**Would Your Review Be Upvoted in Yelp？**

Yurui Mu (ym1495), Jingyi Su (js5991), Lei Xu (lx557), Yidi Zhang (yz3464)

[**Understanding Phase**](#_d5kwc37rr4se)[**2**](#_d5kwc37rr4se)

[Business Understanding](#_vpsrbbtnbu36) [2](#_vpsrbbtnbu36)

[Data Understanding](#_eahyrpdb67b4) [3](#_eahyrpdb67b4)

[Dataset Information:](#_ltqrrzv40go3) [3](#_ltqrrzv40go3)

[Feature and target variable:](#_cyf641r827sj) [3](#_cyf641r827sj)

[Assumptions about the Dataset:](#_3b76voog5hgn) [4](#_3b76voog5hgn)

[Dataset Preparation](#_gv18u6xt7iy0) [4](#_gv18u6xt7iy0)

[Phase 1: Convert json file to csv](#_tuo6dq9tn7z7) [5](#_tuo6dq9tn7z7)

[Phase 2: Deal with problems raised by json conversion and filter individuals related to restaurant.](#_rspm0qujlvhk) [5](#_rspm0qujlvhk)

[Phase 3: Feature selection on datasets with business sense and create new features/ target variable](#_c8jc8hucjhlk) [5](#_c8jc8hucjhlk)

[Phase 4: Recode features, merge three dataset on the basis of ‘review’ and deal with missing data](#_nqw26c4cpfeq) [6](#_nqw26c4cpfeq)

[Phase 5: feature selection based on mutual information and feature engineering.](#_1ib5hfrwncx5) [7](#_1ib5hfrwncx5)

[**Modelling**](#_m1pfdzovgzqi)[**7**](#_m1pfdzovgzqi)

[Modeling Introduction:](#_atf5wp3xztt) [7](#_atf5wp3xztt)

[Baseline Models](#_idavs3f9472l) [8](#_idavs3f9472l)

[Possible Models and Experiments](#_6dt67spi37au) [9](#_6dt67spi37au)

[1. Decision Tree Model](#_4rfa5ac8fgos) [9](#_4rfa5ac8fgos)

[2. Support Vector Machine Model](#_ygy7h5wd9od9) [10](#_ygy7h5wd9od9)

[3. Naïve Bayes Model](#_f3bg10qqox7v) [11](#_f3bg10qqox7v)

[4. Logistic Regression Model](#_y877d45uie0m) [12](#_y877d45uie0m)

[Modelling Summary](#_4ww5l711sb4k) [14](#_4ww5l711sb4k)

[**Deployment**](#_nznig4cvehjo)[**15**](#_nznig4cvehjo)

# **Understanding Phase**

## **Business Understanding**

As a popular crowd-sourced review site, Yelp provides important information for users to understand and evaluate local businesses. However, Yelp has been receiving critics about the legitimacy of reviews published on its site. Thus, it is important for Yelp to identify meaningful reviews and promote them to a more prominent display. Otherwise, Yelp might start to lose popularity among users as well as among local businesses, resulting in profit reduction from business advertising, which has been the major source of Yelp’s revenue[[1]](#footnote-0).

Our project aims to identify reviews which are likely to be considered as useful by Yelp users. The data mining process is based on the content of the review text, the reviewed business characteristics, and reviewer related information. As it is independent with whether the particular review was historically upvoted or not, our model will determine the usefulness of a review once it is written and published. Yelp can then promote the useful reviews identified by our model instantaneously. At the same time, Yelp can filter out the unimportant reviews in the end of the review lists.

As a result of deployment of our model, Yelp would be able to display the most relevant, useful and up-to-date reviews at the most prominent position for each restaurant. Yelp users can find useful reviews easily without going through pages of uninteresting reviews. Once users are able to gain all the necessary information about a restaurant conveniently through Yelp, they will use Yelp whenever they are deciding where to eat. Thus, the churn rate will be minimized. The huge amount of user counts will become a critical competitive advantage for Yelp in generating advertising revenues. Hence, even though our model may not directly generate money for Yelp, it saves money from churn related costs and potentially increases Yelp’s revenue.

## **Data Understanding**

### **Dataset Information:**

Our dataset comes from Yelp Academic Dataset Challenge[[2]](#footnote-1).The whole package originally consists of five datasets and a photo set. Based on our assumption and business case, we will only use ‘business’, ‘review’ and ‘user’ datasets for this project (Table1). To see details about these variables, please refer to Appendix. Due to our business case, we would only use cases related to restaurant business.

Table 1. Basic Fact about Original Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** | **Num of Rows** | **Num of Columns** | **Case (Row)** |
| Business | 85901 | 98 | one business, for example, a restaurant |
| User | 686556 | 23 | one Yelp user |
| Review | 2685066 | 10 | one review, with regard to certain user in certain business |

#### 

### **Feature and target variable:**

1. **Instance**: since our task is estimating the possibility of a review gets acknowledged based on the information we have, each review would be the instance we need to work on. To be more specific, each instance would be a review in restaurant category. We have 1630712 instances in total.
2. **Feature:** To fully describe each review, we will use all valuable information from ‘review’ , ‘user’ and ‘business’ dataset as potential features. Before fitting any model, feature selection will be applied.
3. **Target variable:** There is no variable in the dataset ready to be our target variable. However, ‘votes’ attributes is a good representative for the usefulness of review. If a review has been voted, it means this review is useful in some aspect. Detailed methods will be covered in data preparation phase. The transformed target variable has distribution in Table 2.

### **Assumptions about the Dataset:**

Assuming here Yelp has provided us with all the valid information they have, i.e. there won’t be any discrepancies during sampling. Therefore, no selection bias.

