

IDS 572

Assignment 3

Analyzing text in Yelp reviews - Text mining, Sentiment analysis

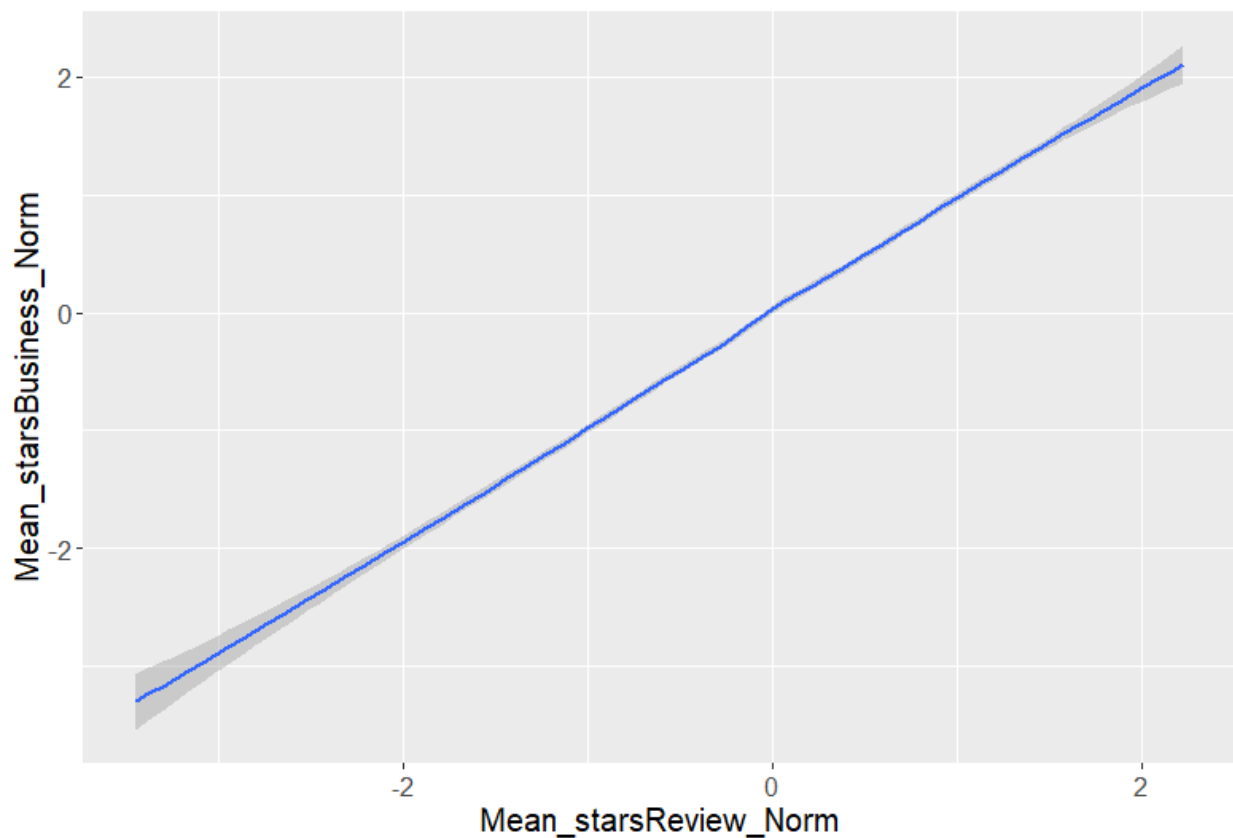


Ketaki Rane
Sanidhya Armarkar
Raj Mehta

1. Explore the data.

(a) How does star ratings for reviews relate to the star-rating given in the dataset for businesses (attribute 'businessStars')? Can one be calculated from the other?

```
# A tibble: 544 x 3
  business_id      mean_starsReview mean_starsBusiness
  <chr>          <dbl>          <dbl>
1 --FBCX-N37CMYDfs790Bnw      3.92            4
2 -865Ps6xb3h1LP67JcQ3mA      3.07            3
3 -9yG5SmYxH8BLg4bML8VQg      3.93            4
4 -guxo51AuUa_J8RuJ70EWg      3.57            3.5
5 -ITj6Pu8Gdw8MmLf0XBKQ      3.85            4
6 -K_qGSp7q61BkkjP_FSsEQ      3.25            3
7 -K3kqmykK1h1B4arCsLHOW      2.86            3
8 -lJtyCOTVINwusU9YF120A      3.43            3.5
9 -OEIW0do96-492qa_luxaw      4.05            4
10 -pG09M4JLQJHiaCORqCgbQ      2.72            3
# ... with 534 more rows
```



We performed a group by business id to determine the average starReview and average starBusiness. In order to calculate the average starReview and average starBusiness covariance, normalization was also utilized. We got to the conclusion that there is a significant association between these two evaluations. We can calculate from one another with certainty because there is a strong positive correlation between the two.

(b) Here, we will focus on star ratings for reviews. How are star ratings distributed? How will you use the star ratings to obtain a label indicating 'positive' or 'negative' – explain using the data, summaries, graphs, etc.?

The data we're examining is based on Yelp user evaluations of various businesses.

```
# A tibble: 5 x 2
# Groups:   starsReview [5]
  starsReview     n
    <int> <int>
1         1  5284
2         2  4649
3         3  6418
4         4 12348
5         5 16301
```

