CIS 520 FINAL PROJECT

NEURAL KNIGHTS

RESTAURANT RATING PREDICTION

Pranav Sahay Vishnu Purushothaman Sreenivasan Vibhor Nigam

1 ABSTRACT

With the rapid growth of user based reviews in the online communities, it has become increasingly important to extract useful information from the vast textual information available online.

The task is to predict the rating of a restaurant from a short descriptive review written by a user. The dataset chosen for this particular task are the reviews obtained from the *yelp* website. The problem is formulated as a regression task rather than a classification task and (*Root mean square error) RMSE* is measured. *Correlation test* was used for feature selection and feature transformation algorithms, including *TFIDF, correlation based feature weighting with sentiment analysis*, *PCA* and classification algorithms, including *Logistic regression*, *Naïve Bayes*, *Perceptrons, SVM*, *Random forests etc.* were employed and the optimal combination was chosen. The final algorithm employed was a combination of *Logistic Regression with Naïve Bayes* and was able to achieve significantly low RMSE in the out of sample test data and was significantly faster than most of the other algorithms employed.

2 OVERVIEW

The steps involved in the project can be minimally broken down to four major steps as shown in the flow chart in *fig.1*.

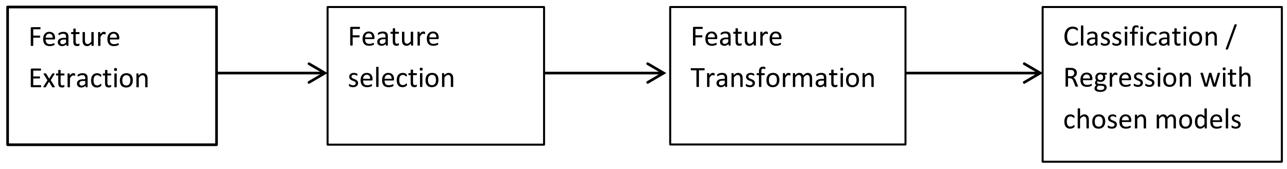


Fig.1 Overview of the Project

The feature extraction involves obtaining parameters or attributes for each of the short textual review and the method employed in this project was the *bag-of-words model*. Due to the large size of the vocabulary there is a dire need for feature selection to pick the most correlated words. The features in the *bag-of-words model* convey useful information regarding the frequency of the word, that is the histogram of the word for every review, but often classifiers may perform significantly better on a transformed feature space, for example a binary feature space whether a word is present or not. Finally we apply our classification or regression models to predict the rating of a review.

3 EXPLANATION OF THE DATASET

The dataset for this project is the reviews of restaurants taken from the *YELP* website (www.yelp.com). The models were first trained on a set 25000 training samples with a *bag-of-words model* including a vocabulary of 56,835 words.

"The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity."[1]

Also provided were the metadata for each review as to the ID of the reviewer and the business ID of the restaurants being classified. There were in total 19766 unique user IDs and 6564 unique business IDs in the training set. There was a validation set called the *quiz set* containing 5000 reviews with the same metadata with new users and businesses. The reviews of the validation set and metadata was observable but the corresponding ratings were unknown. Finally to evaluate the true accuracy of the predictions 10000 completely out of sample reviews and ratings was taken as the *test set* to evaluate the model proposed.