Table 2.Distribution of Target Variable\*

|  |  |  |
| --- | --- | --- |
| **Target Variable** | **Label 0: Not Being Acknowledged** | **Label 1: Being Acknowledged** |
| Num of Instances | 847581 | 783131 |

## **Dataset Preparation**

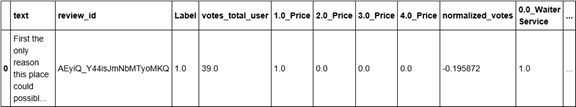
To increase the accessibility to data and structure it in feature vector format (Figure 1) where each row is an instance and columns are predictive features/target variable, we went through the procedure below. (Figure2)

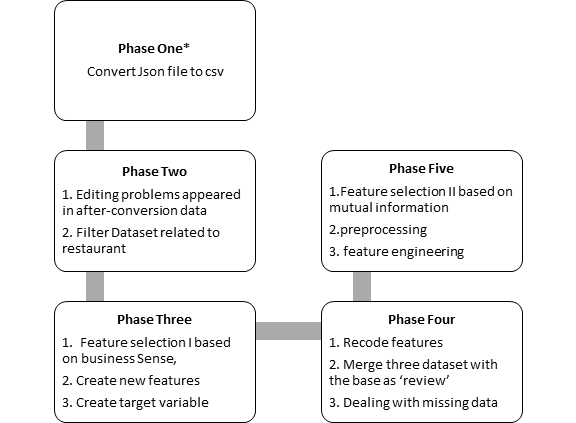
Figure 1. Clip of Final Dataset

Figure 2.Data Preparation Procedure

### **Phase 1: Convert json file to csv**

To make the dataset manageable for all team members, we wrote code and converted the file from json to csv.

### **Phase 2: Deal with problems raised by json conversion and filter individuals related to restaurant.**

There are certain problems coming with data conversion: for example, the binary string variables after conversion all starts with ‘b’’ and columns name in ‘business’ all starts with attributes. Therefore, some cleaning needs to be applied to format the dataset.

Also, due to our goal, we only need to keep the data relate to restaurant business. By using the ‘user\_id’, ‘business\_id’ and ‘review\_id’ as keys, we select the rows fit our purpose from the three datasets individually.

### Phase 3: Feature selection on datasets with business sense and create new features/ target variable

Due to the thinking of computation efficiency, we decided to do necessary cleaning and feature creation before data combination.

1. **Drop feature**s: During this phase, potentially less informative variables and variables with no variation or all missing in three datasets were removed respectively based on business sense. For example, we delete the open and close hour every day in ‘business’ dataset, as we think these details has nothing to do with whether a specific review for the restaurant will be voted or not.
2. **Feature creation I:** To make full use of our variables, we create some new features. For example, in the ‘user’ dataset, from the ‘friends’ column where entries are the list of friends id one user has, we create new variable ‘friends\_count’ which is the count of friends one user has.
3. **Target variable creation:** From our business case, our target variable would be whether one review is voted or not. We created a dummy variable here to indicate this: 1 as ‘was voted’ and 0 as ‘was not voted’. Yelp provides three kinds of votes in the ‘review’ dataset, namely the number of ‘votes\_funny’, ’votes\_useful’ and ’votes\_cool’. To get our target variable, we simply add them up to get ‘votes\_total’, then code 0 for reviews with 0 ‘votes\_total’ and 1 for others.

### **Phase 4: Recode features, merge three dataset on the basis of ‘review’ and deal with missing data**

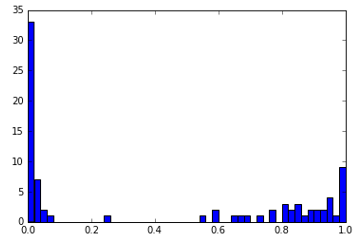
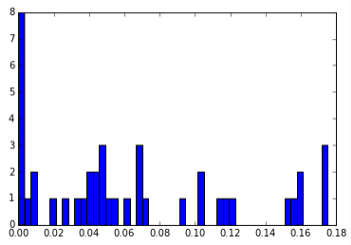
1. **Recode features:** Since string categorical variables can’t be handled for the learning algorithms, so we needed to recode them. Variables that need to be recoded are all in ‘business’. This was done by generating indicator variables for the diﬀerent values a categorical attribute could take. To simplify and balance the case, we did some grouping and then recode them. For example, for ‘smoking’, regardless of missing data, the previous value is ‘yes’, ‘no’ and ‘outdoor’, and ‘outdoor’ is relatively a small group. Therefore, we combine ‘no’ group with ‘outdoor’ group and recoded them as 0 while mapping ‘yes’ to 1.
2. **Merge three datasets:** Set ‘review’ dataset as the base, we linked each review with the related business information and user information. As a result, for each instance (review) in certain restaurant written by certain user, we can know the user and business information.
3. **Dealing with missing data:** After calculating the missing data percentage, we decided to drop those with more than half data missing. In our view, features with more than half data missing is less informative in itself. Though adding a new category to indicate missing is equivalent to adding a new dimension, we would still say that the conclusions drawn from these variables are meaningless. Meanwhile, by drawing the distribution (Figure3), the missing percentage is polarized and we decided to discard those variables with more than 50% missing.

Figure 3. Distribution of Missing Value Percentage for All Columns

### **Phase 5: feature selection based on mutual information and feature engineering.**

1. **Mutual Information Calculation and feature selection II:** For modelling efficiency, we applied another round of feature selection here. Except ‘text’ which is very informative and we decide to use it anyway, we calculated the mutual information between rest of variables and target variables. The distribution is below (figure 4). Finally we pick up 0.08 as a threshold and drop those features with mutual information lower than it. Finally we have variables and their mutual information as in Appendix.
2. **Feature engineering:** In this case, binning is applied to encode all categorical variables, then normalization is applied. Finally we have 30 features and one target variable.