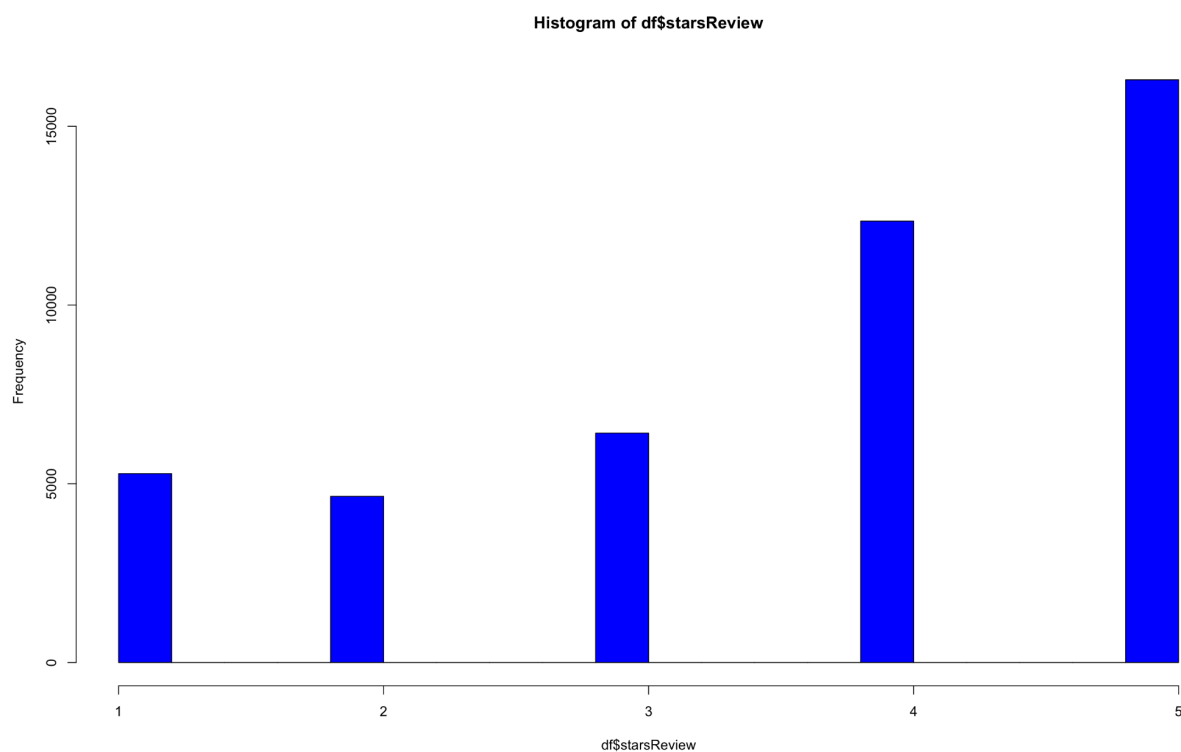


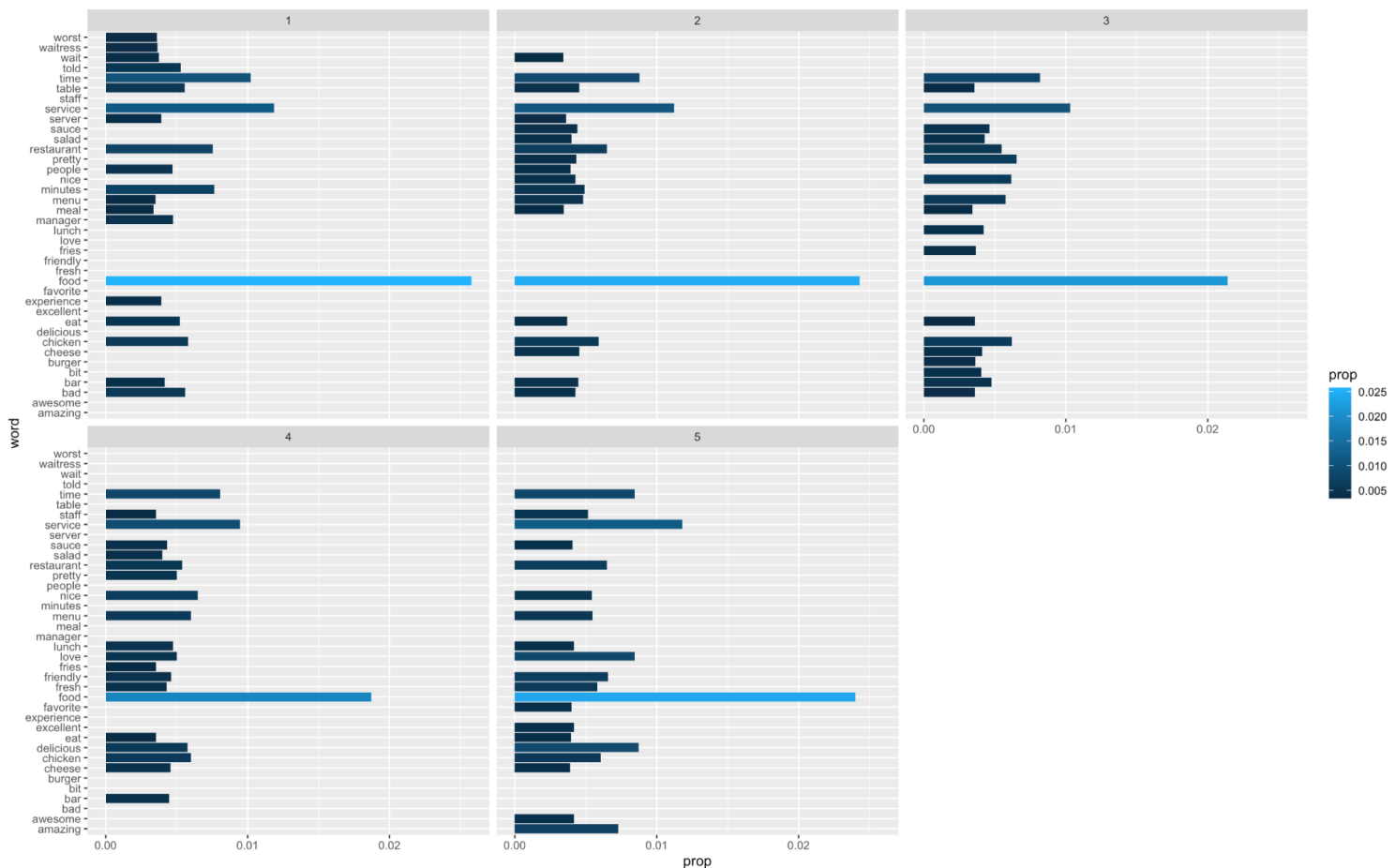
Figure (i.2) shows the overall distribution of star reviews, whereas Figure (i.1) shows the precise amount of ratings for each star review. Ratings range from 3 to 5, with 4 or 5 being the majority. Therefore, we've decided that ratings 1, 2, and 3 will be seen negatively, and ratings 4 and 5 will be viewed favorably.

2. What are some words in the restaurant reviews indicative of positive and negative sentiment – identify at least 20 in each category. One approach for this is to determine the average star rating for a word based on star ratings of documents or reviews where the word occurs. Do these ‘positive’ and ‘negative’ words make sense in the context of user reviews for restaurants being considered? (For this, since we’d like 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).

Before continuing with the analysis, we reduced tokens using the following criteria:-

1. Words that were absent from ten or more texts were eliminated.
2. Words with three characters or fewer and those longer than 15 characters
3. The digitized terms have been eliminated.
4. The token's stop words were eliminated

- We calculated the percentage of each term for each starReview.
- The average star rating for the 20 words that appeared most frequently in reviews was then used to construct the graph.
- Before removing common words across starReview, we can see that words like "food," "time," "restaurant," and "service" have a greater presence in the text.