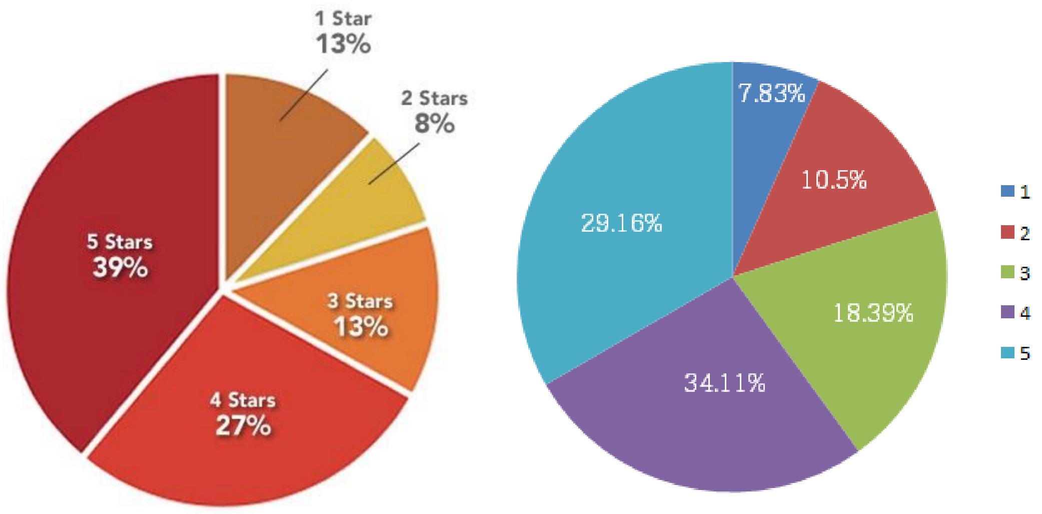


Figure 2. The pie chart on the left shows the distribution of ratings on the entire yelp website (obtained from the yelp site) and the one on the right shows the distribution of the ratings in the given training set.

4 METHODOLOGY EMPLOYED

This report explores the methodology explored by our group in chronological fashion, the feature selection and prediction models used at each milestone of this project are explained and analyzed in each section separately. The accuracy of the predictions was measured in RMSE. According the formula below:

$$RMSE = \sqrt{\frac{\sum_{\{t=1\}}^{n}(y_t - \widehat{y_t})^2}{n}}$$

where,
y_t - true labels
$\widehat{y_t}$ - predicted labels
n - number of samples

4.1 MILESTONE 1 – RMSE 1.3

For this milestone we did not perform any feature selection, feature transformation or dimensionality reduction.

4.1.1 PREDICTION MODELS

The prediction model used for this baseline was Multinomial Naïve Bayes. Other models such as lasso, ridge and random forests failed to build with such a huge dataset without any feature selection or dimensionality reduction.

4.1.1.1 NAÏVE BAYES

The classification model used for the first baseline of 1.3 RMSE was Multinomial Naïve Bayes. Since the data is in the form of counts, it is an ideal candidate to be trained with multinomial Naïve Bayes. The entire dataset of 25000 training sample with 56,835 features was used for this baseline without any feature selection or feature transformation. On 10-fold cross validation Multinomial Naïve Bayes gave an average RMSE of 1.0543 and mean misclassification error of 0.5122. The model acquired an RMSE of 1.0408 on the validation set. As this well above the required baseline no further processing on the data was performed.

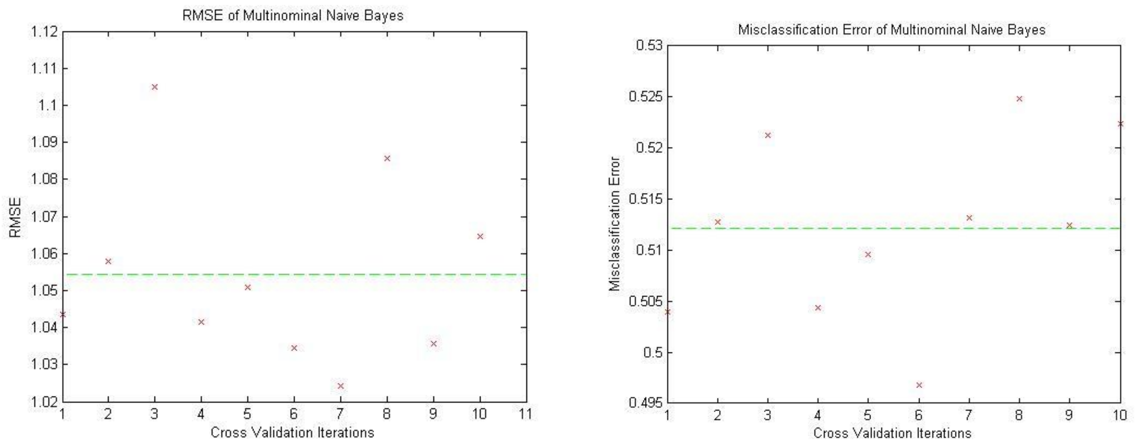


Figure.3 In the left hand side plot, x shows the RMSE of Multinomial Naïve Bayes calculated at each iteration of cross-validation and green line shows the mean RMSE of 1.0543 and the right hand side plot shows the misclassification error of Multinomial Naïve Bayes in the same format with a mean misclassification error of 0.5122