Figure 4. Mutual Information Distribution

# **Modelling**

## **Modeling Introduction:**

We first defined our training, validation and testing data sets(56%, 24% and 20% respectively). everyone use training dataset (56%), adjust their feature sets and hyper-parameter, finally four best models’ performance will be compared on a validation dataset (24%), select the one with best performance, combine the training and validation and retrain using the best configuration, we figured and finally show the results using AUC on the test set (20%).

The same training and validation data sets were applied to baseline models as well as the improved models for model selection and comparison. The test data was then used to evaluate the selected best model.

1. **Configuration**

**Algorithms:** Decision Tree; Naive Bayes, Support Vector Machine, Logistic Regression

**Feature Sets:** baseline features (text only); text+29 features extracted from ‘user’ and ‘business’; predict ‘text’ score and use it as feature in the second model

**Hyperparameter:** the hyper parameters are tested to determine the feature set and hyper-parameter configuration.

**2) Evaluation Metric:** The area under the ROC curve (AUC) is the model evaluation metrics. It gives the probability that a review that should be upvoted will be identified compared to a review that should not be upvoted. It is a suitable metric for the binary target variable classification predictions, and hence for our project.

## **Baseline Models**

The baseline model uses only the review text for prediction. Three types of vectorization methods are used for text mining, i.e. binary, term frequency-inverse document frequency and count methods. For each vectorization method, we applied two models: logistic regression model and naïve bayes model. Default parameters are applied to models and vectorizations.

We trained the models using the training set and compared the area under the ROC curve (AUC score) of each model and vectorization combination when applying to the validation set. As the logistic regression from count vectorization method has the highest AUC score of 0.655, we aim to improve performance from there.

Table 3. Baseline Model AUC Results

|  |  |  |  |
| --- | --- | --- | --- |
| **AUC results** | **Binary Vectorization** | **TF-idf Vectorization** | **Counts Vectorization** |
| Logistic Regression | 0.644 | 0.651 | 0.655 |
| Naive Bayes | 0.640  (BernoulliNB) | 0.640  (BernoulliNB) | 0.629  (MultinomialNB) |

## **Possible Models and Experiments**

### **1. Decision Tree Model**

By splitting the predictor variable, decision tree gains more information for target variable prediction. Its abilities to handle categorical or text features and detect interactions among features makes decision tree suitable for our project.

The advantages of decision tree include its easiness for interpretation and implementation, cost effective as the total operations equal to the depth of tree and minimal feature engineering necessary. However, there are also some concerns. The greedy algorithm is not very stable and it is not very suitable as the sample size shrinks and the feature size grows. It also has an issue of overfitting, which has shown through the parameter selection process (see Appendix A).

Three types of vectorization methods with their default settings are used for text mining, i.e. binary, term frequency-inverse document frequency and count methods. The sparse matrices from vectorization transformation are then merged with the other 29 business specific and reviewer related features to form the feature matrices. The training set’s feature matrices and target label are used to fit decision tree classifiers with different parameter settings. The validation set are used to measure performance through AUC.

For each vectorization method, basic decision trees with min\_samples\_leaf=1, max\_depth=3 are created to compare performance of model with entropy with gini criterion. Results (Appendix A:Table 1) show no significant difference. Hence, criterion of entropy is used in the following parameter selections.

With the consideration of performance and running time, hyperparameters of max\_depth from 7 to 10 and min\_samples\_leaf of 100, 500, 1000 and 2000 are tested to determine the best performing parameters. The reuslts are shown in Table 4.

Table 4. The best performance parameter set and AUC value for each vectorization method:

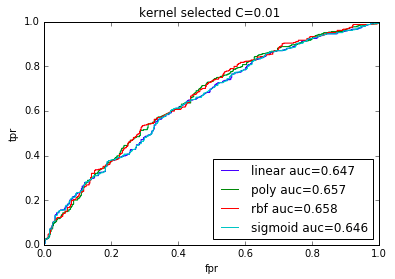
|  |  |  |  |
| --- | --- | --- | --- |
|  | **Binary Vectorization** | **Counts Vectorization** | **TF-idf Vectorization** |
| Best AUC | 0.675 | 0.676 | 0.673 |
| Min\_samples\_leaf | 500 | 100 | 500 |
| Max\_depth | 8 | 10 | 8 |

**Model Evaluation:** The best decision tree model parameter and text mining method combination is decision tree with minimal leaf number of 100 and depth of 10 for text transformed from count presentation. It improved the baseline model from 0.655 to 0.676 (3.2% improvement).

### **2. Support Vector Machine Model**

Support Vector Machine model separates examples to two different classification by constructing hyperplanes. This makes SVM model suitable for classifying whether a review should be upvoted. The advantages of SVM includes its effectiveness in high dimensional spaces, and in cases where the number of dimensions is greater than the sample size. It is also memory efficient, which is critical for the our large data set. The versatility of SVM is also attractive for kernel function customization. However it performs poorly when the number of features is much larger than the sample size and it doesn’t provide direct probability estimates.

Tf-idf text mining method was used to transform text features which are then attached with the other 31 features for SVM model. GriSearchCV method optimized by cross validated grid search over parameter grid is utilized for parameter selection in SVM. As ‘precomputed’ kernel doesn’t support sparse matrix, ‘linear’, ‘poly’, ‘rbf’ and ‘sigmoid’ kernels are tested. The penalty parameter of the error term, C, was tested from [ 1000, 100, 1, 0.1, 0.01, 0.001 ], and kernel coefficient, gamma, was tested from [ 2^(-5), 2^(-3), 2^(-1), 2^(1), 2^(3)]. The results are shown in figure and Appendix B

.