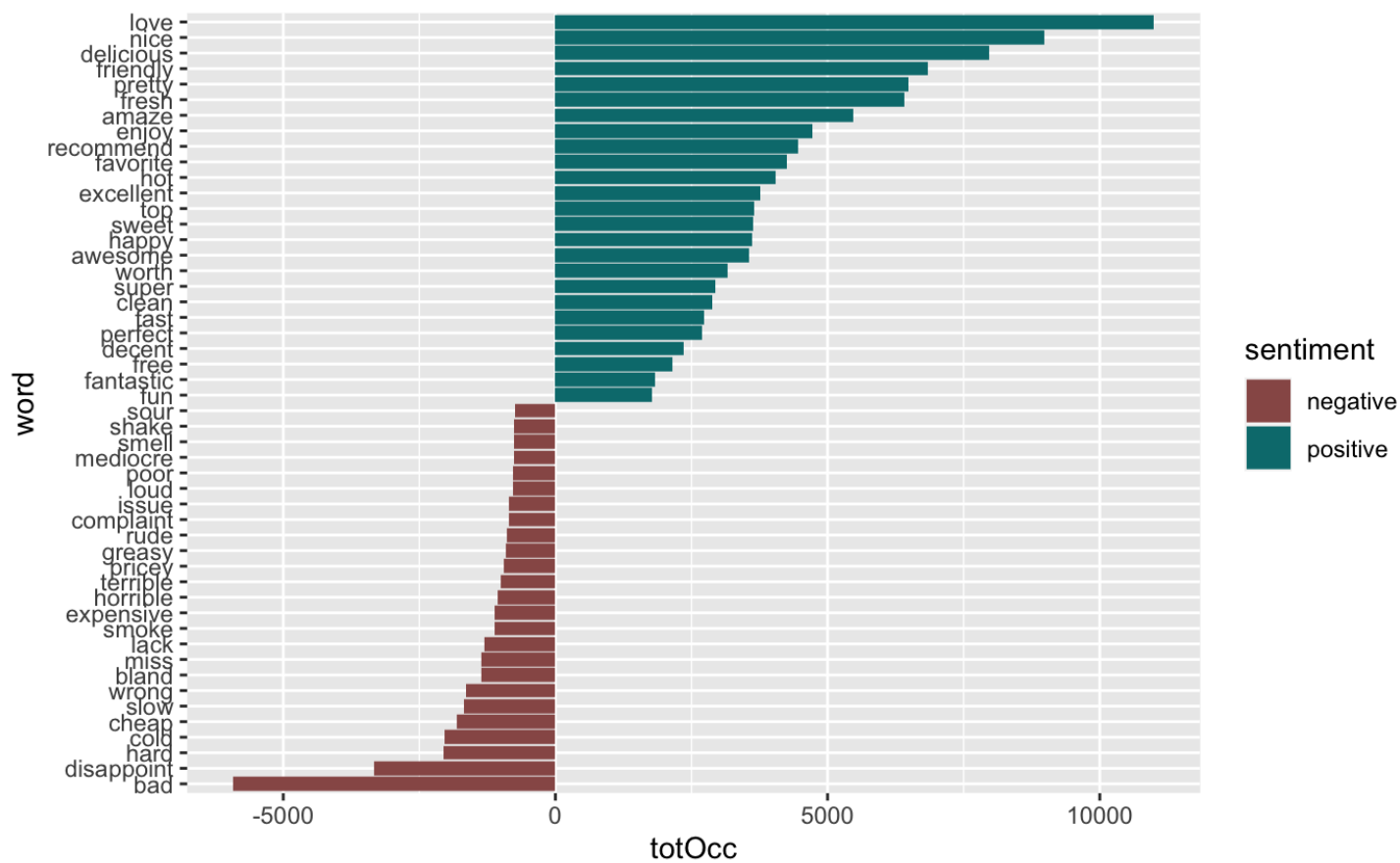
Below, you can see these graphs after we eliminated words that were frequently used throughout starReview.



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

Yes, in the context of customer feedback, the labels Positive and Negative make sense. Because the use of words like amaze, yummy, luscious, worthy, delightful, and wonderful tends to make reviews more positive while the use of words like useless, rude, horrible, weird, poor, and terrible tends to make reviews more unfavorable.

We plotted the top 25 positive and negative words after using an inner join to maintain only matched words and a Bing dictionary to determine each word's overall frequency.



Similar to this, we tallied the instances of each word for each sentiment before plotting the graph. However, we must translate these emotions into positive and negative evaluations of NRC. The i.c0 section contains the logic.

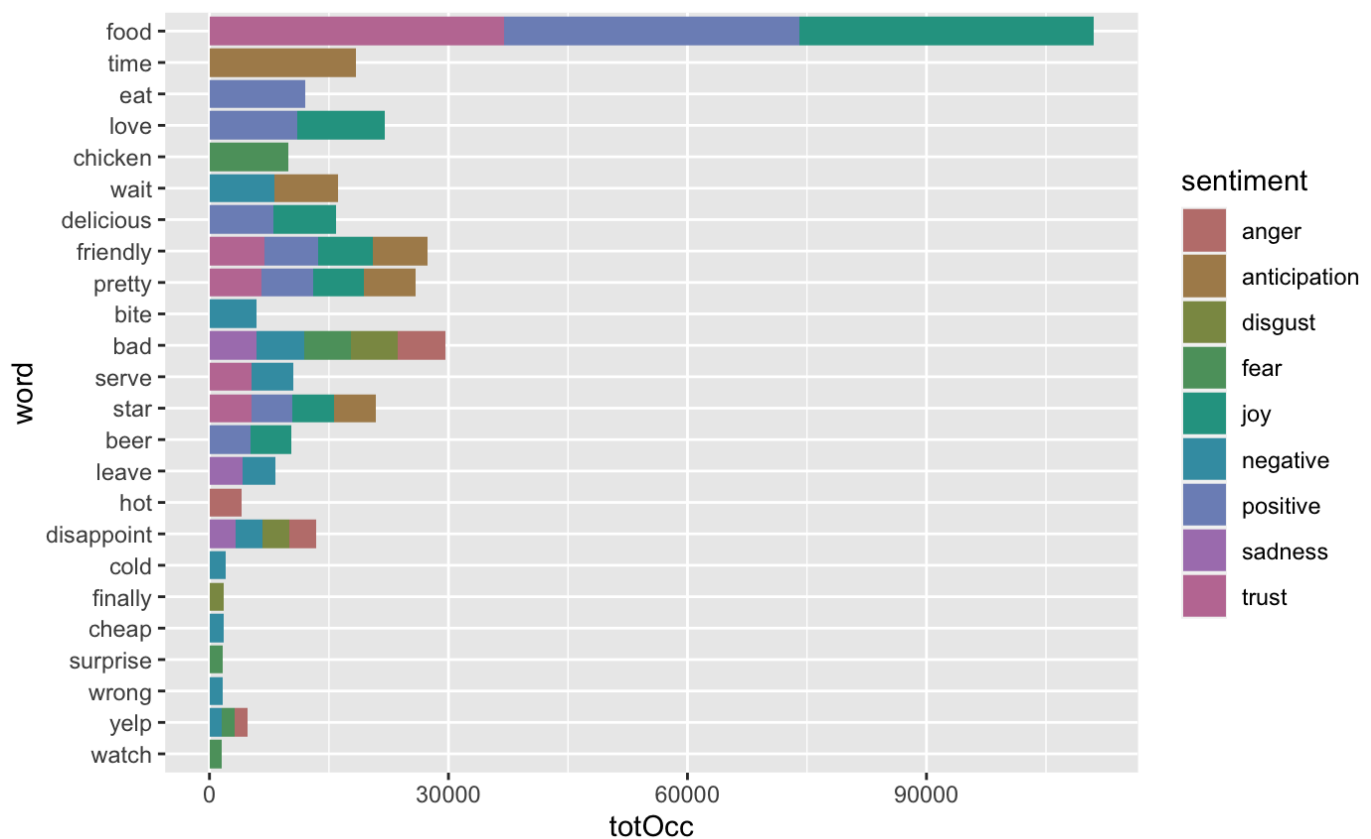


figure i.18

(a) How many matching terms (i.e. terms in your data which match the dictionary terms) are there for each of the dictionaries?

Dictionary	Positive Matching Word	Negative Matching Word	Total Matching Word
Bing	525	662	1187
NRC	653	781	1639
AFINN	393	253	646

- For NRC, we first divided "anger," "disgust," "fear," "sadness," "negative," "positive," "joy," "anticipation," and "trust" into positive and negative attitudes.
- We utilized value for AFINN. For each word, positive if value > 0, else negative.

(b) What is the overlap in matching terms between the different dictionaries? Based on this, do you think any of the three dictionaries will be better at picking up sentiment information from your text of reviews?



Definition: Overlap in matching terms provides a measure of the level of duplication between different dictionaries. In other words, the terms that appear frequently in all three dictionaries. Even if we eliminate every term that overlaps in these three dictionaries, the NRC dictionary will still have a higher word count and be able to extract more sentiment data.

(c) Consider the positive and negative terms you determined in Q 2 above; which of these terms match with terms in each of the three dictionaries?

