We weren't able to test our method against the winning method because the competition had already closed.

There isn't any academic work which tackled a similar problem of trying to predict which issues would be urgent for users.

There has been prior work in the field of predicting movie ratings and revenues using Movie Reviews and meta-data[3][4]. They describe a model where they take into account both the review text of the movie as well as metadata about the movie to predict the opening weekend revenue.

A lot of the ideas for our project were drawn from this work as these methodologies were directly applicable to our dataset as well.

III. Predictive Task

In this project, we try to predict the number of votes a particular issue might get based on the description of the issue as well as other meta-data associated with an issue such as time and date of creation, source, type etc.

To evaluate the model, we will be using the Mean Squared Error (MSE) between the actual and predicted number of votes. Two baseline models will be used to compare against our model:

- Predicts the mean of the number of votes received during the training set.
- Predicts a random number of votes between the minimum and maximum number of votes seen in the training set.

To evaluate our mode, we performed a 70:30 split on the data. The first part of the data was used as the training set and the second half as the test set.

IV. Dataset Analysis

A. Dataset

The dataset used for this project was available as part of the Kaggle challenge. It contains 223,129 issues with each issue containing a summary, a description, the number of votes, comments and views received, the latitude and longitude where the issue was created, the time and date it was created, the tag for the issue as well as the source of the issue (Web, Mobile, Map Widget).

We also had to preprocess the data to fill in missing rows. A lot of issues either had their source or their tag type missing. We filled these missing rows by creating a "missing" type so we could differentiate them.

Fig. 1. Votes vs Comments

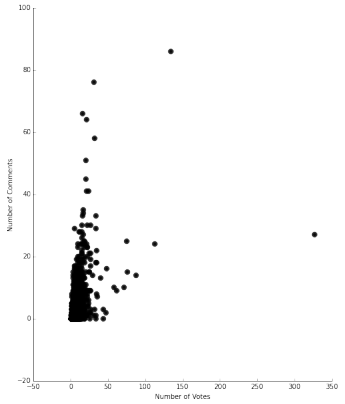
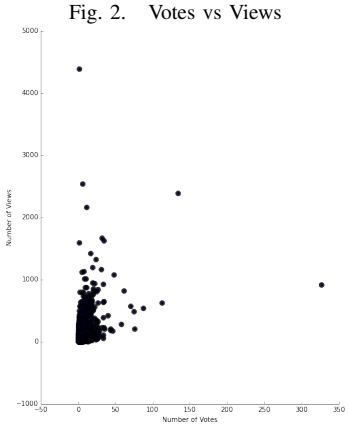


Fig. 2. Votes vs Views



B. Analysis

The first analysis carried out on the dataset was to compare the relation between the number of votes and the number of comments and the number of views. Figures 1 and 2 show this relation. Surprisingly, there doesn't seem to be a correlation between either the number of votes and the number of comments or the number of votes and the number of views.

Next, we analyzed whether location played any role in the number of votes an issue received. The dataset contained data from 4 different cities. Our findings are presented in Table 1. Issues created in New Haven, CT on average received more votes whereas issues created in Chicago, IL received the least amount of votes.

Next, we explored how the number of votes changed depending on the type of tag associated with each issue.

TABLE I
VOTES VS LOCATION

City	Mean Votes
Oakland, CA	2.846
Richmond, VA	2.388
Chicago, IL	1.015
New Haven, CT	3.104

TABLE II
VOTES VS TAG TYPE

Tag Type	Mean Votes
Missing (No Tag)	1.116
Hydrant	1.265
Other	2
Overgrowth	2.355

TABLE III
VOTES VS TAG TYPE

Tag Type	Mean Votes
Drug Dealing	5.160
Prostitution	5.269
Bad Driving	6.318
Public Concern	11

There were more than 300 different tags used in the dataset for the different issues. Tables 2 and 3 show the tags with the least and most votes. As expected, serious issue tags such as Drug Dealing and Public concern received the highest votes whereas issues which didn't have any tags or had generic tags such as "Other" ended up not receiving too many votes.

We then analyzed the relation between the day of the week the issue was created and the number of votes. As shown in Table 4, issues created on the Weekend received more votes as compared to issues created in the middle of the week.

We also explored the relation between the time of creation of the issue and the number of votes. As shown in Figure 3, issues created in the latter half of the day (after 4 pm) tend to receive more votes as compared to issues created early in the morning.

Next, we analyzed if the source of the Issue had any effect on the number of votes received. From Table 5, we can see that issues created using the Mobile Site or the Map Widget received more votes than the other sources. Some of the sources were omitted from the table for brevity.

We then analyzed the relation between between the length of the text description and the number of votes for a particular issue. The results are shows in Figure 4.