**Model Evaluation:** Based on the test score from Appendix, SVC with Kernel=’rbf’, gamma=0.5 and C=1 has the best performance. The AUC score from the best performing SVM model is 0.661, which 1% improvement from the baseline model.

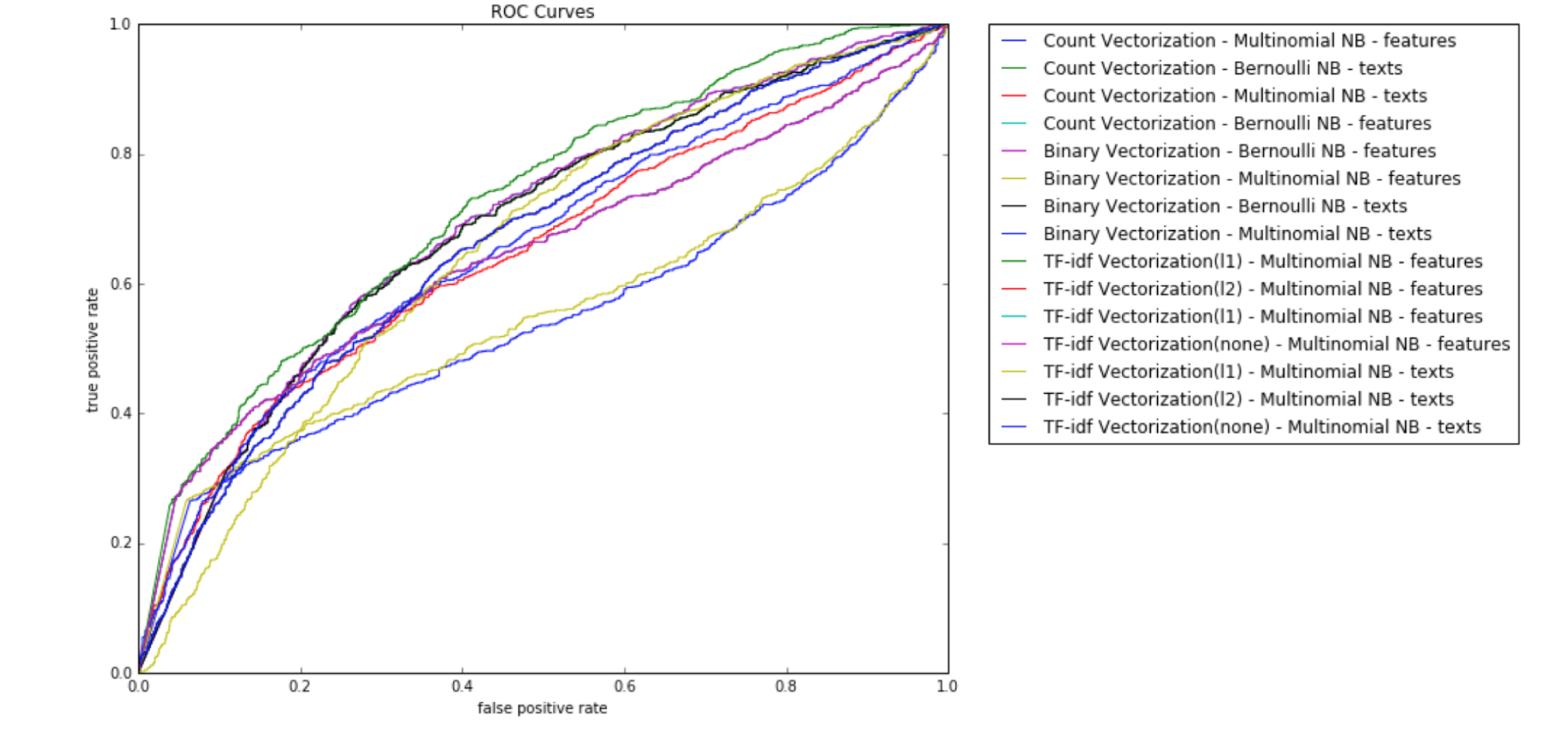
### **3. Naïve Bayes Model**

One other model we experimented was Naïve Bayes model. There are several types of Naïve Bayes models: Bernoulli Naïve Bayes, Multinomial Naïve Bayes, and Gaussian Naïve Bayes, which mainly differ from the assumptions regarding the distribution of . As it seems unlikely for the sparse text counts vectors to have a Gaussian distribution, we chose to test on Bernoulli and Multinomial Naïve Bayes models.

