Team member
Vanisa Achakulvisut UIN667903568 vachak2@uic.edu
Abhijit Zarekar UIN676178928 azarek3@uic.edu

(a) Explore the data. How are star ratings distributed? How will you use the star ratings to obtain a label indicating 'positive' or 'negative' – explain using the data, graphs, etc.? Do star ratings have any relation to 'funny', 'cool', 'useful'? Is this what you expected?

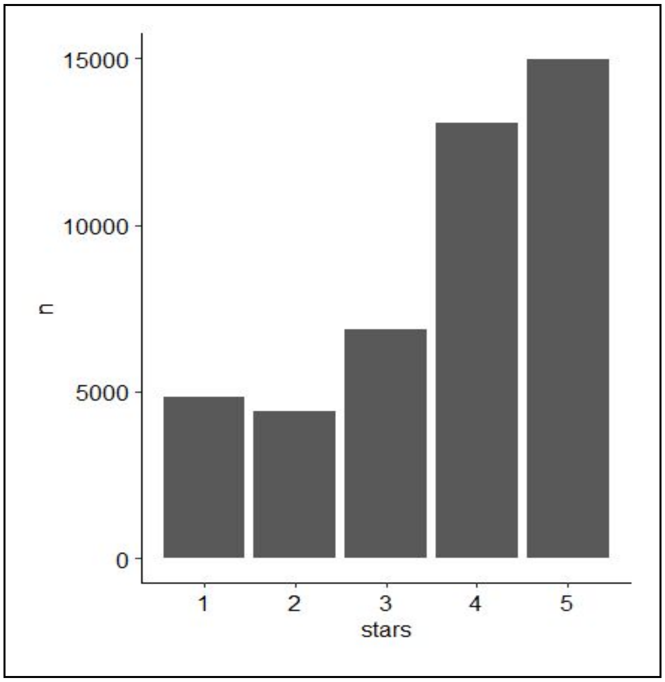
Data exploration is the first step of starting the project. In order to understand the data, we had taken a look at every attribute that was presented on the data. The list of features are as follows;

No.	Feature name	Description	Remarks
1	business_id	The business identification of each restaurant	
2	cool	The number of customer rating "cool" each restaurant	*Positive word
3	date	Date	
4	funny	The number of customer rating "funny" each restaurant	*Positive word
5	review_id	Review ID	
6	stars	The number of customer star rating each restaurant	**True label of Predicted Y for modeling part
7	text	Reviews from customers	*** Token all these into a single word and find its sentiment.
8	type	Type of text which is review	
9	useful	The number of customer rating "useful" each restaurant	*Positive word
10	user_id	User identification	
11	address	Address of restaurant	
12	attributes.0	Describe the atmosphere and facilities they provided. Restaurant attributes i.e. alcohol, wifi, parking, noise level	

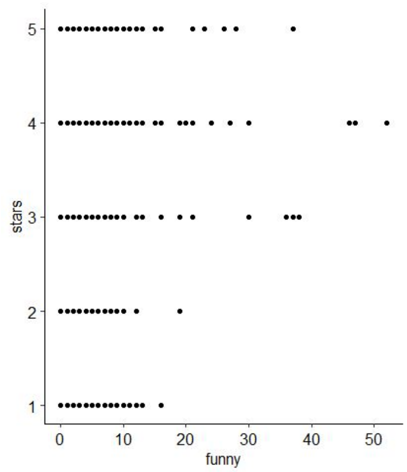
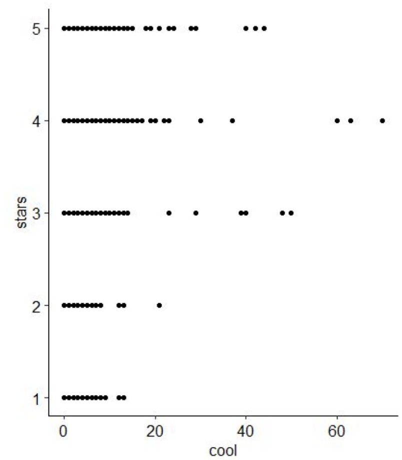
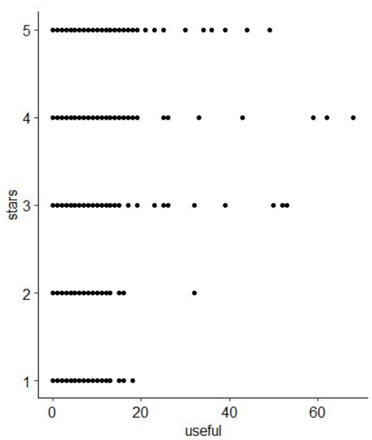
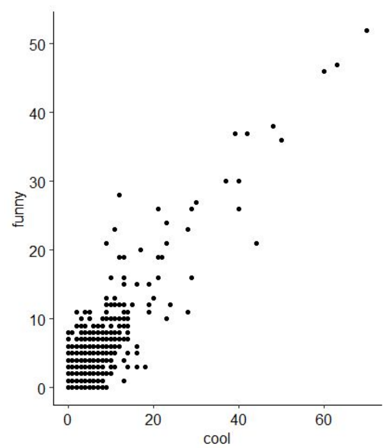
13	attributes.1	Describe the atmosphere and facilities they provided. Restaurant attributes i.e. alcohol, wifi, parking, noise level	
14	attributes.2	Describe the atmosphere and facilities they provided. Restaurant attributes i.e. alcohol, wifi, parking, noise level	
15	categories.0	Restaurant categories i.e. American, Seafood, Barbeque, Pubs	
16	categories.1	Restaurant categories	
17	categories.2	Restaurant categories	
18	city	Location of restaurant	
19	is_open	Status	
20	latitude	Location of restaurant	
21	longitude	Location of restaurant	
22	name	Name of restaurant	
23	neighborhood	Neighborhood of restaurant	
24	postal_code	Postal code of restaurant	
25	review_count	Number of review for this restaurant	
26	state	Location of restaurant	

Our studying goal is to understand the importance of words presented in the reviews, infer their sentiment and attempt to find the sentiment prediction model. The target variable for this this project is “sentiment” which is binary classification with positive sentiment (1) and negative sentiment (0)

Data Visualization to get basic understanding of dataset:



The above graph indicates number of star ratings for each of 1 to 5 star ratings:
Star rating 5 has the highest number of votes, followed by 4 star rating. Star ratings 1 and 2 have significantly less number of votes than star ratings 4 and 5.