We also performed the Latent Dirichlet allocation algorithm on the text review to see if there were any particular topics

TABLE IV
VOTES VS DAY OF THE WEEK

Day of the Week	Mean Votes
Monday	1.521
Tuesday	1.5102
Wednesday	1.479
Thursday	1.499
Friday	1.485
Saturday	1.562
Sunday	1.624

Fig. 3. Votes vs Time of Day

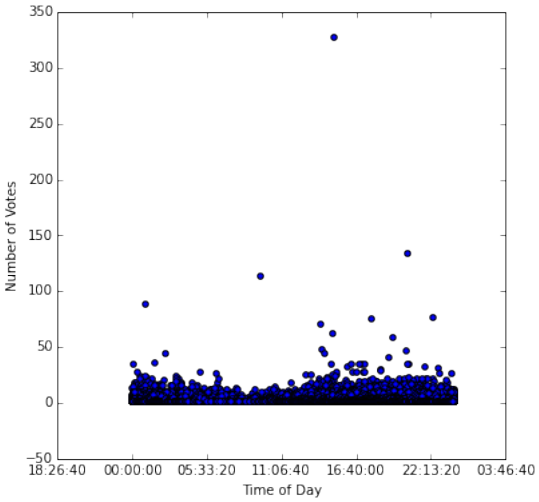


Fig. 4. Votes vs Length of Description

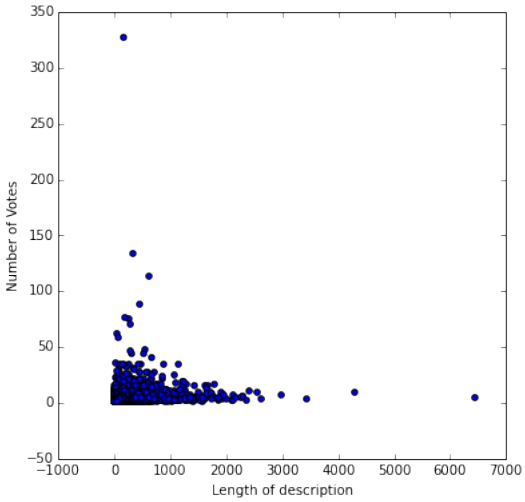


TABLE V
VOTES VS SOURCE

Source	Mean Votes
Remote API	1.009
City Initiated	1.961
Missing (No Source)	2.512
Web	3.259
Map Widget	3.685
Mobile Site	3.685

in the issues which received higher votes. Some of the topics being discussed in the description of the issues is depicted in Table 6 and Table 7 shows which are the prominent topics being discussed in the issues which get the maximum votes.

V. FEATURE SELECTION

After the exploratory analysis on the dataset, we looked for correlation between all the different features under consideration and the target label i.e. the number of votes. We also computed the correlation between each of the features under consideration so that we could eliminate redundant features.

TABLE VI
TOPICS IN THE DESCRIPTION

Topic Number	Topic
1	street cars city parking car people new
2	street potholes ave road pothole block lane
3	tree limbs brush debris alley needs branches
4	trash emergency garbage truck resolved dumping property

TABLE VII
TOPICS FOR ISSUES WITH MOST VOTES

Topic Number	Votes
1	134
1	62
2	58
1	48

The features were divided into two categories:

- Meta-data Features
  1) **Number of Views & Number of Comments** - Both the number of views and the number of comments had a negative correlation with the number of votes. These features were determined to be not very useful as features as they span Y (number of votes) values.

  2) **Day & Time of Creation** - As we saw in the previous section, issues created during the weekend (Saturday & Sunday), on average received more votes than issues created during the week. Similarly, issues created during the later half of the day (4 pm - 12 am), received on average more votes than issues created earlier in the day.

     To use these features, we had preprocess the data. The data initially contained one column which contained the Date and Time of creation. That column was split into two separate columns where one contained whether the issue was created during the earlier part of the day or at night. Similarly, the other column contained whether the issue was created during on a weekday or a weekend.

  3) **Location** - After analyzing the dataset, we found that there were issues from 4 different locations in the dataset. Issues created in New Haven, CT on average received the highest votes (3.104) whereas issues created in Chicago, IL received only 1.105 votes.

     The dataset initially contained latitude/longitude pairs for each row of data. A Reverse Geo-coding API was used to preprocess the data and convert

it to a City/State pair. This processed data was then used as a feature vector.

4) **Source** - As we saw in the previous section, there were multiple sources using which an issue could be created. Issues created using different sources varied widely in the amount of votes in they received. For example, an issue created using the mobile site on average received 3.68 votes whereas an issue create via the remote API received only 1.009 votes on average.

5) **Tag Type** - Tag Types such as Public Concern and Prostitution received more votes on average than tag types which were missing. This told us that Tag Type would be an important feature in our model.

- Text Features - To extract text features, we used the n-gram approach. We considered both bigrams and trigrams.

VI. Model

We use a regression model to predict the number of votes a particular issue would receive. A regression based model was used as we wanted to estimate the relationship amongst the features and the number of votes. Moreover, since the problem statement involved a prediction and not a classification, it is more suited to a Regression model.

We applied three different regression techniques:

1) **Linear Least Squares Regression** - Linear Regression is a linear model and attempts to model the relationship between the input and output variables by fitting a linear equation. One of the benefits is performance. However, the disadvantage of using linear regression is that it only looks at linear relationships between the variables.