4.2 MILESTONE 2 – RMSE 1

For this part of the project milestone we performed a crude feature selection technique, also performed feature selection using correlation analysis of the data and also performed dimensionality reduction (SVD or PCA equivalently). The prediction models attempted for this baseline were Perceptrons, Regression, Naïve Bayes, LibLinear and SVM, adaboost etc.

4.2.1 Feature Selection and Feature transformation

4.2.1.1 Crude Feature Selection

We initially performed a crude feature selection technique which removed all the features or words in the vocabulary which never occurred in any of the reviews. This left behind 46093 features. We also performed a crude feature selection technique which removed all the features or words in the vocabulary which occurred only up to 2 times in the entire dataset. This left behind 20926 features.

4.2.1.2 TF-IDF

tf–idf, term frequency–inverse document frequency, is a numerical statistic that reflects how important a word is to a document in a collection or corpus.[2] It is often used as a weighting factor in information retrieval and text mining. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus(collection of documents), which helps to control for the fact that some words are generally more common than others. For instance, through this approach a less weightage is given to a more commonly appearing word such as “the” which has a higher frequency in the document but minimal effect on the final prediction as it has a high frequency in every document.

The formula used for calculating the tf-idf value is

$$tf - idf = \log\ (1 + count) \times \log \frac{N}{N_w}$$

$count$ = count of the words in a single document,

N = total no. of documents,

N_w= No of documents in which the word occurred.

4.2.1.2 CORRELATION TEST

The best feature selection technique for a bag of words model was correlation analysis on the data. The Pearson correlation coefficient was computed for every feature against the labels and the top features were selected for each algorithm by using cross validation. The correlation test produced excellent result as the top correlated words on the vocabulary were *‘worst’*, *‘bad’* etc.

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

where,

cov - Covariance
σ_X - Standard deviation of X
σ_Y - Standard deviation of Y

4.2.1.3 PCA or SVD

PCA can be computed by mean centering the data and performing an SVD on it and taking the first few columns (principal components) of the V matrix produced by SVD as the principal components.

$$X = U\,\Lambda V^T$$

where,
X is the data matrix

Since the data is very sparse mean centering the data does not shift the coordinates significantly so the SVD of the data can be approximated as the PCA for the data. PCA was performed on the data where all the features which occurred only 2 times or less in the entire dataset was removed (20926 features).

4.2.2 PREDICTION MODELS

4.2.2.1 PERCEPTRONS

We used Perceptrons with a passive aggressive update algorithm on the data with 46093 features(removing the features with zero counts in all the documents). The model was built as a 1 vs each, in which we obtained the weights for rating 1 vs 2, 2 vs 3 and so on. So we derived 10 weights and used it for prediction on the test set. The cross validated error (5-fold) was 1.0056 and the error on the validation set was 0.9997.

4.2.2.2 NAÏVE BAYES

Multinomial Naïve Bayes was built using the reduced feature set (46093). The cross validated error (10 fold) was 1.0473 and the error on the validation was 1.0224.

4.2.2.3 REGRESSION

Regression does not work well on this data as the data is rank deficient. We also tried removing any observation with all feature counts as 0. But the data is still rank deficient and predictions were completely wrong. We still could not perform lasso or ridge as the dataset was too large.