<table>
<tr><td colspan="2">The scatter plot of “funny”, “cool”, “useful” and “cool vs funny”</td></tr>
<tr><td>
</td><td>
</td></tr>
<tr><td>In this graph as the number of star ratings increase few users found the restaurant funny. For example for 4 star ratings 52 users found the reviews to be funny.</td><td>The graph shows the number of star rates of “cool”. There are high rates of “Cool” in 5 - 4 stars, few users found the restaurant cool on lower stars.</td></tr>
<tr><td>
</td><td>
</td></tr>
<tr><td>The graph shows the number of star rates of “Useful”. There are high rates of “Useful” in 5 - 4 stars, few users found the restaurant “useful” on lower stars.</td><td>This graph shows the relationship between the occurrence of “Funny” vs “Cool” in each restaurant. As it can be seen on the graph, there is high density on low numbers of cool and funny. This points out that there are several restaurants that get few votes with these two positive words. On the other hand, on the top right of the graph, it points out that only a few restaurants get really good votes on “funny” and “cool” at the same time.</td></tr>
</table>

(b) What are some words indicative of positive and negative sentiment? (One approach is to determine the average star rating for a word based on star ratings of documents where the word occurs). Do these 'positive' and 'negative' words make sense in the context of user reviews? (For this, since we wish to get a general sense of positive/negative terms, you may like to consider a pruned set of terms -- say, those which occur in a certain minimum and maximum number of documents).

Data Cleansing

It is essential to install the library; library(tidytext), library(SnowballC), library(textstem) before performing the text processing.

Only review_id, stars, text were selected and stored in rrTokens. Review_id identifies the unique review_id, stars in dictates the positive-negative (mentioned later in this document), text (contained word for sentiment prediction)

First we tokenize the review from the text column. 89,415 words are found from this text review. The next step is to remove the stop words which are not meaningful for sentiment interpretation. Therefore the amount of distinct words are reduced to 88,710 words

After applying the tokenize method on review features, many words were generated. This will cost very expensive computational costs. Setting the criteria, scoping down the amount of word for sentiment prediction will be useful for modeling

Treating the data steps;

1. Tokenize
2. Transform case (to all lower/upper)
3. Filter stopwords - filter tokens by length - say, min 3 and max 15.
4. Stemming, Lemmatization - others as you find useful.
5. Remove the words which are not present in at least 10 reviews
6. Remove the data where it contains the numeric value i.e. 6oz, 1.15 etc.

Then we visualize the frequency of each word for every stars ranking.

```
> head(rrTokens_stem)
               review_id stars    word word_stem
1 90P3RlRPhSXrIb-yiJbSjA     3 chinese    chines
2 90P3RlRPhSXrIb-yiJbSjA     3    fast      fast
3 90P3RlRPhSXrIb-yiJbSjA     3    food      food
4 90P3RlRPhSXrIb-yiJbSjA     3   serve      serv
5 90P3RlRPhSXrIb-yiJbSjA     3    cans       can
6 90P3RlRPhSXrIb-yiJbSjA     3 bottles     bottl
> head(rrTokens_lemm)
               review_id stars    word word_lemma
1 90P3RlRPhSXrIb-yiJbSjA     3 chinese    chinese
2 90P3RlRPhSXrIb-yiJbSjA     3    fast       fast
3 90P3RlRPhSXrIb-yiJbSjA     3    food       food
4 90P3RlRPhSXrIb-yiJbSjA     3   serve      serve
5 90P3RlRPhSXrIb-yiJbSjA     3    cans        can
6 90P3RlRPhSXrIb-yiJbSjA     3 bottles     bottle
```

Picture on the left shows the word after Stemming and Lemmatization.

Stemming and Lemmatization both generate the foundation of the words. The only difference is that stemmer operates on a single word without knowledge of the context, and therefore cannot discriminate between words which have different meanings depending on part of speech.

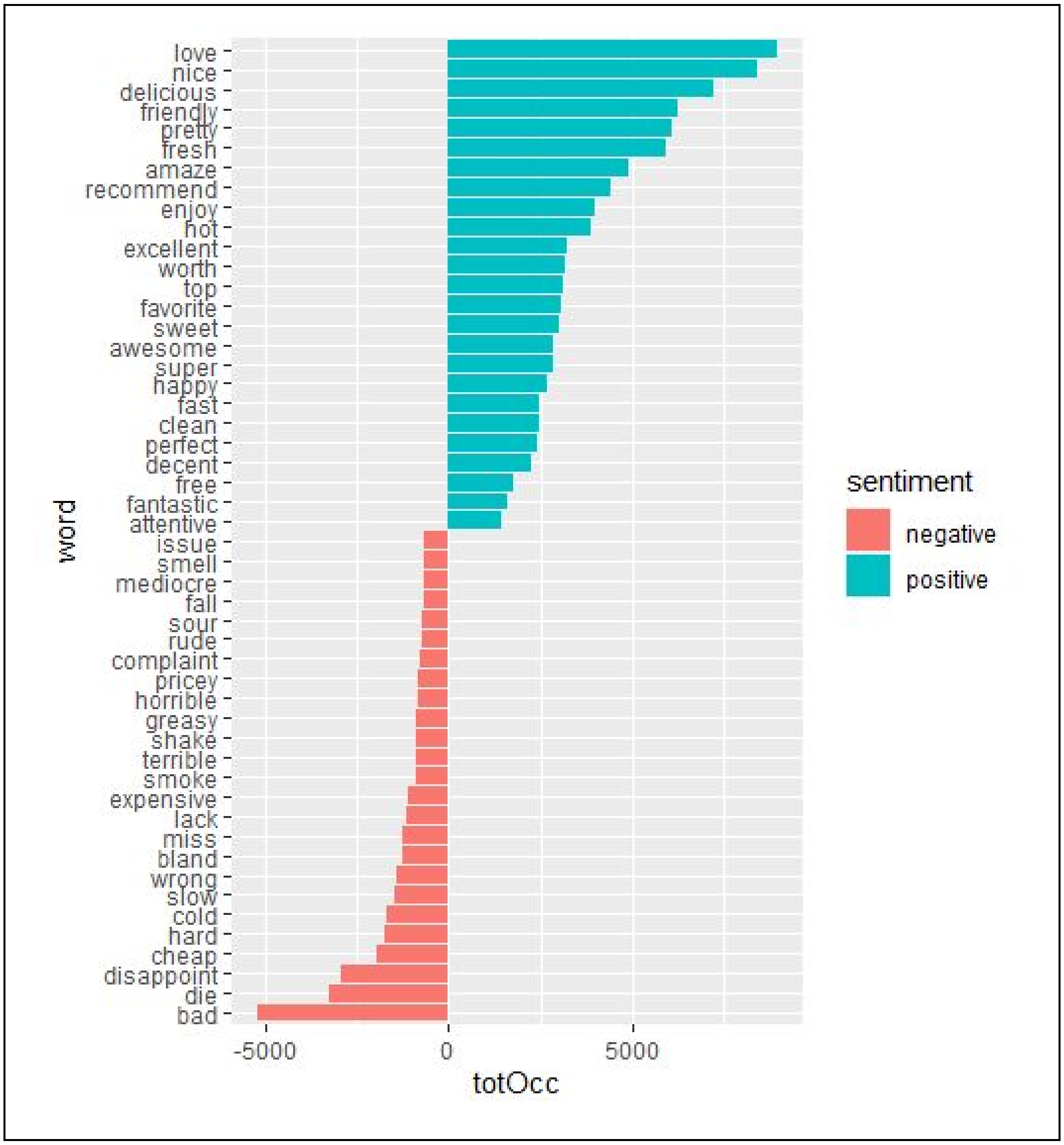
In this assignment we will use lemmatization technique because the meaning of words is necessary for mapping the sentiment in each dictionary (BING, NRC, AFINN).