2) **Ridge Regression** - Ridge Regression is similar to Linear Regression. However, it eliminates some of the problems of Linear Regression by introducing a regularization parameter. One of the advantages of using Ridge Regression would be since it introduces a penalty on the coefficients, it might generate a model which might be more generic than Linear Regression and gives better results.

3) **Random Forest Regression** - Random Forests are an ensemble learning method that construct multiple decision trees during training and output the mean prediction of the individual forests. Random Forest would work better than the previous two models if the relationship between the variables in non-linear. Tree-based models can usually approximate functions with any shape.

TABLE VIII
MSE on Metadata Features

Model	MSE
Mean Predictor	1.780
Random Predictor	35272.68
Linear Regression (using all Features)	1.12
Ridge Regression (using all Features)	1.00
Random Forest Regression (using all Features)	1.47

TABLE IX
Time on Metadata Features

Model	Time(in seconds)
Ridge Regression (using all Features)	0.431
Random Forest Regression (using all Features)	1.285

VII. Results

A. Metadata Features

Table 8 shows the Mean Squared Error achieved by each of the models. As expected, a random predictor performs the worst with its MSE being orders of magnitude higher than any of the other models. The linear models perform better than Random Forest with the Ridge Regression model performing the best. One of the reasons for this is that there is probably a linear relation between input and output variables. The Ridge regression model performed better than linear regression as it introduces a regularization term and penalized the linear model when a particular coefficient became too large.

Table 9 shows the amount of time each of the two models (Ridge & Random Forest Regression) took on the test data. Ridge Regression, as expected, was faster than Random Forest.

Next, we also evaluated the MSE for some combinations of features to verify which of the features were actually predictive and which of the features were redundant. As we can see from Table 10, the Source and Tag Type were the most predictive features whereas as the Day/Time of creation feature wasn't as predictive and proved to be redundant. We got the same MSE even after removing it from the feature set.

B. Text Features

Next, we analyzed our regression models on the various text features associated with the dataset. We ran three different experiments, one using just the summary, one with the just

TABLE X
MSE on Metadata Features using Ridge Regression

Feature	MSE
Source	1.12
Source + Weekday/Time	1.12
Source + Lat/Long	1.08
Source + Tag Type	1.07
Description + Tag Type	1.12
Source + Description + Tag Type	1.02

TABLE XI
MSE FOR SUMMARY

Model	MSE
Ridge Regression	1.81
Linear Regression	1.82
Random Forest	2.14

TABLE XII
MSE FOR DESCRIPTION

Model	MSE
Ridge Regression	2.01
Linear Regression	2.03
Random Forest	2.30

the description and then the combination of the two. The results are depicted in Tables 11,12 and 13.

A model based on just the summary text features was a better indicator of the number of votes an issue would receive. One of the possible reasons for this could be that since a lot of issues didn't have any description associated with them, the summary was a better indicator for users when they voted on issues.

C. Text + Metadata

After generating models discussed above, we tried to combine the prediction from both the text and metadata models to provide a new prediction which would be a weighted average of the predictions of the individual models. The motivation for doing so was that we could achieve a better model which would incorporate the strengths of both the individual models.

$Prediction_{combination}$ = α * $Prediction_{text}$ + $(1-\alpha)$ * $Prediction_{metadata}$
Where $\alpha \in [0, 1]$
We ran the algorithm for different values of α to try and find if there was a particular value at which this combined model would perform better than the individual model based on the metadata. However, we were unable to find any such value of α.

A probable reason for this is that the metadata features are a better predictor of the number of votes and thus, there isn't any linear combination of the two approaches which would give a better result. It is possible that there could be a non-linear relationship between the models which could perform better.

TABLE XIII
MSE FOR SUMMARY + DESCRIPTION

Model	MSE
Ridge Regression	2.00
Linear Regression	2.03
Random Forest	2.29

VIII. CONCLUSION

We conclude that a ridge regression based model performed better than both Linear (ordinary least squares) Regression and Random Forest model for the dataset used for this project. This is probably because there is a linear relation between the variables and thus the linear models perform better. We also saw that metadata features were a better predictor of the number of votes as compared to the text features. We also tried a combination of the two models but we saw that there wasn't any linear combination of the two models which performed better than the metadata predictor. References are important to the reader; therefore, each citation must be complete and correct. If at all possible, references should be commonly available publications.

REFERENCES

[1] http://www.kaggle.com/c/see-click-predict-fix
[2] http://bryangregory.com/Kaggle/DocumentationforSeeClickFix.pdf
[3] Mahesh Joshi, Dipanjan Das, Kevin Gimpel, and Noah A. Smith. Movie Reviews and Revenues: An Experiment in Text Regression. Proc of the North American Chapter of the Associatoin for Computational Linguistics Human Language Technologies Conference. (2010.)
[4] Aju Thalappillil Scaria, Rose Marie Philip, Sagar V Mehta (svmehta), Predicting Star Ratings of Movie Review Comments.