4.2.2.4 ADABOOST WITH PERCEPTRONS AND NAÏVE BAYES

Since we developed several models to predict the output, the logical step is to combine the predictions in the most optimum way by weighting the predictions of each model. In order to do this, Perceptrons and Naïve Bayes were assumed as weak classifiers and their predictions were pushed to Adaboost and upon boosting for 1000 rounds the in sample training error was significantly small (0.2023). The model was severely overfit and gave an error of 1.0867 on the validation set and so Adaboost was dropped as a method of weighting the predictions.

4.2.2.4 LIBLINEAR

LIBLINEAR is a linear classifier for data with millions of instances and features[3]. It is an optimized SVM with a linear kernel. Several different variations of the data were tried for the different algorithms in Liblinear. On cross validation we found that L2- regularized logistic regression on the primal worked best for most forms of the data. And best combination was obtained, by performing Liblinear on the PCA'ed tf-idf data using only the features which occurred at least thrice in the entire dataset. We chose the scores of the first 500 principal components. On 5 fold cross validation we got an RMSE of 0.9801.

4.2.2.4 INTERSECTION KERNEL SVM

We first tried to perform intersection kernel SVM using the libsvm library[6] with the entire dataset by using a model similar to Perceptrons as 1vs each model, that is 1vs 2, 1 vs 3 and so on. Therefore 10 models was required to be built and the model took a very long time to be built. On an i-5 2nd generation 2.3Ghz Processor with 8 GB of RAM, it took approximately 15 hours to be computed and the prediction for 5000 samples took approximately 2 hours. The final in sample error RMSE was computed to 1.2 and the model was dropped for both inaccuracy and infeasibility for prediction. Then we performed intersection Kernel SVM using only the top 2200 correlated features (after removing all the features which occurred less than 3 times in the entire dataset). The reason for choosing only the top 2200 correlated features was because we cross validated the correlated features data with LibLinear and this gave the least RMSE for the non-transformed dataset, as we require a histogram of features for the intersection kernel. We also performed a numerosity reduction on the data by choosing only the data which was either rated useful, cool or funny at least once; this reduced the dataset to 12472 observations. Using this model we got a cross validated error of 0.98 on the dataset.

4.2.2.5 INTERSECTION KERNEL SVM, PERCEPTRONS, NAÏVE BAYES, LIBLINEAR

First Naïve Bayes was computed with the top 2300 correlated features (after removing the features which occurred less than 3 times in the entire dataset). This increased the accuracy of Naïve Bayes and it gave a cross-validated error (10 fold) of around 0.98. The Perceptron, LibLinear and the Intersection Kernel SVM models explained above were used. By taking a voted majority (in case of a tie we took the prediction by intersection kernel SVM), we were able to get an RMSE of 0.9863 on the validation set.

4.2.2.6 FIND SIMILAR ALGORITHM[4]

The Find Similar Algorithm applied was a variant of Rocchio's method for relevance feedback. It basically calculates an approximate centroid for all labels by individually calculating the means of feature weights for each label. The 'tf-idf' scores of the data were used as the weights of the features.

$$c_{ij} = \frac{\sum_{i \in rel} x_{ij}}{n_r}$$

$$C_i = [c_{i1}\ c_{i2}\ c_{i3} \ldots\ldots c_{im}]\ for\ i = 1,2,3,4,5$$

There is no explicit error minimization involved in computing the Find Similar weights. Thus, the learning time is negligible, except for taking the sum of weights from positive examples of each category. Test instances are classified by calculating their tf-idf weights' distances with the approximate centroids and the closest rating centroid is chosen as the prediction. The **in-sample RMSE** of this implementation was **1.25** and hence, it was dropped since we had already reached an RMSE of 0.968. We believe the poor performance can be attributed to the crude approximation employed in calculating the centroids.

After the aforementioned experiments, we employed a naïve method of Sentiment Analysis on binary (absence/presence) data and previously employed correlation feature selection to come up with the Final Prediction model.

4.3. Final Submission

Having tried a bunch of learning techniques and feature selection methods (as uploaded in the unused code folder), we submitted the final submission which includes a combination of Multinomial Logistic Regression and Naïve Bayes. We speculate that these methods performed better (gave lower RMSE) than the previous versions on account of the data modifications applied as well as using probabilistically weighted models rather than conventional category prediction.