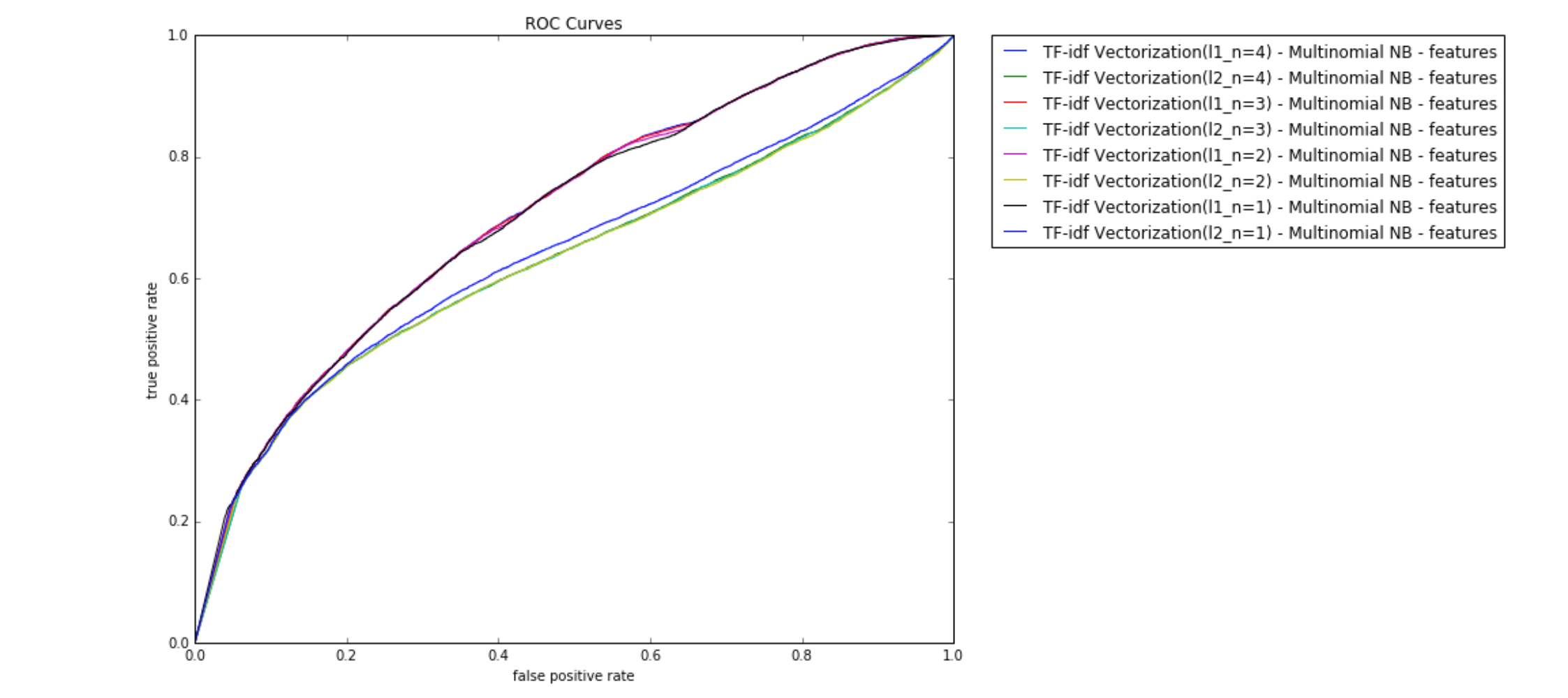
Also, as Bernoulli Naïve Bayes model will binarize any data input by default, we used only counts representation. As for Multinomial Naïve Bayes model, we did 5 kinds of representations: counts, binary counts, Tf-idf with norm, Tfi-df with norm, and Tf-idf with no norm(min\_df = 10 to get rid of rare exceptions). All six combinations are used to test and select feature set between text-only as feature set and text with selected restaurant features as feature set. From the results below, we can tell that Multinomial Naïve Bayes model performs the best with Tf-idf ( norm) vectorized text together with selected features as feature set.

Table 5. Naive Bayes AUC results with different feature set and vectorization methods

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Count** | **Binary** | **Tf-idf (None)** | **Tf-idf(l1)** | **Tf-idf(l2)** |
| Sparse-Bernoulli | 0.652 | 0.652 | 0.652 | 0.652 | 0.652 |
| Text-Bernoulli | 0.651 | 0.651 | 0.651 | 0.651 | 0.651 |
| Sparse-Multinomial | 0.660 | 0.666 | 0.669 | 0.682 | 0.689 |
| Text-Multinomial | 0.612 | 0.608 | 0.620 | Unable to get | 0.626 |

Figure 5. ROC Curves for Naive Bayes experiments

Moreover, we wanted to find out whether setting ngram would improve the prediction result. Thus we did another experiment with the best-performing Multinomial models with Tf-idf norm and norm, to test on the optimal number of . And the result shows us that ngram () does not affect the fitting result as much as we expected. Thus bigram or default setting would be enough for our dataset. The bright side of Naïve Bayes is that it works extremely fast, comparing to other models. Multinomial Naïve Bayes, especially, could provide a prediction accurate enough in a time efficient manner. To further improve Naïve Bayes model, we could use a Monte Carlo method to generate a locally optimal weight array for this bag of words.

Figure 6. ROC Curves comparing ngram(n = 1, 2, 3, 4)

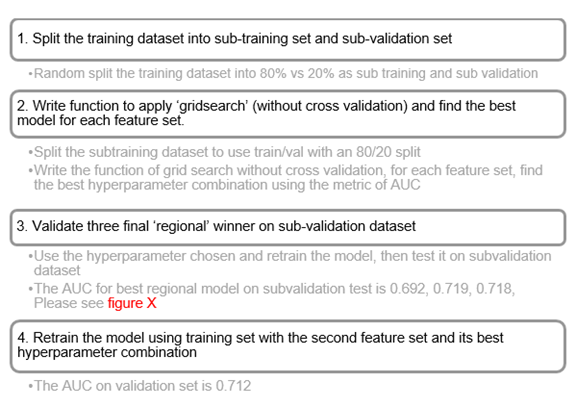
**Model Evaluation:** The bright side of Naïve Bayes is that it works extremely fast, comparing to other models. Multinomial Naïve Bayes, especially, could provide a prediction accurate enough for text mining in a time efficient manner. To further improve Naïve Bayes model, we could use a Monte Carlo method to generate a locally optimal weight array for the bag of words.