Matching 1187 distinct terms from Bing Dictionary

	word
1	enjoyable
2	hard
3	memorable
4	negative
5	fast
6	happy
7	improve
8	pan
9	favorite
10	solid
11	crisp
12	love
13	enjoy
14	exceptional
15	friendly
16	healthy
17	recommend
18	worth
19	attentive
20	awkward

Matching 1639 distinct terms from NRC Dictionary

 word 	
1	food
2	job
3	memorable
4	negative
5	serve
6	star
7	delivery
8	happy
9	improve
10	bite
11	favorite
12	solid
13	chicken
14	crisp
15	forget
16	green
17	honey
18	love
19	blue
20	enjoy

Matching 646 distinct terms from AFINN Dictionary

	word
1	hard
2	negative
3	stop
4	happy
5	improve
6	favorite
7	solid
8	forget
9	love
10	yummy
11	enjoy
12	friendly
13	healthy
14	recommend
15	worth
16	awkward
17	excellent
18	goodness
19	impress
20	pay

4. Consider a basic approach (not developing a predictive model like a decision tree, random forests etc.) to use the dictionary based on positive and negative terms to predict sentiment (positive or negative based on star rating) of a review. One approach for this is: based on 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.

(a) Describe how you obtain the aggregated scores, and predictions based on these scores

(b) What is the performance of this approach (for each dictionary). Does any dictionary perform better?

We began by determining the average score, which we accomplished by using

1) The ratio of positive and negative words is different in the Bing dictionary

- 2) The entire list of uplifting terms in the NRC Dictionary
- 3) The sum of all Afinn Dictionary evaluations

The remaining reviews were then changed to good (>3) and negative (3) labels, which we consider to be the true outcome, after the reviews with three stars were erased (hiLo).

For all dictionaries, we labeled positive when the aggregated score was greater than zero and vice versa.

Then, we generated a confusion matrix using the actual and anticipated hiLo labels. Visit i.c2 to get a confusion matrix for all dictionaries.

Are you able to predict review sentiment based on these aggregated scores, and how do they perform?

Yes, as was noted in the inquiry before this one, we could predict review sentiment based on score. Let's examine each dictionary's performance.

Does any dictionary perform better?

Based on AUC, Confusion Matrix, and ROC Curve, we can say that the Bing dictionary performs better than other dictionaries.

5. 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). You should consider models built using only the terms matching the sentiment dictionaries, as well as 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 choiceLasso logistic regression (why Lasso?), xgb, random forest (use ranger for faster run-times) – use the same three modeling techniques with each of the dictionaries, with the combination of dictionary terms, and wit the broader set of terms.

(a)How do you evaluate performance? Which performance measures do you use, why?

(b)Which types of models does your team choose to develop, and why? Do you use term frequency, tfidf, or other measures, and why?

(c) Develop models using only the sentiment dictionary terms – try the three different dictionaries; how do the dictionaries compare in terms of predictive performance? Then with a combination of the three dictionaries, ie. combine all dictionary terms. What is the size of the document-term matrix? Should you use stemming or lemmatization when using the dictionaries? Why?

(d) 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 or lemmatization here, and why?

(e) Compare performance of the models. How does performance here relate to that from Question 4 above. Explain your findings (and is this what you expected).

We used word frequency and tfidf to reduce the size of the data set and assess the model. These two terms were chosen because they help us determine if the feedback on the document will be more favorable or unfavorable.

In most other situations, stemming is a good idea because it decreases the amount of traits that must be learned, but we didn't use stemming in this situation.

Although it would have taken less time to fit the model and we would have had less features as a result of using stemming.

Similar to the previous task, before moving on to train the model, we first chose the ideal threshold for each phrase.

Random Forest:

The best threshold for each dictionary is shown in Figure i.25.

```
> bThrBing<-coords(rocTrnBing, "best", ret="threshold", transpose = FALSE)
> as.numeric(as.character((bThrBing)))
[1] 0.746859
> bThrNRC<-coords(rocTrnNRC, "best", ret="threshold", transpose = FALSE)
> as.numeric(as.character((bThrNRC)))
[1] 0.7294011
> bThrAFINN<-coords(rocTrnAFINN, "best", ret="threshold", transpose = FALSE)
> as.numeric(as.character((bThrAFINN)))
[1] 0.7560527

> table(actual=revDTM_sentiBing_trn$hiLo, preds=revSentiBing_predTrn[,2]>bThrBing[1,1])
      preds
actual FALSE  TRUE
   -1   4339   426
    1   2360 11649

> table(actual=revDTM_sentiBing_tst$hiLo, preds=revSentiBing_predTst[,2]>bThrBing[1,1])
      preds
actual FALSE  TRUE
   -1   4306   559
    1   2558 11351

> #NRC
> table(actual=revDTM_sentiNRC_trn$hiLo, preds=revSentiNRC_predTrn[,2]>bThrNRC[1,1])
      preds
actual FALSE  TRUE
   -1   4021   920
    1   1567 12652

> table(actual=revDTM_sentiNRC_tst$hiLo, preds=revSentiNRC_predTst[,2]>bThrNRC[1,1])
      preds
actual FALSE  TRUE
   -1   3920  1025
    1   1893 12322

> #AFINN
> table(actual=revDTM_sentiAFINN_trn$hiLo, preds=revSentiAFINN_predTrn[,2]>bThrAFINN[1,1])
      preds
actual FALSE  TRUE
   -1   4166   548
    1   2663 10988

> table(actual=revDTM_sentiAFINN_tst$hiLo, preds=revSentiAFINN_predTst[,2]>bThrAFINN[1,1])
      preds
actual FALSE  TRUE
   -1   4059   629
    1   2909 10769
```

```

> table(actual=revDTM_sentiBing_trn$hiLo, preds=revSentiBing_predTrn[,2]>0.5)
      preds
actual FALSE  TRUE
   -1  1023  3742
    1     4 14005
> table(actual=revDTM_sentiBing_tst$hiLo, preds=revSentiBing_predTst[,2]>0.5)
      preds
actual FALSE  TRUE
   -1   883  3982
    1    14 13895
> table(actual=revDTM_sentiNRC_trn$hiLo, preds=revSentiNRC_predTrn[,2]>0.5)
      preds
actual FALSE  TRUE
   -1  1044  3897
    1     4 14215
> table(actual=revDTM_sentiNRC_tst$hiLo, preds=revSentiNRC_predTst[,2]>0.5)
      preds
actual FALSE  TRUE
   -1   799  4146
    1    15 14200
> table(actual=revDTM_sentiAFINN_trn$hiLo, preds=revSentiAFINN_predTrn[,2]>0.5)
      preds
actual FALSE  TRUE
   -1  1370  3344
    1    26 13625
> table(actual=revDTM_sentiAFINN_tst$hiLo, preds=revSentiAFINN_predTst[,2]>0.5)
      preds
actual FALSE  TRUE
   -1  1198  3490
    1    49 13629
> rocTrnBing <- roc(revDTM_sentiBing_trn$hiLo, revSentiBing_predTrn[,2], levels=c(-1, 1))