As stated before a crude selection was done by eliminating features which appeared less than 3 times in the entire training dataset. Another noteworthy change employed in the data set was that a Binary representation of the word counts was used, i.e. the presence of word was accounted as '1' and the absence as '0'. This, we believe helped us in two ways. First, it brings the verbose reviews and the reticent on the same level. Furthermore, it takes care of the relevance of the stop words like 'THE', 'IS', 'WAS' etc. since these can be present a lot of times in a review. **Thus, making them binary reduced the correlation of stop-words with the label vector.**

4.3.1. Multinomial Logistic Regression

Data modification and feature selection

The training of Logistic regression is a time taking process especially for data with large number of features. Final Logistic Regression model is based on a clever tweaking of data which makes use of the sentiment dictionary [5]. To all the 'positive words' present after crude selection, we multiplied it by +2 and to all the 'negative words' we multiplied -3. We arrived at +2 and -3 after analyzing different integer values (for simplicity) and the sum of each review (a (25000x1) vector) and its correlation to the Training label vector, a **correlation value of 0.54**. The indices of positive and negative words are being passed in the final model. Over the combined dataset of training and quiz set with values of 1,0,+2 and -3, a PCA was performed to get scores for the entire. These top 100 principal components are passed in the final model for test data modification in the make_final_prediction.m.

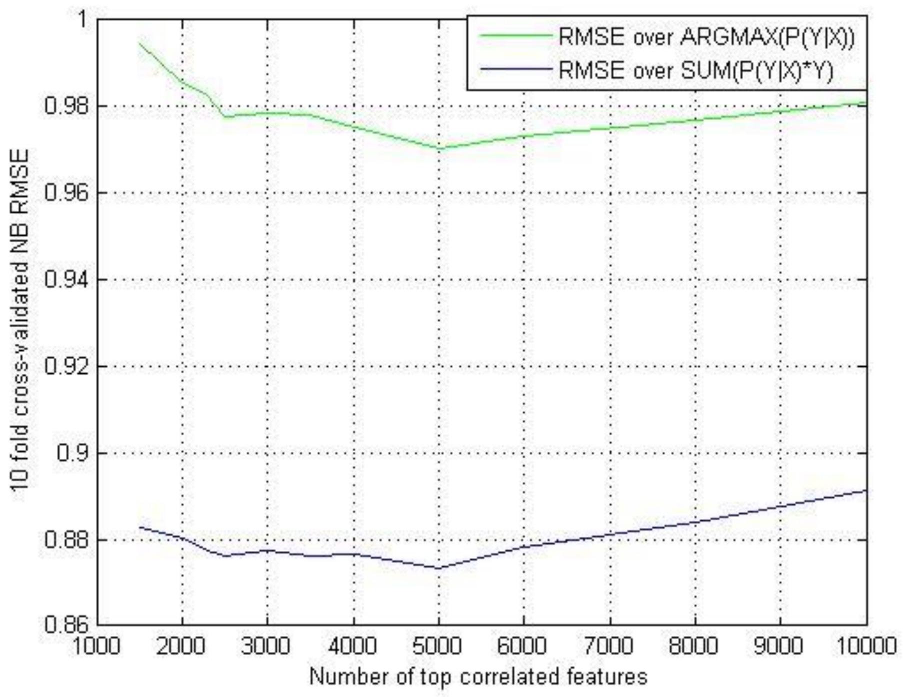
Model training:
This PCAed scores of the training data set was then used for training a Multinomial logistic regression model for nominal responses.

Logistic Regression Prediction:
The prediction for the logistic regression is done by probabilistically weighted sum of the rating values and not the conventional category classification by multinomial logistic regression. This was done as per the advice mentioned in the project slides since, a weighted combination helps us reduce the RMSE. This method on the Validation Quiz set gave an **RMSE of 0.8702.**

4.3.2. Naïve Bayes

Data modification and feature selection

For Naïve Bayes too, the results were better for the binary (absence/presence) data than the word counts data provided. For the final prediction, we performed a two vector correlation test with each feature vector and the training label and selected the top 5000 correlated features on the basis of cross validation as shown below. We pass the correlation indices in the final model and select the top 5000.