### **4. Logistic Regression Model**

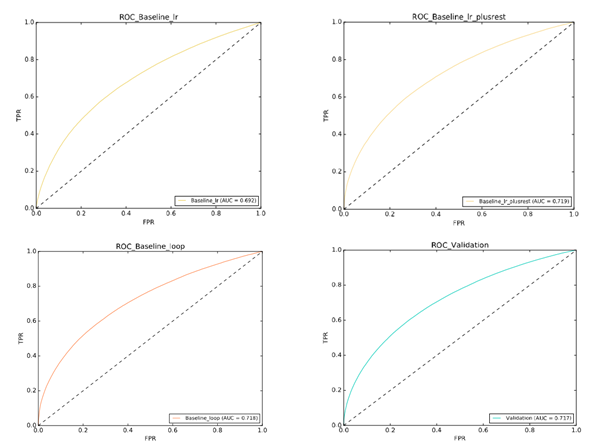
Given logistic regression, we have different feature sets and different hyperparameter .Also, because it deals with text, the hyper parameter also includes the way of training the converter. In my case, the algorithm is fixed, the feature sets will be three feature sets and my hyper-parameters are in below table.

Tabel 6. Basic Information about Tuned parameters\*[[3]](#footnote-2)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Hyper-  parameter | Model Num | Countvectorizer | | Tfidftransformer\* | SDGClassifier | |
| max\_features | ngram\_range | use\_idf | alpha | penalty |
| Feature Set 1 | 144 | [None, 3000, 5000] | (1,1), (1,2) | [True, False] | 10i in Range(-2,4) | [‘l1’,’l2’] |
| Feature Set 2 | 14 | None | (1,2) | False | 10i in Range(-3,4) | [‘l1’,’l2’] |
| Feature Set 3 | 14 | None | (1,2) | False | 10i in Range(-3,4) | [‘l1’,’l2’] |

Figure 7. Logistic Regression Model Training Procedure

Overall, there are 172 models trained in total. The best configuration overall is [feature set2, alpha=0.001, penalty=’l2’] [[4]](#footnote-3)

Figure 8. Logistic Regression ROC Curves 

**Model Evaluation:** Like what we saw, the logistic regression algorithm has its own advantage: compared to other single model (contrary to ensemble model), logistic regression has the best performance. It beats the baseline model and performance is quite robust. It sufficiently shows the advantages of this algorithm.

Feature set: As we expected, to decide if a review is useful, content itself is not enough. Certain attributes of the user and business where the review happens is also important.

Though the result is the best, the model suffers from underfitting. Since each time, whether train on the sub-training and test on sub-validation, train on the whole training and test on whole validation or use training dataset as test data, the AUC is always near 0.72. It is not bad since it is better than 0.5 guess, but it is clear that the model tuned doesn't sufficiently catch the information provided by more data.

## **Modelling Summary**

Dataset used in this projects contains 1,630,712 reviews and 31 variables: Attire, Good for Kids, Waiter, Service, Price Range, open, Wheelchair Accessible, Ambience\_score, Noise Level, Good For Groups, Take-out, Alcohol, Votes\_total\_user, Accepts Credit Cards and corresponding dummy variables. Dataset contained large amount of missing values. Considering that the number of missing values represent large percentage of the total dataset , thus, there are 12 dummy variables containing information on whether a value is missing or not. Additional manipulation of data includes converting Json file to csv, removing analytically meaningless variables, recoding features, merging three dataset, etc.

A simple naive bases baseline model relating reviews to voted label performs with 0.641 AUC. In order to improve the model and determine the optimal feature set and parameter set, four different classifiers: Logistic Regression, Support Vector Machine, Naive Bayes and Decision Tree are undertaken. After comparing four models, logistic regression classifier is thought to have the best performance. Table shows the comparing result. The analysis reveals that LR has the best performance.

Table 7. Optimal AUC Results of All Four Models

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | LR | SVM | BAYES | DT |
| AUC | **0.719** | 0.661 | 0.681 | 0.676 |

# **Deployment**

There are large amount of reasons a review might be recommended. For instance, the review seems more helpful or it may have been posted by a more established reviewer. Some of reviews are originating from computer or with specific bias. Model proposed in this report could solve this problem. The model suggests that whether a review would be upvoted by other users. That is, yelp could use the model to determine which reviews are more likely to be preferred or to be thought helpful by users, so that Yelp could recommend these reviews to enable these kind of reviews will be viewed by more chances. And this is reasonable since the reviews are thought more helpful for users. In this way, Yelp would be able to provide the most relevant and helpful information for Yelp users. Once users gain a convenient way to learn about nearly all the critical information they need, they are less likely to have churn behaviors. And in the long run, large amount of users will return more useful feedback and form a virtuous circle, which means a huge business channel with millions of potential source of revenue.

The model shows that test data set achieves a relatively ideal classification results. However, there are several limitations of the model. While the number of reviews reach to a specific level, the model cannot capture the useful information in text anymore. It may due to the model may not well fit the data set enough. There are other variable in this model have not been fully explored by this analysis, which we will explore in further study.

**Challenges and Limitations**

We have a really large data set of more than 1.6 million instances. It is very time-consuming and computational demanding for cleaning data, training and testing models. It restrains the amount of hyper-parameter testing can be performed for each model.

Our laptops’ RAM also limits what we can do. An attempt to solve the problem is to connect to the CIMS server to train and test models, however, python 2 and sklearn 0.12.1 have to be used instead. The code can be different from what we used to learn and practice for this class using our personal computer.