```

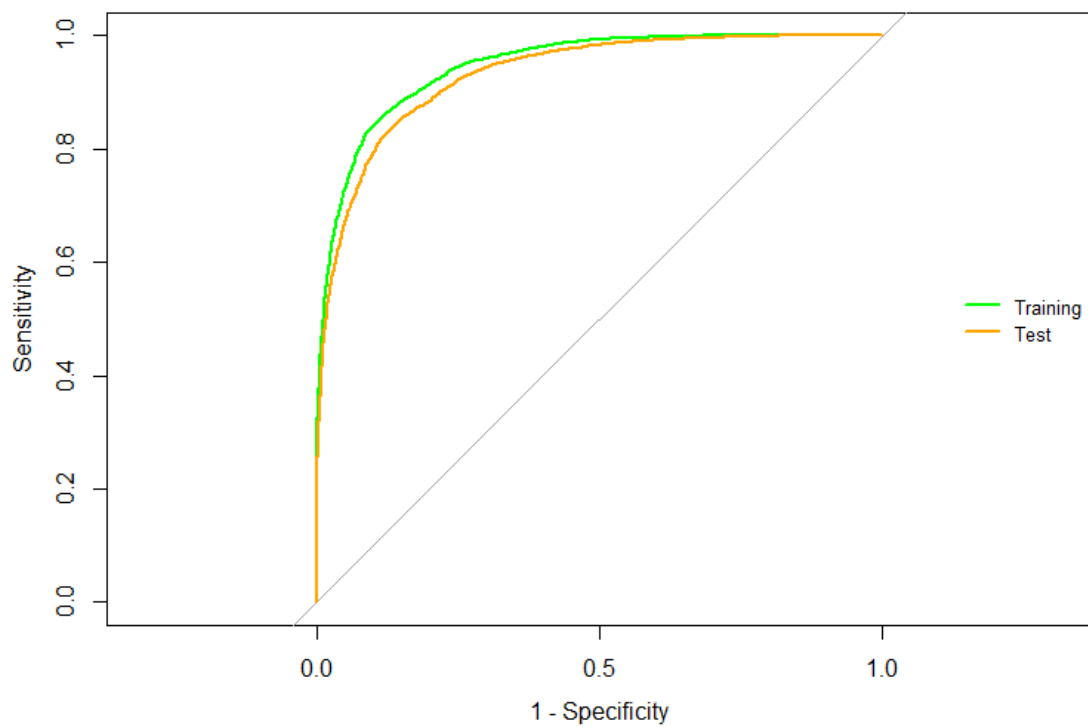
The training and test confusion matrix for all three dictionaries is illustrated in Figure i.26.

```

> #AUC
> auc(as.numeric(revDTM_sentiBing_trn$hiLo), revSentiBing_predTrn[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.9453
> auc(as.numeric(revDTM_sentiBing_tst$hiLo), revSentiBing_predTst[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.9299
> auc(as.numeric(revDTM_sentiNRC_trn$hiLo), revSentiNRC_predTrn[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.9385
> auc(as.numeric(revDTM_sentiNRC_tst$hiLo), revSentiNRC_predTst[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.9172
> auc(as.numeric(revDTM_sentiAFINN_trn$hiLo), revSentiAFINN_predTrn[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.9256
> auc(as.numeric(revDTM_sentiAFINN_tst$hiLo), revSentiAFINN_predTst[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.9058

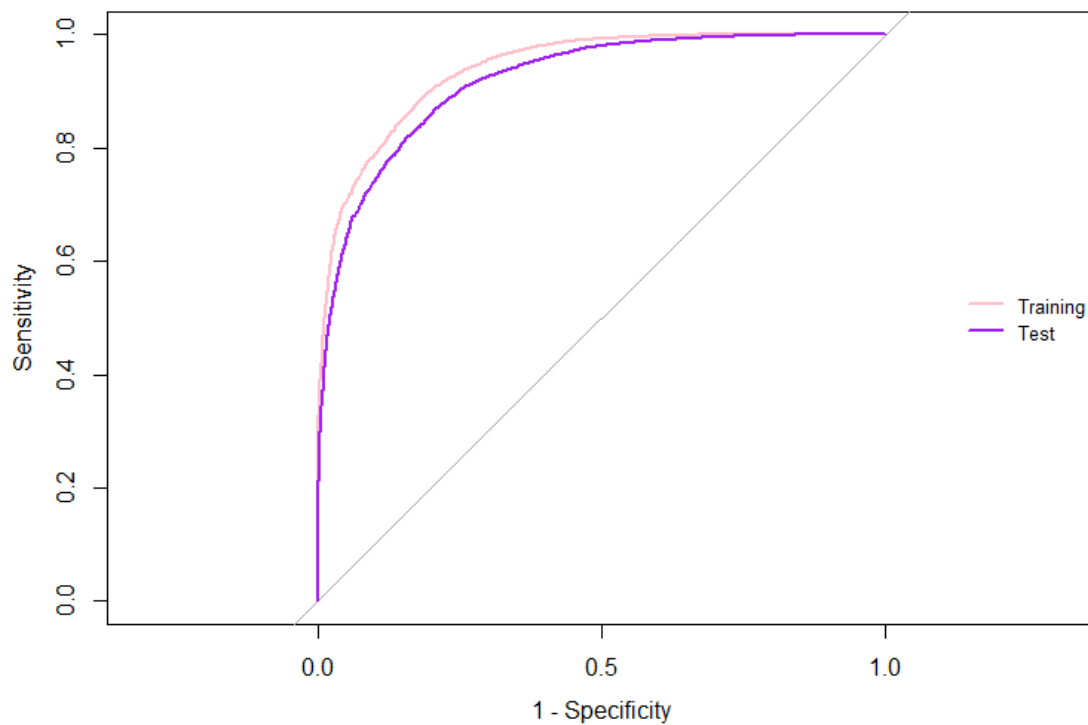
```

The training and test AUC values for all three dictionaries are displayed in Figure i.27.



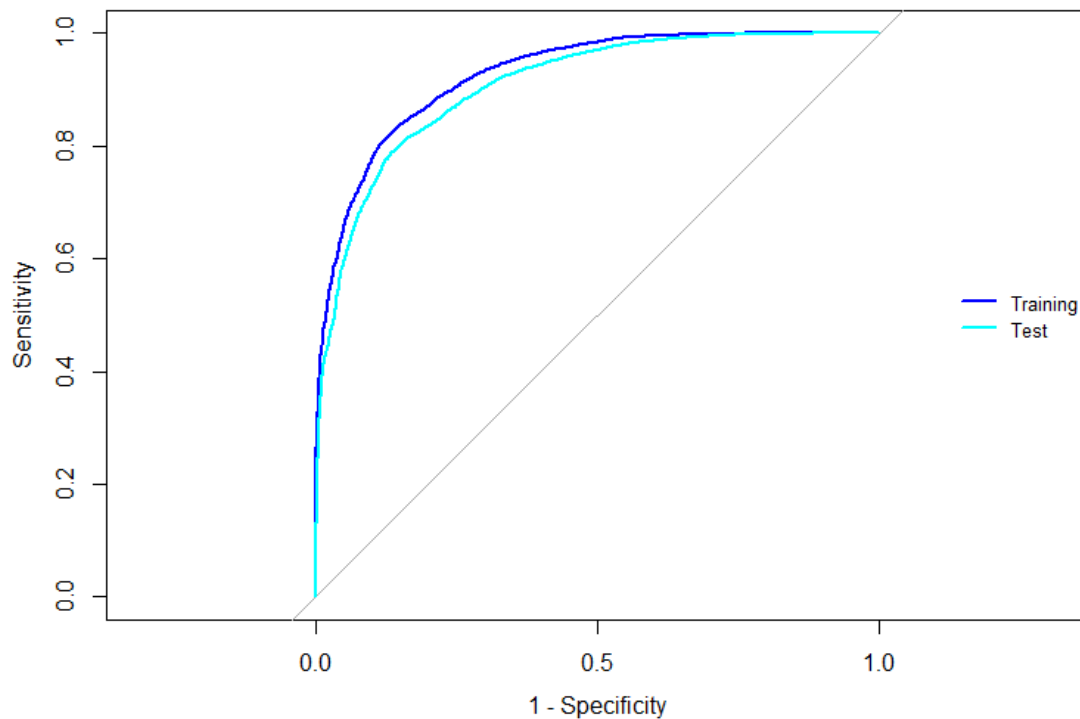
BING

The training and test ROC curves for the Bing dictionary are given in Figure i.28.