Model training:

The Naïve Bayes model was then trained on this modified data and stored. We pass this model as a model parameter.

Naïve Bayes Prediction:

The NB final prediction is also probabilistically weighted sum of the rating values as in the case Logistic Regression. The **cross-validated RMSE itself was 0.876** as shown in the figure and hence a combined model of both Logistic Regression and Naïve Bayes was the logical next step.

4.3.3 NEURAL NETWORKS

Neural Networks was applied on the correlation data (positive words multiplied by +2 and negative multiplied by -3) and then scores of top 100 principal components were taken. The model was tried with **2, 10** and **20** hidden layers. Most of the predictions done by the model were of label 4 and 5 and the in-sample **RMSE was 0.2** for the model, which showed that there was a significant case of over fitting.

4.3.4 K-MEANS

K-means was applied on the same correlation data (positive words multiplied by +2 and negative multiplied by -3) and then scores of top 100 principal components were taken. We tried clustering it into 5 different groups but **4 out of 5 different groups had a mode of 4**. This is probably attributed to the fact that majority of the data has a label of 4. We tried running k-means several times with different starting points but still the results were found to be similar.

4.3.5 RANDOM FORESTS

We tried to build random forests also on the same data set i.e. correlation data (positive words multiplied by +2 and negative multiplied by -3) and then scores of top 100 principal components were taken. We tried building the **random forests with 50, 100, 150 and 300 decision trees** and the **RMSE for these varied from 1.35 to 1.25**. Also the time taken to build and predict the model was too large. Due to these performance issues the idea was not carried out further.

4.3.6 Final Prediction

The final prediction is a weighted combination of the Logistic Regression and Naïve Bayes predictions. The weights were calculated using **Ridge Regression** on random chunks of 5000 training labels and corresponding predictions and taking a mean of the acquired values to avoid over-fitting. The penalty chosen in Ridge regression was 0.01. The regression weights calculated were **0.6506 for Naïve Bayes predictions** and **0.3356 for Logistic Regression predictions**. However, it was noted that the values changed negligibly for ridge regression penalty coefficients varying from 10^{-4} to 10.
The final **RMSE on the validation set achieved was 0.8434** and for the **Final test set as per the Final Leaderboard the RMSE was 0.8503 with 10,000 predictions in 290 seconds.** The speed of the prediction can be attributed to simple models like Logistic Regression and Naïve Bayes. However, as evident these simple models perform significantly well with the aforementioned data modifications.

Table 1. RMSE and Cross-Validation error of the different Algorithms employed which performed significantly well on the dataset

S.No	**Methods Used**	**Cross Validation Error / In Sample Error on Training Data**	**Final RMSE on Quiz set**	**Data Used**
1.	**Naïve Bayes**	**1.0543** (In Sample)	**1.0408**	Original data set. No feature Selection
2.	**Perceptron** and **Naïve Bayes** (prediction based on taking the mean and rounding it off for classification)	**1.01** (Cross Validation)	**1.005**	Data with features removed whose sum across all observation was zero
3.	**Liblinear, Perceptron** and **Naïve Bayes** (prediction made by taking a voting/mode between all three)		**0.9863**	Data with features removed whose sum across all observation was zero
4.	**Liblinear,Perceptron**,**Naïve Bayes** and **intersection kernel SVM**		**0.9683**	**Liblinear** on tf-idf pca-ed top 500 (tf-idf and pca was performed on top 29000 features after removing all features with sum of 2 over observations), **Peceptrons** on data set with features removed whose sum was zero across all observations, **Naïve Bayes** on binary top 2500 correlated data, **Intersection Kernel** on top 2000 correlated data
5.	**Sentiment Analysis** and **Logistic Regression**	**0.874**	**0.8702**	**Sentiment Analysis** done using sentiment dictionary on top **100 pca-ed correlation data**
6.	**Sentiment Analysis** and **Naïve Bayes**	**0.876**		**Sentiment Analysis** done using sentiment dictionary on **top 5000 binary data**