(c) We will consider three dictionaries, available through the tidytext package – the NRC dictionary of terms denoting different sentiments, the extended sentiment lexicon developed by Prof Bing Liu, and the AFINN dictionary which includes words commonly used in user-generated content in the web. The first provides lists of words denoting different sentiment (for eg., positive, negative, joy, fear, anticipation, …), the second specifies lists of positive and negative words, while the third gives a list of words with each word being associated with a positivity score from -5 to +5.
How many matching terms are there for each of the dictionaries?

3 different dictionaries (BING, NRC, AFINN) were used for mapping the sentiments/value of tokened words. Each dictionary gives different sentiments/ values. The sentiment and value from those dictionaries are follows;

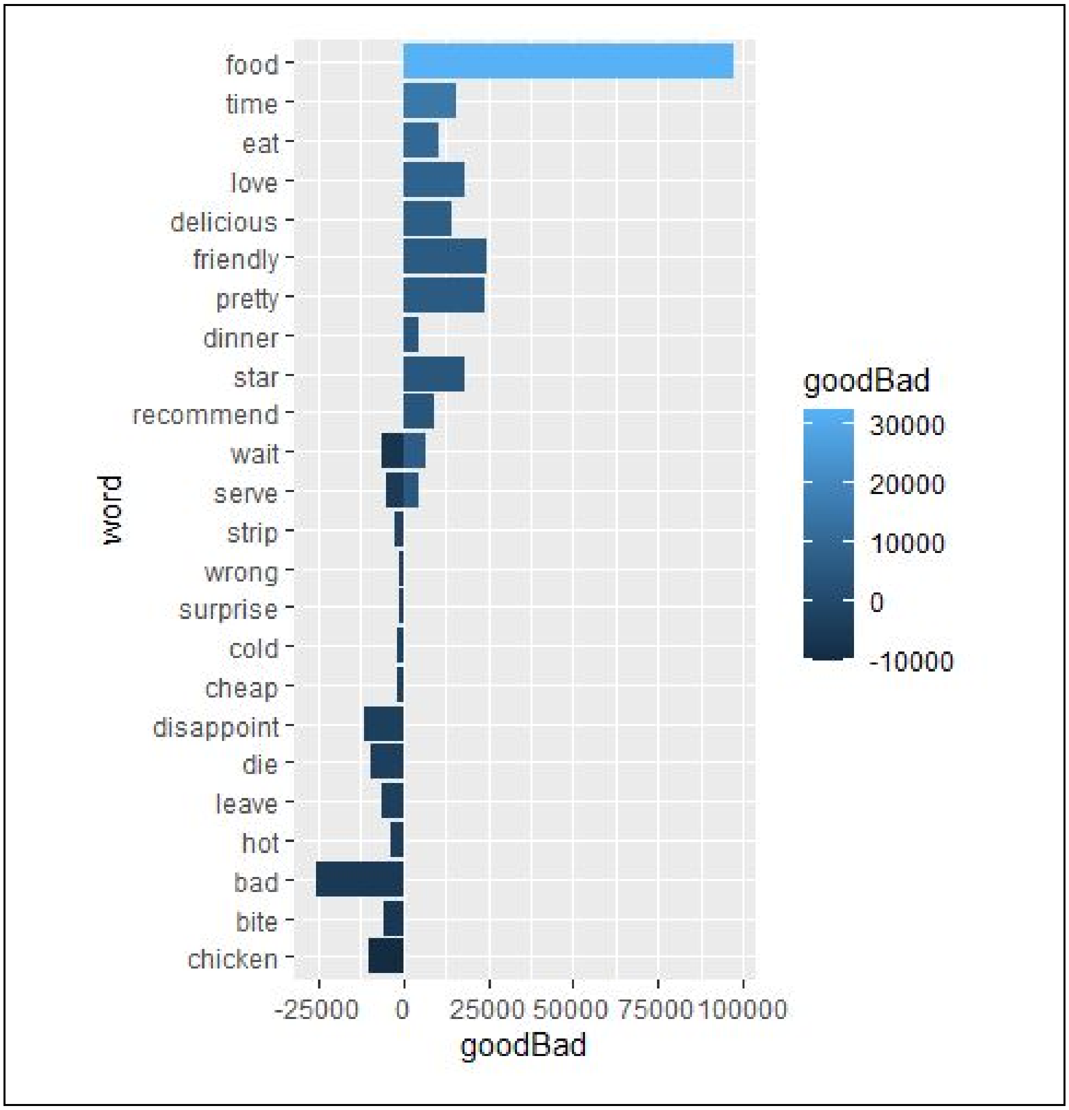
DICTIONARY	Sentiment (Bing, NRC) / Value (AFINN)
Bing	Positive, Negative
NRC	Trust, fear, negative, sadness, anger, surprise, positive, disgust, joy, anticipation
AFINN	-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5

Bing Dictionary:



Graph shows the total number of word occurrences in each positive and negative word which was mapped sentiment with the Bing dictionary. I can be clearly seen that “love”, “nice”, “delicious”, “friendly”, “pretty” are the most popular positive words in review. Similarly to “bad”, “die”, “disappoint”, “cheap”, “hard” are the most popular negative words in review describing how bad it is. The sentiment mapping from Bing dictionary is reasonable compared to the actual meaning of the words.