**Future Improvement:**

In the next step, Random Forest, Neural Network, Gradient Boosted Trees and other ensemble algorithms can be used to improve the accuracy and efficiency of the classification. For instance, we can use bagging technique to learn N models off of N bootstrap samples and get the mean of the N prediction results. Or we can use an iterative process where each model is fit on the residual of the cumulative prediction from all prior models.1

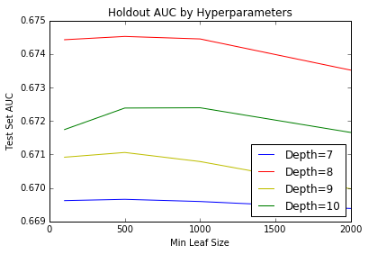
**Appendix A**

Table 1. Area under the ROC curve on the validation data for decision trees with min\_samples\_leaf=1, max\_depth=3

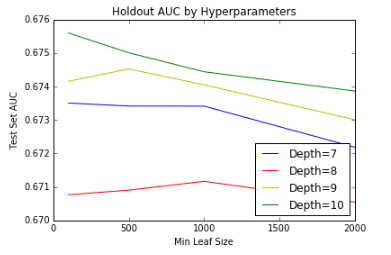
|  |  |  |  |
| --- | --- | --- | --- |
| Decision Tree Criterion | Binary Vectorization | Counts  Vectorization | TF-idf  Vectorization |
| Entropy | 0.659 | 0.658 | 0.659 |
| Gini (default) | 0.658 | 0.658 | 0.658 |

Decision Tree Model Parameter Selection Holdout AUC by Hyperparameter Curves:

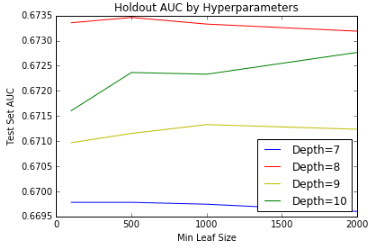
Binary Representation



Count Representation



TF-idf Representation



**Appendix B**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Table 1. SVM Model Parameter Selection Table | | | | | | | | | | |
|  | **mean\_fit\_time** | **mean\_score\_time** | **mean\_test\_score** | **mean\_train\_score** | **param\_C** | **param\_gamma** | **param\_kernel** | **params** | **rank\_test\_score** | **split0\_test\_score** |
| **56** | 0.19870694478352866 | 0.08866167068481445 | 0.6029411764705882 | 0.680773732440583 | 0.1 | 2 | sigmoid | {'gamma': 2, 'kernel': 'sigmoid', 'C': 0.1} | 1 | 0.5824175824175825 |
| **40** | 0.20390399297078451 | 0.09166097640991211 | 0.5980392156862745 | 1 | 1 | 2 | rbf | {'gamma': 2, 'kernel': 'rbf', 'C': 1} | 2 | 0.5787545787545788 |
| **10** | 0.21986770629882812 | 0.09072367350260417 | 0.5968137254901961 | 1 | 1000 | 2 | rbf | {'gamma': 2, 'kernel': 'rbf', 'C': 1000} | 3 | 0.575091575091575 |
| **25** | 0.2021336555480957 | 0.09065937995910645 | 0.5968137254901961 | 1 | 100 | 2 | rbf | {'gamma': 2, 'kernel': 'rbf', 'C': 100} | 3 | 0.575091575091575 |
| **34** | 0.198898712793986 | 0.09740368525187175 | 0.5968137254901961 | 0.7732838126549385 | 1 | 0.125 | rbf | {'gamma': 0.125, 'kernel': 'rbf', 'C': 1} | 3 | 0.5824175824175825 |
| **37** | 0.19816931088765463 | 0.08905529975891113 | 0.5906862745098039 | 0.9454475400665951 | 1 | 0.5 | rbf | {'gamma': 0.5, 'kernel': 'rbf', 'C': 1} | 6 | 0.5897435897435898 |
| **38** | 0.19224762916564941 | 0.08888037999471028 | 0.5906862745098039 | 0.8480332504620615 | 1 | 0.5 | sigmoid | {'gamma': 0.5, 'kernel': 'sigmoid', 'C': 1} | 6 | 0.5897435897435898 |
| **3** | 0.2079900105794271 | 0.09171096483866374 | 0.5833333333333334 | 1 | 1000 | 0.125 | poly | {'gamma': 0.125, 'kernel': 'poly', 'C': 1000} | 8 | 0.5860805860805861 |
| **6** | 0.2040729522705078 | 0.09068775177001953 | 0.5833333333333334 | 1 | 1000 | 0.5 | poly | {'gamma': 0.5, 'kernel': 'poly', 'C': 1000} | 8 | 0.5860805860805861 |
| **9** | 0.21142967542012533 | 0.0917206605275472 | 0.5833333333333334 | 1 | 1000 | 2 | poly | {'gamma': 2, 'kernel': 'poly', 'C': 1000} | 8 | 0.5860805860805861 |
| **12** | 0.2067867120107015 | 0.08975768089294434 | 0.5833333333333334 | 1 | 1000 | 8 | poly | {'gamma': 8, 'kernel': 'poly', 'C': 1000} | 8 | 0.5860805860805861 |
| **21** | 0.1986576716105143 | 0.08922799428304036 | 0.5833333333333334 | 1 | 100 | 0.5 | poly | {'gamma': 0.5, 'kernel': 'poly', 'C': 100} | 8 | 0.5860805860805861 |
| **24** | 0.20085763931274414 | 0.09098307291666667 | 0.5833333333333334 | 1 | 100 | 2 | poly | {'gamma': 2, 'kernel': 'poly', 'C': 100} | 8 | 0.5860805860805861 |
| **27** | 0.20300626754760742 | 0.08933472633361816 | 0.5833333333333334 | 1 | 100 | 8 | poly | {'gamma': 8, 'kernel': 'poly', 'C': 100} | 8 | 0.5860805860805861 |
| **39** | 0.20976789792378744 | 0.08985670407613118 | 0.5833333333333334 | 1 | 1 | 2 | poly | {'gamma': 2, 'kernel': 'poly', 'C': 1} | 8 | 0.5860805860805861 |