7.	**Ridge regression** on **Logistic Regression and Naïve Bayes**		**0.8434**	Did **ridge regression** on the predictions and then chose a weight of **0.6506 for Naïve Bayes** and **0.3356 for Logistic Regression**
8.	**Ridge regression** on **Logistic Regression and Naïve Bayes**		**0.8503 (out of sample/final test set)**	Did **ridge regression** on the predictions and then chose a weight of **0.6506 for Naïve Bayes** and **0.3356 for Logistic Regression**

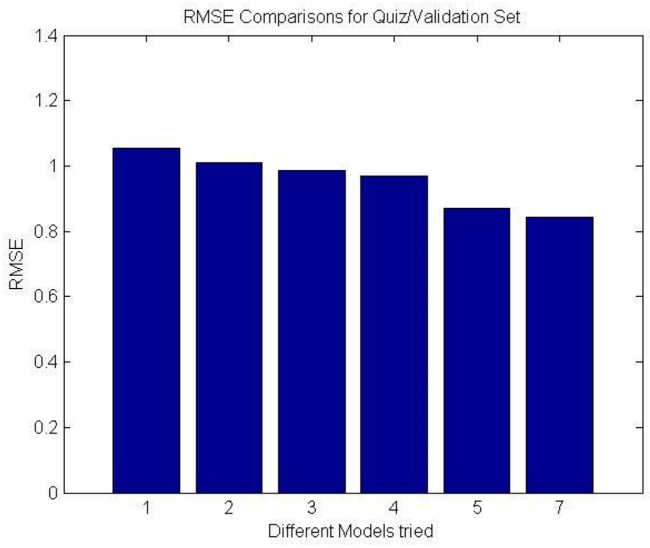


Figure 4. RMSE for the Validation set. Here 1 2 3 5 and 7 are the different models specified in the table above

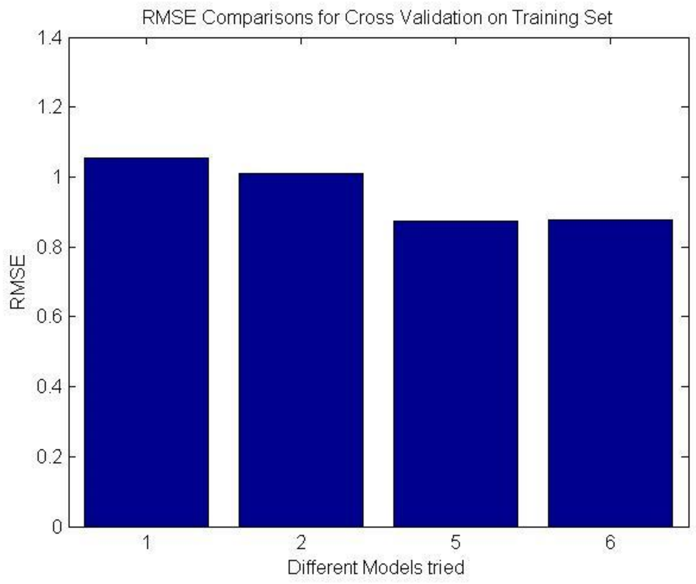


Figure 4. Cross Validation Error. Here 1 2 5 and 6 are the different models specified in the table above

5 REFERENECES

[1] Wikipedia

[2] Rajaraman, A.; Ullman, J. D. (2011). "Data Mining". Mining of Massive Datasets. pp. 1–17. doi:10.1017/CBO9781139058452.002. ISBN 9781139058452. edit

[3] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A Library for Large Linear Classification, Journal of Machine Learning Research 9(2008), 1871-1874.

[4] Platt, Dumais, Heckerman and Sahami. "Inductive Learning Algorithms and Representations for Text Categorization", Microsoft Research

[5] Minqing Hu and Bing Liu. "Mining and Summarizing Customer Reviews." Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Aug 22-25, 2004, Seattle, Washington, USA.

[6] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011.