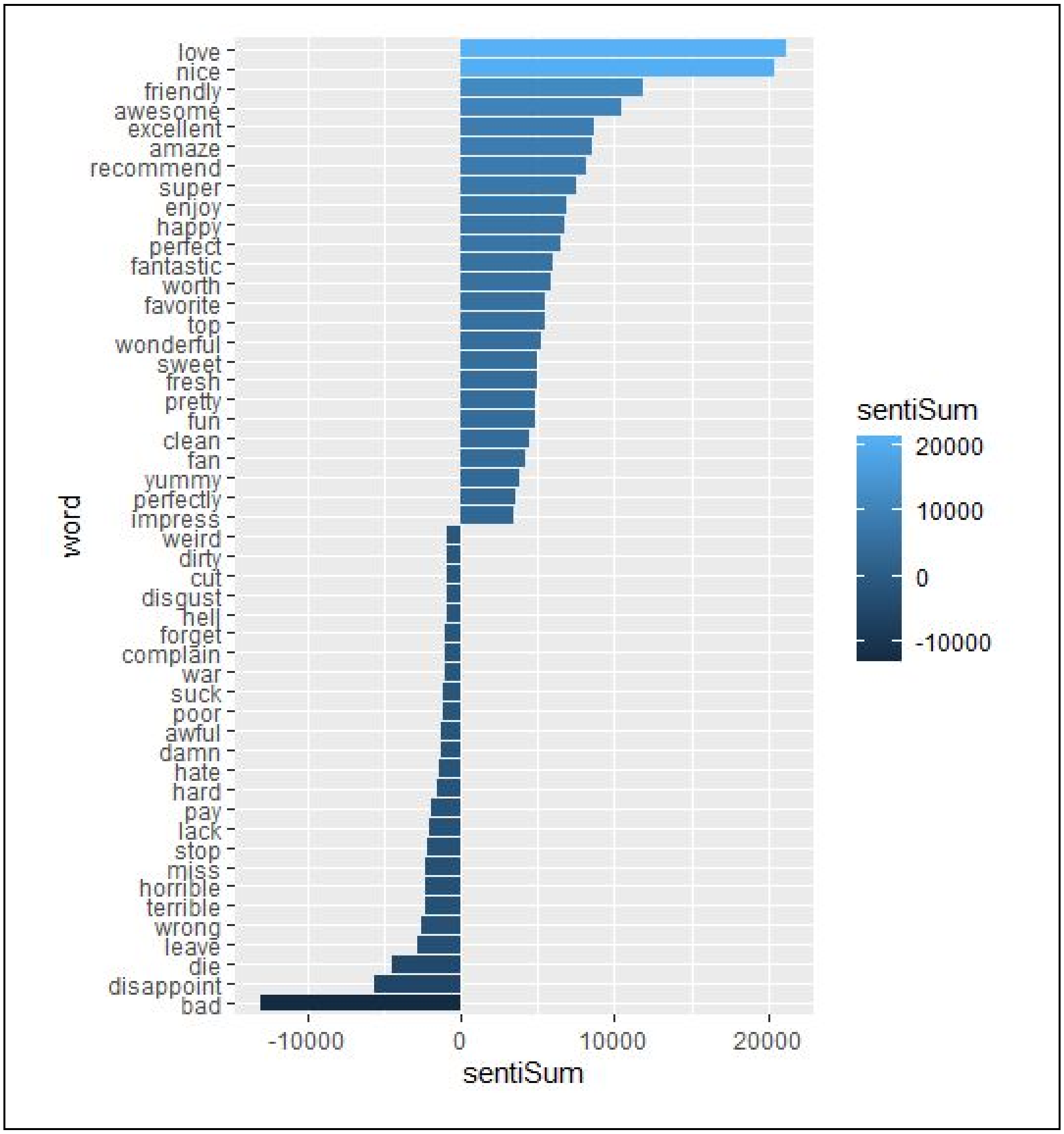
NRC Dictionary:



NRC dictionary contains words defining different sentiments (describing in words, not positive or negative score). These word's sentiments are converted to goodBad. {anger, disgust, fear sadness, negative} denote 'bad' reviews, and {positive, joy, anticipation, trust} to denote 'good' reviews. Then these words are given the positive and negative sign to total the number of words occurring.

As it can be seen from the graph, the most occurring positive is "food" which does not indicate the relationship between positive and negative sentiment in the actual world. However, "friendly", "pretty", "love" are the next popular words which their sentiments match with Bing dictionary as well. Similarly to the word; "bad", "disappoint", "die" are the words that indicate the negative feeling where match with Bing

AFINN Dictionary:



AFINN dictionary is different from Bing and NRC in terms of the sentiment value. Bing and NRC give the positive and negative feeling; however, AFINN gives the value in the int range of (-5, 5). For negative words AFINN gives negative values -5 to -1. For positive words AFINN gives positive values 1 to 5. The word Bad has got the highest negative sum which is less than -10000. So the “word” bad appears in the reviews the highest number of times. Similarly, for the word “love”(positive sentiment) the sentiSum is over 20,000 which indicates that this word was used in the reviews many times.

Top 3 Occurrence Words		
DICTIONARY	**Positive**	**Negative**
Bing	1. Love 2. Nice 3. Delicious	1. Bad 2. Die **3.** Disappoint
NRC	1. Friendly 2. Pretty **3.** Love	1. Bad 2. Disappoint **3.** Die
AFINN	1. Love 2. Nice **3.** Friendly	1. Bad 2. Disappoint 3. Die

Consider using the dictionary based positive and negative terms to predict sentiment (positive or negative based on star rating) of a movie. One approach for this is: using each dictionary, obtain an aggregated positiveScore and a negativeScore for each review; for the AFINN dictionary, an aggregate positivity score can be obtained for each review. Are you able to predict review sentiment based on these aggregated scores, and how do they perform? Does any dictionary perform better?