NRC

The training and test ROC curves for the NRC dictionary are given in Figure i.29.



AFFIN

The training and test ROC curves for the AFFIN dictionary are given in Figure i.30.

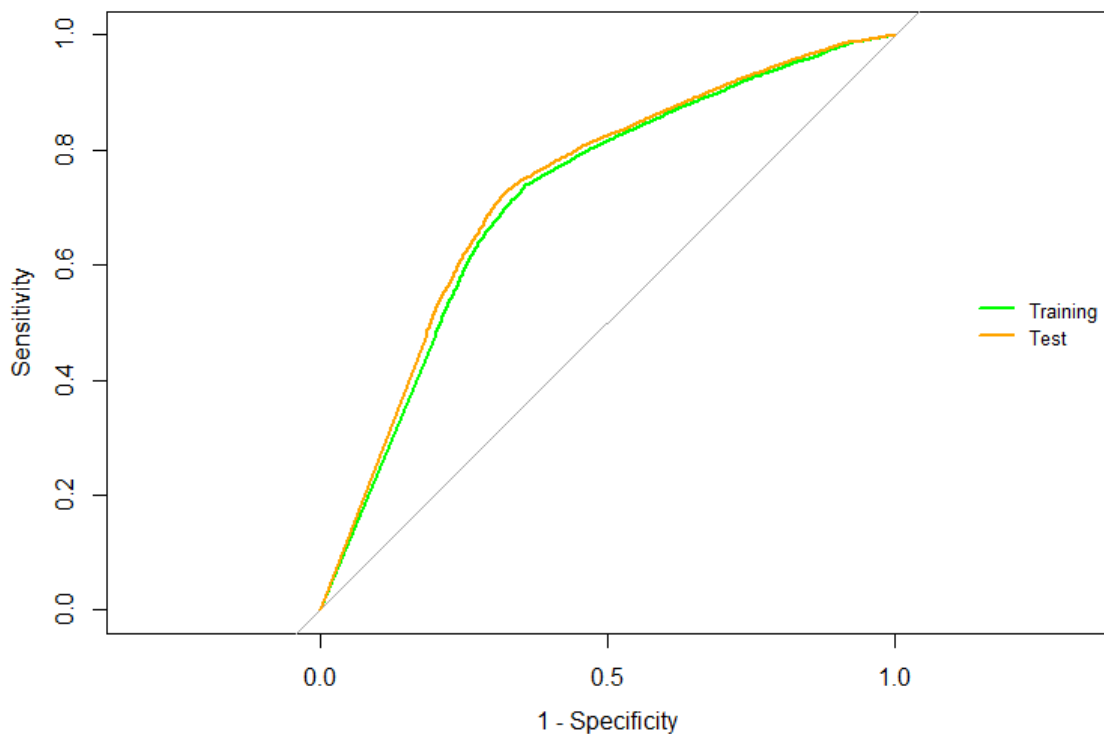
Naive Bias:

```
Setting direction: controls < cases
Area under the curve: 0.7155
> auc(as.numeric(revDTM_sentiBing_tst$hiLo), revSentiBing_NBpredTst[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.7284
> auc(as.numeric(revDTM_sentiNRC_trn$hiLo), revSentiNRC_NBpredTrn[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.6873
> auc(as.numeric(revDTM_sentiNRC_tst$hiLo), revSentiNRC_NBpredTst[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.6997
> auc(as.numeric(revDTM_sentiAFINN_trn$hiLo), revSentiAFINN_NBpredTrn[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.7423
> auc(as.numeric(revDTM_sentiAFINN_tst$hiLo), revSentiAFINN_NBpredTst[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.7466
```

The training and test confusion matrix for all three dictionaries is given in Figure i.31.

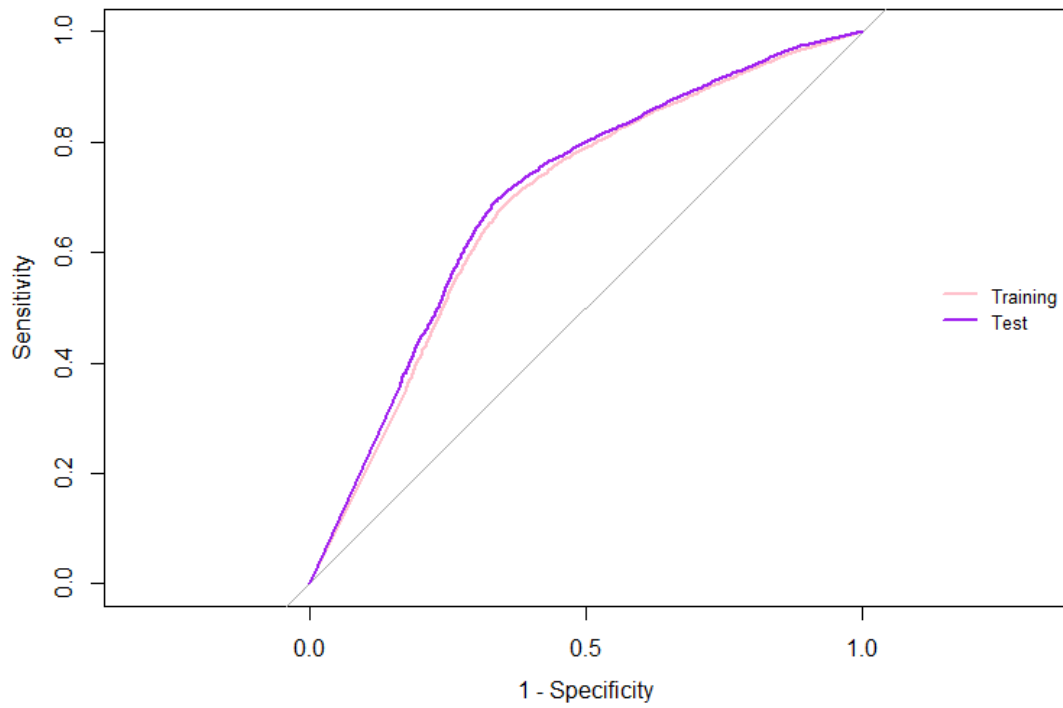
```
> #Bing
> table(actual=revDTM_sentiBing_trn$hiLo, preds=revSentiBing_NBpredTrn[,2]>0.5)
      preds
actual FALSE TRUE
-1     3648 1117
 1     6277 7732
> table(actual=revDTM_sentiBing_tst$hiLo, preds=revSentiBing_NBpredTst[,2]>0.5)
      preds
actual FALSE TRUE
-1     3800 1065
 1     6195 7714
> #NRC
> table(actual=revDTM_sentiNRC_trn$hiLo, preds=revSentiNRC_NBpredTrn[,2]>0.5)
      preds
actual FALSE TRUE
-1     3957  984
 1     8430 5789
> table(actual=revDTM_sentiNRC_tst$hiLo, preds=revSentiNRC_NBpredTst[,2]>0.5)
      preds
actual FALSE TRUE
-1     4009  936
 1     8330 5885
> #AFINN
> table(actual=revDTM_sentiAFINN_trn$hiLo, preds=revSentiAFINN_NBpredTrn[,2]>0.5)
      preds
actual FALSE TRUE
-1     3473 1241
 1     4527 9124
> table(actual=revDTM_sentiAFINN_tst$hiLo, preds=revSentiAFINN_NBpredTst[,2]>0.5)
      preds
actual FALSE TRUE
-1     3478 1210
 1     4459 9219
```

The training and test AUC values for all three dictionaries are displayed in Figure i.32.