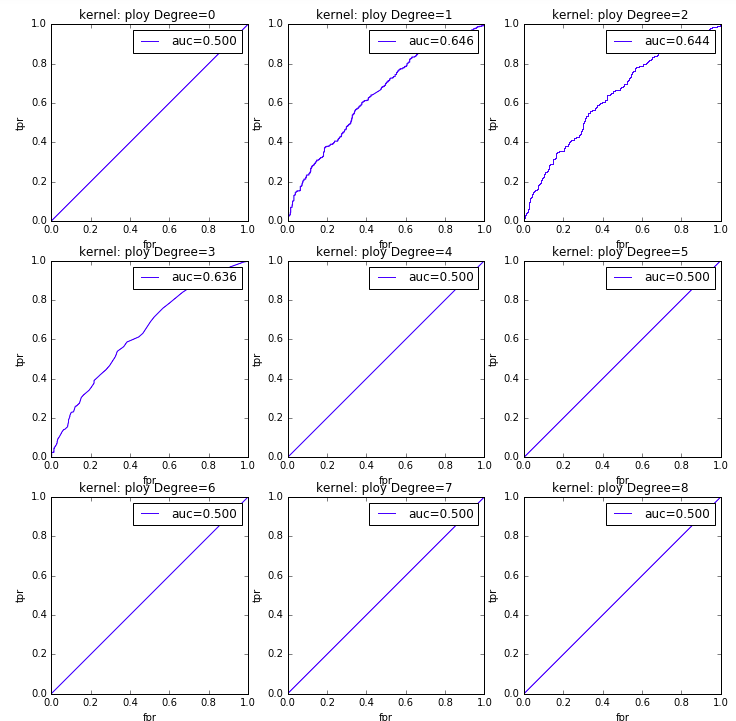
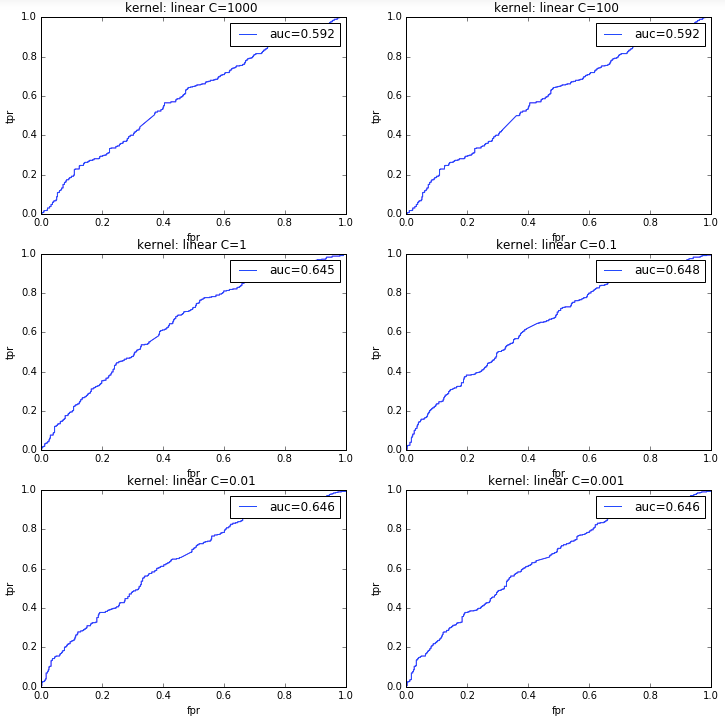
Figure 1.ROC Curves for Poly Kernel Degree Parameter Selection 

Figure2.ROC Curves for Linear Kernel C Parameter Selection 

**Appendix C**

**Team Member Contribution**

**Data:**

Data Extraction from JSON to CSV: Yidi, Yurui

Data Load/Transfer- Combine and Clean Data: Jingyi

Data Load/Transfer- Create New Feature: Lei

Model:

Text Mining and Baseline Model: Jingyi, Yurui

Feature Selection: Lei, Yidi

Decision Tree Model: Jingyi

Support Vector Machine: Yidi

Naïve Bayes Model: Yurui

Logistic Regression: Lei

**Report**: All

1. From Wikipedia: [*https://en.wikipedia.org/wiki/Yelp*](https://en.wikipedia.org/wiki/Yelp), accessed on Dec 9th 2016 [↑](#footnote-ref-0)
2. From Yelp:[*https://www.yelp.com/dataset\_challenge*](https://www.yelp.com/dataset_challenge), accessed on Dec 9th 2016 [↑](#footnote-ref-1)
3. (1) Tfidfvector is equal to using countvectorizor and tfidftransformer. (2) The max\_features is tuned for the consideration of combining after-conversion text in the next step, to make the importance of other features less trivial. (3) Since I am using SDGClassifier, the parameter for logistic regression is alpha and penalty.(4) To be computational efficient, the text conversion option in baseline is directly applied, as we believe it provides a fairly well information for logistic regression to catch. It might not be the best combination we can tune but it saves tones of time since actually the procedure of converting text is the most time-consuming.(5) Learning from the outcome of first model, we extend the range of alpha. [↑](#footnote-ref-2)
4. The AUC of second feature set is only 0.001 higher than that of third feature set. It is too far to say this configuration is better than the third one. However, since loop the model doesn’t significantly boost the simply plug-in model, we prefer the one easier to compute. [↑](#footnote-ref-3)