The star ratings will be used here to indicate the sentiment label.
For binary classification, we will need to convert the 1-5 scale rating values to {positive(1), negative(0)} values.

It would make sense to associate 4-5 star reviews with a positive sentiment , 1-2 star reviews with a negative sentiment and 3-star reviews would be neutral. 3-star reviews are likely to contain both positive and negative sentiment which is ambiguous; therefore, 3-star reviews are discarded in predicting models.

The actual number of positive reviews (1) and negative reviews (-1) are derived from star reviews. Below information shows the comparison of actual sentiment derived from star and sentiment form 3 different dictionaries.

Bing: Accuracy = **(**27052+108926)/(27052+108926+18290+32636) = **0.7275286**

PREDICTED/ACTUAL	-1	1
-1	27052	18290
1	32636	108926

NRC: Accuracy = **(**78396+376366) / (78396+376366+103987+110966) = **0.6790381**

PREDICTED/ACTUAL	**-1**	**1**
-1	78396	103987
1	11096 6	376366

AFINN: Accuracy = (5474 + 23205) /(5474 + 23205+3128+3119) = **0.8211361**

PREDICTED/ACTUAL	**-1**	**1**
-1	5474	3128
1	3119	23205

(Actual, predicted): (-1,-1), (1,1) denote the matching number of sentiment from Star and dictionary. (-1,1), (1,-1) denote the unmatching classes. The highest accuracy presents in the table comparison between actual and dictionary Afinn. Thus, dictionary Afinn gives the highest accuracy among 3 dictionaries which had been considered.

Combined (part D):
Accuracy = (87894 +429836) /(87894 +429836 +11044+123601) = **0.793608**

PREDICTED/ACTUAL	**-1**	**1**
-1	87894	111044
1	123601	429836

The combined dictionary gives the accuracy in the middle of all 3 dictionaries which reflect the combination of accuracy of three dictionaries.

(d) Develop models to predict review sentiment. For this, split the data randomly into training and test sets. To make run times manageable, you may take a smaller sample of reviews (minimum should be 10,000). One may seek a model built using only the terms matching any or all of the sentiment dictionaries, or by using a broader list of terms (the idea here being, maybe words other than only the dictionary terms can be useful). You should develop at least three different types of models (Naïve Bayes, and at least two others of your choice ….Lasso logistic regression (why Lasso?), xgb, svm, random forest (ranger).

In this part we will develop and evaluate classification models to predict sentiment polarity (negative, positive).

In this assignment, we are going to build the model

1. Naive Bayes
2. Glm logistic regression with Lasso
3. Random Forest

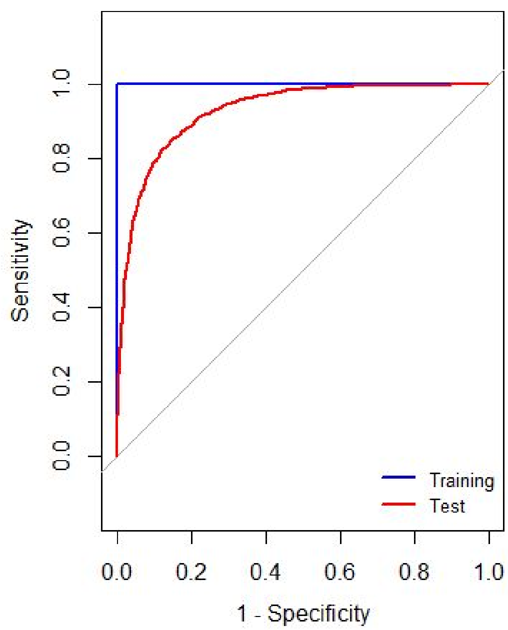
We developed models to predict review sentiments. We used 10,000 rows.
Here we developed 3 models (Naïve Bayes, Random Forest and Lasso Logistic regression) using broader list of terms using Bing dictionary.

Naive Bayes
Training: Area under the curve: **0.4653**
Testing: Area under the curve: **0.5006**

Random Forest model:
> rfModel1
Ranger result

Call:
ranger(dependent.variable.name = "hiLo", data = revDTM_sentiBing_trn %>% select(-review_id), num.trees = 500, importance = "permutation",   probability = TRUE)

Type:	Probability estimation
Number of trees:	500
Sample size:	5000
Number of independent variables:	8394
Mtry:	91
Target node size:	10
Variable importance mode:	permutation
Splitrule:	gini
OOB prediction error (Briers):	0.09874822



Training Dataset Confusion Matrix:

PREDICTED/ACTUAL	1	-1
1	1241	2
-1	1	3756

Training dataset accuracy:
(1241+3756)/(1241+3756+2+1) = **0.99**

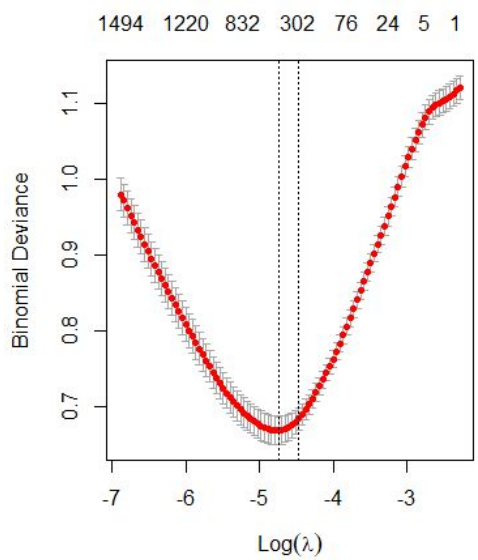
Test Dataset Confusion Matrix:

PREDICTED/ACTUAL	1	-1
1	625	557
-1	57	3761

Test dataset accuracy:
(3761+625)/(3761+625+557+57) = **0.8772**

> bThr<-coords(rocTrn, "best", ret="threshold", transpose = FALSE)
> bThr
Best threshold: 0.5483545

Logistic Regression with Lasso:



> confusionMatrix (glm_M1_test_pred,yDTst2, positive="1")

<u>Confusion Matrix and Statistics</u>

Confusion matrix for Train Dataset:

Prediction\Reference	1	-1
1	3710	537
-1	0	0

Accuracy : 0.8736

Confusion matrix for Test Dataset:

Prediction\Reference	1	-1
1	3751	625
-1	67	557

Accuracy : 0.8616

Why Lasso regression?