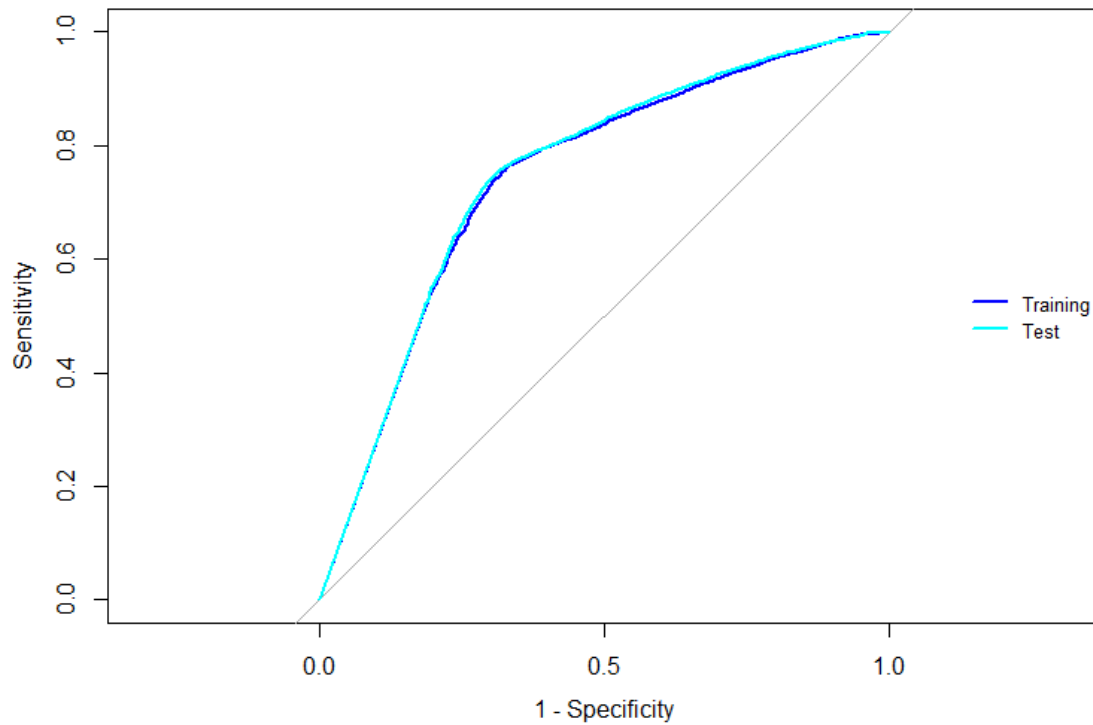
BING

The training and test ROC curves for the Bing dictionary are given in Figure i.33.



NRC

The training and test ROC curves for the NRC dictionary are given in Figure i.34.



AFFIN

The training and test ROC curves for the AFINN dictionary are given in Figure i.35.

SVM :

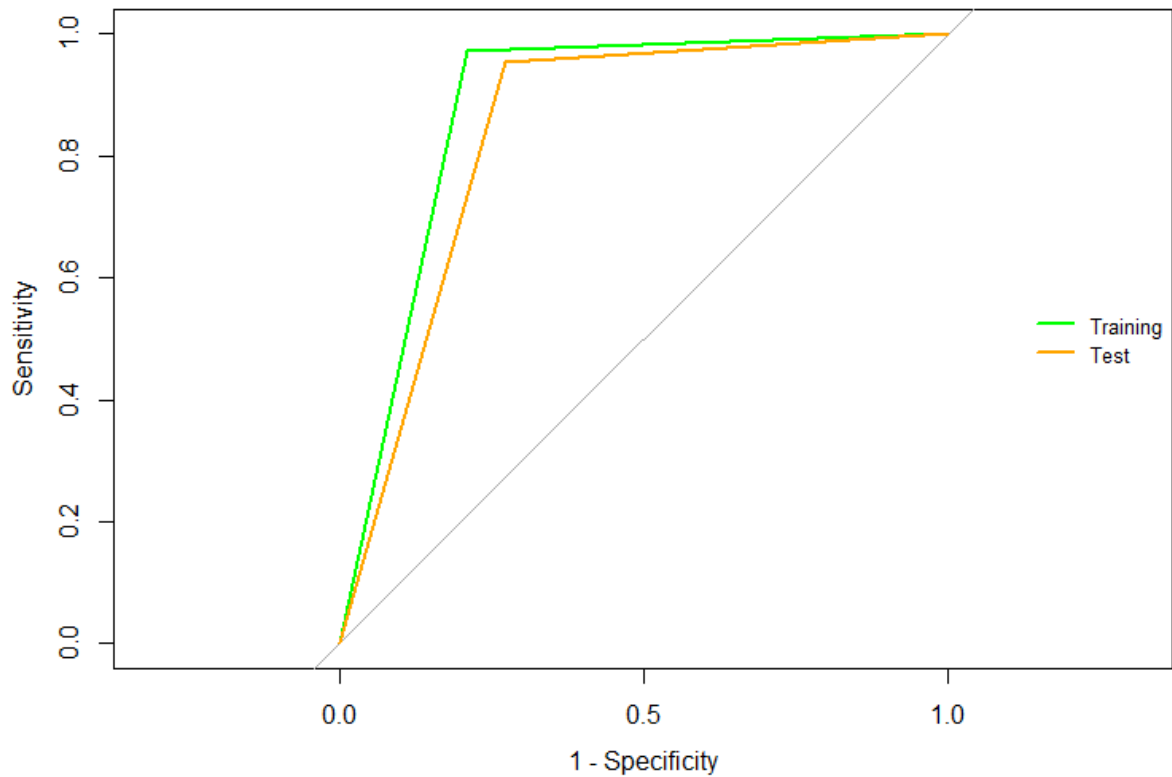
We then created the SVM model, which will be discussed in further depth further below.

```
> #Parameter Tuning
> system.time(svm_tuneBing <- tune(svm, as.factor(hiLo) ~., data = revDTM_sentiBing_trn_2 %>% select(-review_id), kernel="radial", scale
=FALSE, ranges = list( cost=c(0.1,1,10), gamma = c(0.5,1,5))))
  user  system elapsed
572.60   15.13  1623.16
```

BING

```
> #AUC
> auc(as.numeric(revDTM_sentiBing_trn_2$hiLo), as.numeric(revDTM_predTrn_svmBing))
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.88
> auc(as.numeric(revDTM_sentiBing_tst_2$hiLo), as.numeric(revDTM_predTst_svmBing))
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.8401
> #Bing
> table(actual= revDTM_sentiBing_trn_2$hiLo, predicted= revDTM_predTrn_svmBing)
      predicted
actual   -1    1
   -1 2048  548
    1  214 7190
> table(actual= revDTM_sentiBing_tst_2$hiLo, predicted= revDTM_predTst_svmBing)
      predicted
actual   -1    1
   -1 1868  704
    1  343 7085
```

Figure i.37 shows the AUC values and Confusion Matrix for the Bing dictionary using the SVM model.