The “LASSO” stands for Least Absolute Shrinkage and Selection Operator. Lasso regression is a regularization technique. Lasso is used over regression methods for a more accurate prediction. In order to avoid overfit of the model we use regularization techniques like Lasso. The lasso technique is for simple, sparse models (i.e. models with fewer parameters). After implementing pivot_wider() and ungroup() functions we get a dataframe we have many NA’s (i.e. very few parameters). Hence, the distribution of the values is sparse in our Document Term Matrix (refer the following image of the dataframe).

	review_id	stars	buffet	copper	food	kettle	price	service	soo	star	suck	tempe	top	appetizer	average
1	--9qM_dRW4rrKTWO_SX_qQ	1	0.3710398	0.646604	0.06529675	0.6913746	0.16786417	0.10728892	0.5593517	0.22605298	0.4314110	0.5319978	0.253034	NA	NA
2	--9vqIJ0xGKY2L1Uz-L9Eg	3	NA	NA	0.01381277	NA	NA	NA	NA	NA	NA	NA	NA	0.06131679	0.0688886
3	--AbaUbQdJjw2sfObU_0tA	5	NA	NA	0.05985535	NA	0.15387549	NA	NA	NA	NA	NA	NA	NA	NA
4	--bRqe3_4P1G3eWOKeBYpQ	3	NA	NA	0.02476773	NA	NA	NA	NA	0.08574423	NA	NA	NA	NA	NA
5	--d7FO3sgkqvNvkuoQsM2w	4	0.1774538	NA	0.01040963	NA	0.02676095	NA	NA	NA	NA	NA	NA	NA	NA
6	--EOY1nAhWEhCyhaccXe2g	4	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
7	--F1tbZdtqrqx_Uh8rFFBw	2	NA	NA	0.08450167	NA	NA	NA	NA	NA	0.1395741	NA	NA	NA	NA
8	--F3IDF1PBm7KPEE-br0Tg	2	NA	NA	NA	NA	NA	0.07376113	NA	NA	NA	NA	NA	0.19927957	NA
9	--f8_qtGLfQep7hfRYzABg	2	NA	NA	0.01773492	NA	NA	NA	NA	0.03069855	NA	NA	NA	NA	NA
10	--gaw9doU7mIXfQUnQQ8vQ	2	NA	NA	0.01751864	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(i) Develop models using only the sentiment dictionary terms – try the three different dictionaries; how do the dictionaries compare in terms of predictive performance for rating ? Then with a combination of the three dictionaries, ie. combine all dictionary terms. Do you use term frequency, tfidf, or other measures, and why? What is the size of the documentterm matrix? Should you use stemming or lemmatization when using the dictionaries?

Here we develop 3 models for each of the 3 dictionaries using only the sentiment dictionary terms. For using only the sentiment dictionary terms we use inner_join() function.

Dimension of each dataset or the DTM:

Dictionary	No. of rows	No. of columns
Bing	42290	1099
NRC	43505	1535
AFINN	41280	593
Combined	43772	2041

We use tf(term frequency) idf(inverse document frequency) because:

1. Term frequency gives the frequency of the terms that occur multiple times in the document.
2. Inverse Document Frequency gives the frequency of the terms that occur across many documents are not useful for differentiating between documents. If value of IDF is high that means the term is occurring fewer times.
3. Therefore, tf_idf value gives a weight to the terms in the document indicating occurrences of the words in the document.

Lemmatization is used because Lemmatization keeps the meaning of the word which can be mapped to the interested dictionary (BING, NRC, AFINN).

For the modeling part, 10,000 rows of data were sampled for the calculation.
There are 12 models for the comparison in this assignment.

1) Bing: Naive Bayes
2) Bing: Logistic Regression with Lasso
3) Bing: Random Forest
4) NRC: Naive Bayes
5) NRC: Logistic Regression with Lasso
6) NRC: Random ForestBing: Naive Bayes
7) AFINN: Naive Bayes
8) AFINN: Logistic Regression with Lasso
9) AFINN: Random Forest
10) Combination: Naive Bayes
11) Combination: Logistic Regression with Lasso
12) Combination: Random Forest

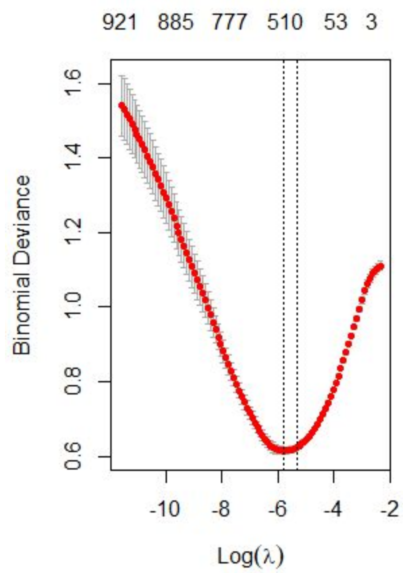
First, **Bing** dictionary is used in the model as follows:

1) **Bing: Naive Bayes:**

Training dataset: Area under the curve: 0.5895

Test dataset: Area under the curve: 0.6981

2) **Bing: Logistic Regression with Lasso:**



Confusion matrix for Train Dataset (bing dictionary):

Prediction\Reference	**1**	**-1**
1	3728	398
-1	50	824

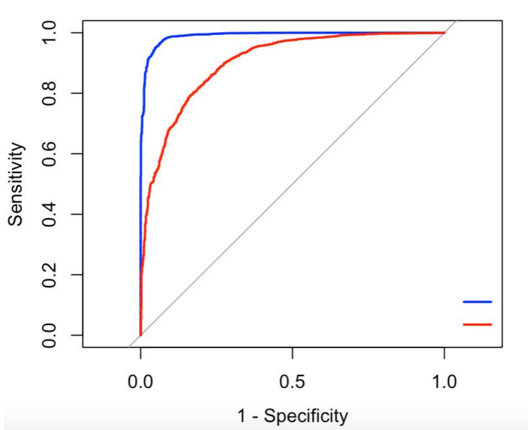
Accuracy : 0.9104

Confusion matrix for Test Dataset (bing dictionary):

Prediction\Reference	**1**	**-1**
1	3636	565
-1	139	660

Accuracy : 0.8592

3) **Bing: Random Forest:**



Confusion matrix for Train Dataset (bing dictionary):

Prediction\Reference	**1**	**-1**
1	**1104**	**132**
-1	**45**	**3719**

Accuracy : 0.9646

Confusion matrix for Test Dataset (bing dictionary):

Prediction\Reference	**1**	**-1**
1	**736**	**473**
-1	**165**	**3626**

Accuracy : 0.8724

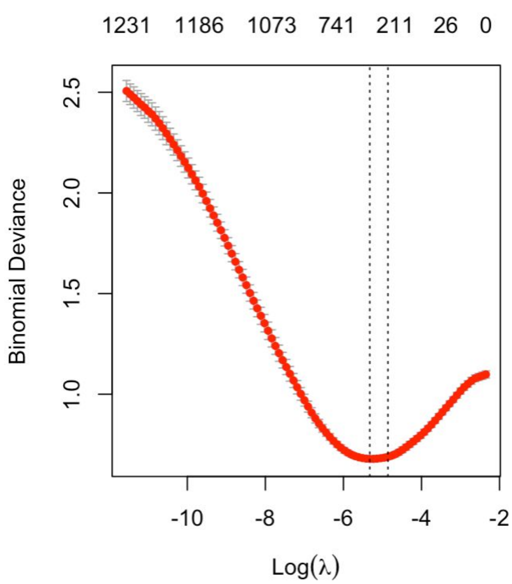
Second, **NRC** dictionary:

4) NRC: Naive Bayes

Training dataset: Area under the curve: 0.5176

Test dataset: Area under the curve: 0.5009

5) NRC: Logistic Regression with Lasso



Confusion matrix for Train Dataset (NRC dictionary):

Prediction\Reference	1	-1
1	3764	559
-1	44	633

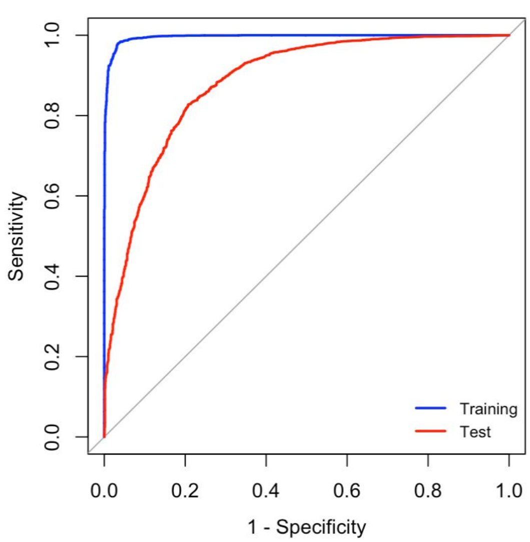
Accuracy : 0.8794

Confusion matrix for Test Dataset (NRC dictionary):

Prediction\Reference	1	-1
1	3617	723
-1	109	551

Accuracy : 0.8336

6) NRC: Random ForestBing: Naive Bayes



Confusion matrix for Train Dataset (NRC dictionary):

Prediction\Reference	1	-1
1	**1054**	**138**
-1	**15**	**3793**

Accuracy : 0.9694

Confusion matrix for Test Dataset (NRC dictionary):

Prediction\Reference	1	-1
1	**652**	**622**
-1	**111**	**3615**

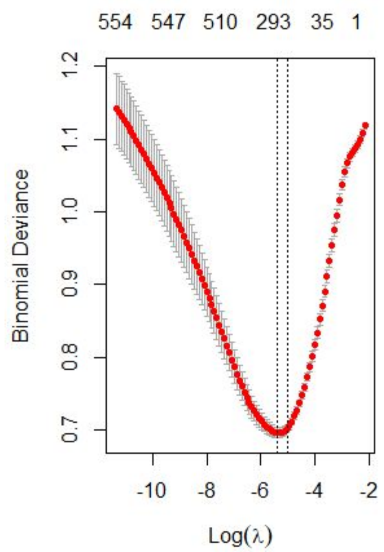
Accuracy : 0.8532

Third, **AFINN** dictionary:

7) AFINN: Naive Bayes

Training dataset: Area under the curve: 0.6463
Test dataset: Area under the curve: 0.71

8) AFINN: Logistic Regression with Lasso



Confusion matrix for Train dataset (AFFIN dictionary):

Prediction\Reference	1	-1
1	3686	562
-1	75	677

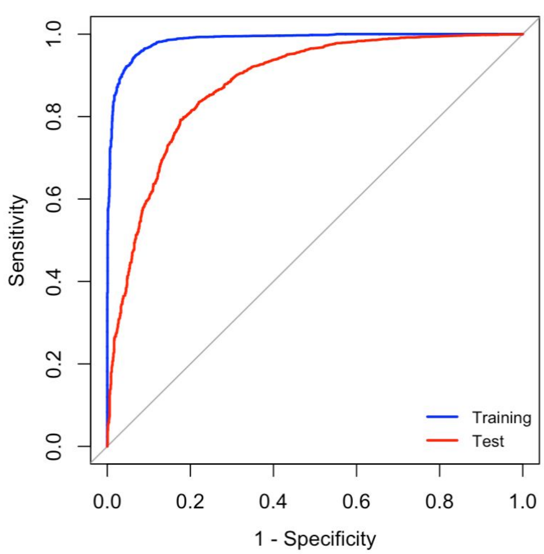
Accuracy : 0.8726

Confusion matrix for Test dataset (AFFIN dictionary):

Prediction\Reference	1	-1
1	3658	623
-1	126	593

Accuracy : 0.8502

9) AFINN: Random Forest



Confusion matrix for Train Dataset (AFINN dictionary):

Prediction\Reference	1	-1
1	**1046**	**181**
-1	**53**	**3720**

Accuracy : 0.9532

Confusion matrix for Test Dataset (AFINN dictionary):

Prediction\Reference	1	-1
1	**676**	**540**
-1	**181**	**3603**

Accuracy : 0.8558

Finally, dictionary we used is the **combination result** of dictionaries

No.	BING	NRC	AFINN	Combined (Sum of votes)	Flag (1,-1)
1	1	1	1	3	1
2	1	1	-1	1	1
3	1	-1	1	1	1
4	1	-1	-1	-1	-1
5	-1	1	1	1	1
6	-1	1	-1	-1	-1
7	-1	-1	1	-1	-1
8	-1	-1	-1	-3	-1
9	0	1	-1	0	0
10	0	0	0	0	0

To combine the dictionaries, the count of votes has been used as the criteria. Three dictionaries, Bing, NRC, AFINN, are considered. First, the positive and negative sentiment are flagged as 1 and -1 ,respectively. Next, the combined column is created which stores the value of sum scores from different dictionaries. Votes > 0 indicates positive sentiment which is flagged and presented in “Flag” Column. Votes < 0 indicates negative sentiment and flagged as -1 in the “Flag” column.
Above table illustrates how votes score can be calculated.

If the word has the contradiction (illustrated in table row#9) among interested dictionaries. For example, “grape” is unknown in Bing dictionary, the vote score is given equal to zero. “Grape” is positive sentiment in NRC but it is negative sentiment in AFINN. The total score will be zero. This contradiction word will be neglected in the prediction model.

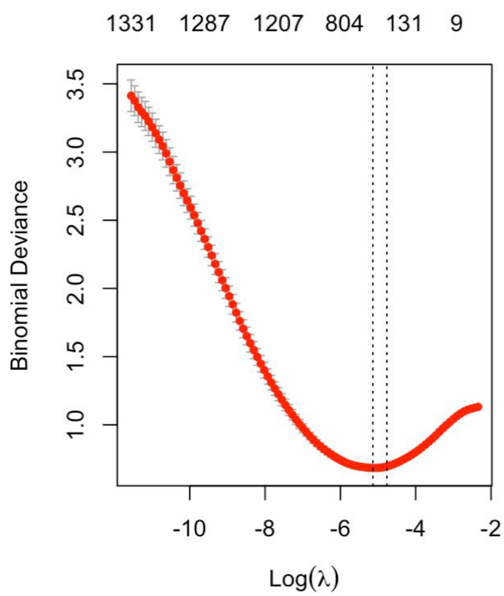
Similarly for the unmatching words with the dictionaries, the vote scores are given to zero where it will be neglected in the model (illustrated in table row#10).

10) Combination: Naive Bayes

Area under the curve: 0.5038

Area under the curve: 0.6435

11) Combination: Logistic Regression with Lasso



Confusion matrix for Train dataset (Combined dictionary):

Prediction\Reference	1	-1
1	3675	551
-1	58	716

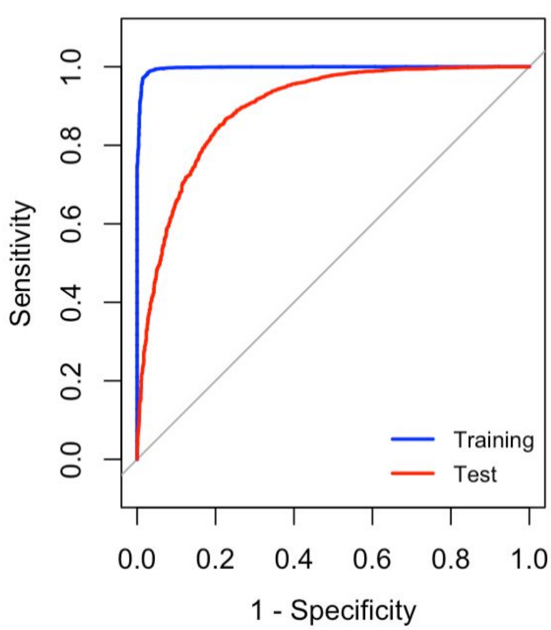
Accuracy : 0.8782

Confusion matrix for Test dataset (Combined dictionary):

Prediction\Reference	1	-1
1	3676	643
-1	88	593

Accuracy : 0.8538

12) Combination: Random Forest



Confusion matrix for Train dataset (Combined dictionary):

Prediction\Reference	1	-1
1	1178	89
-1	15	3718

Accuracy : 0.9792

Confusion matrix for Test dataset (Combined dictionary):

Prediction\Reference	1	-1
1	671	565
-1	119	3645

Accuracy : 0.8632

(ii) Develop models using a broader list of terms (i.e. not restricted to the dictionary terms only) – how do you obtain these terms? Will you use stemming here? Report on performance of the models. Compare performance with that in part (c) above. How do you evaluate performance? Which performance measures do you use, why.

We developed models using a broader list of terms by using left_join() function on the Bing dictionary.

In this assignment we will use lemmatization technique instead of stemming because the meaning of words is necessary for mapping the sentiment.

	Naive Bayes	**Glm(Logistic Regression) with Lasso**	**Random forest**
Bing (Left Join)	0.5006	0.8616	0.8772
Bing (Inner Join)	0.6981	0.8592	0.8724

The above table shows the Comparison of the performance of the models based on the use of a broader set of terms beyond the Bing dictionary and terms only limited to the Bing dictionary.
We have listed the accuracies of the test dataset for comparing the performance of the models.
For Naive Bayes model the accuracy is better for model using only dictionary terms.
For Logistic Regression, the accuracy is slightly better for model using a broader set of terms beyond the dictionary terms.
For Random Forest, the accuracy for model using only dictionary terms is almost close to the accuracy of the model using a broader set of terms.
Overall, for the above table, the Random forest model shows better accuracy with 0.877 where it uses a broader set of terms beyond the dictionary.

Comparison of performance of 3 different models using 3 different dictionaries with performance in part (c)

	Stars	Model		
		Naive Bayes	Glm(Logistic Regression) with Lasso	Random forest
Bing	0.728	0.698	0.859	0.872
NRC	0.679	0.501	0.834	0.853
AFINN	0.821	0.710	0.850	0.856
Combined	0.794	0.644	0.854	0.863

NRC is the least accurate dictionary based on the star rating and sentiment score for the dictionary terms. As it can be seen on the above table,
AFINN is the most accurate dictionary based on the star rating and sentiment score for the dictionary terms.
Comparing these part C results with the models we obtain better accuracy for the Random Forest model using the Bing dictionary.