BING

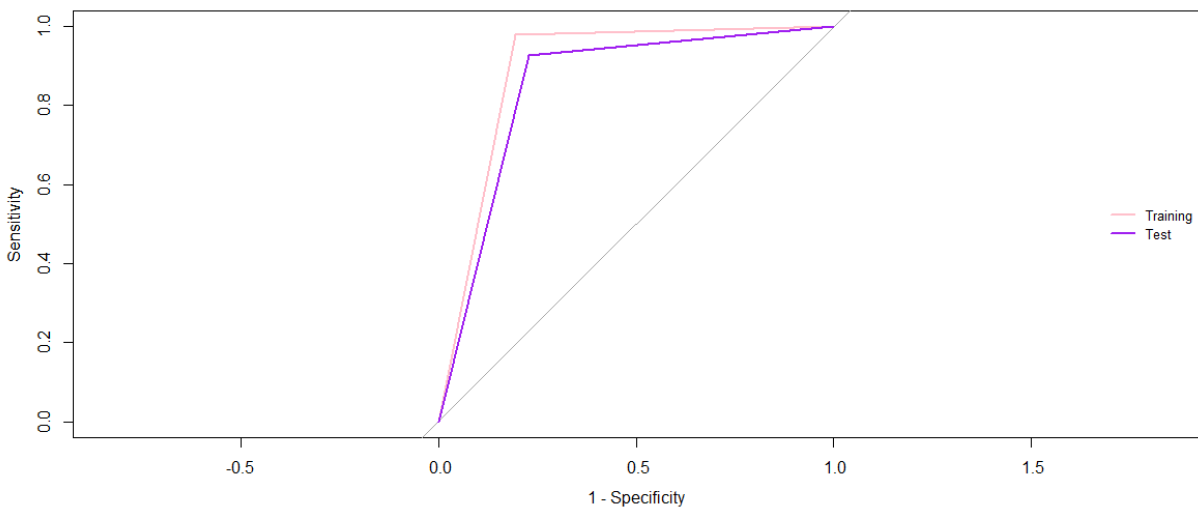
The ROC Curve for the Bing Dictionary using the SVM model is illustrated in Figure i.38.

NRC :

```
> #Parameter Tuning
> system.time(svm_tuneNRC <- tune(svm, as.factor(hiLo) ~., data = revDTM_sentiNRC_t
rn_2 %>% select(-review_id), kernel="radial", scale=FALSE, ranges = list( cost=c(0.
1,1,10), gamma = c(0.5,1,5))))
   user  system elapsed
977.83   22.53  2644.86
```

Figure i.39

Figure i.39 shows the AUC values and Confusion Matrix for the NRC dictionary using the SVM model.



NRC

Figure i.40

The ROC Curve using the SVM model for the NRC dictionary is given in Figure i.40.

```
> #AUC
> auc(as.numeric(revDTM_sentiNRC_trn_2$hiLo), as.numeric(revDTM_predTrn_svmNRC))
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.8929
> auc(as.numeric(revDTM_sentiNRC_tst_2$hiLo), as.numeric(revDTM_predTst_svmNRC))
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.8505
> #Confusion Matrix
> #NRC
> table(actual= revDTM_sentiNRC_trn_2$hiLo, predicted= revDTM_predTrn_svmNRC)
      predicted
actual -1    1
-1  2038  490
 1   152 7320
> table(actual= revDTM_sentiNRC_tst_2$hiLo, predicted= revDTM_predTst_svmNRC)
      predicted
actual -1    1
-1  2904  853
 1   449 5794
```

Figure i.41

Figure i.41 shows the AUC values and Confusion Matrix for the NRC dictionary using the SVM model.

Afinn

```
> #Parameter Tuning
> system.time(svm_tuneAFINN <- tune(svm, as.factor(hiLo) ~., data = revDTM_sentiAFI
NN_trn_2 %>% select(-review_id), kernel="radial", scale=FALSE, ranges = list( cost
=c(0.1,1,10), gamma = c(0.5,1,5))))
      user  system elapsed
817.15   16.42  1190.13
```

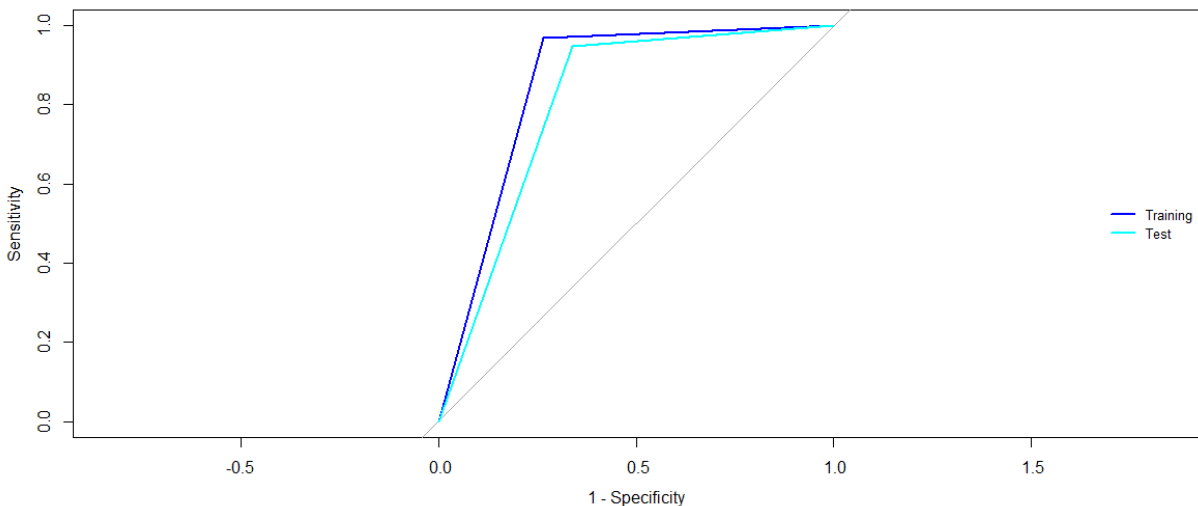
Figure i.42

```

> #AUC
> auc(as.numeric(revDTM_sentiAFINN_trn_2$hiLo), as.numeric(revDTM_predTrn_svmAFINN))
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.8522
> auc(as.numeric(revDTM_sentiAFINN_tst_2$hiLo), as.numeric(revDTM_predTst_svmAFINN))
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.8048
> #Confusion Matrix
> #AFINN
> table(actual= revDTM_sentiAFINN_trn_2$hiLo, predicted= revDTM_predTrn_svmAFINN)
      predicted
actual -1    1
   -1 1829  658
    1  233 7280
> table(actual= revDTM_sentiAFINN_tst_2$hiLo, predicted= revDTM_predTst_svmAFINN)
      predicted
actual -1    1
   -1 1646  838
    1  399 7117

```

Figure i.43



AFINN

Figure i.44

The ROC Curve using the SVM model for the AFINN dictionary is given in Figure i.44.

Then with a combination of the three dictionaries, ie. combine all dictionary terms. Do you use term frequency, tfidf, or other measures, and why?

We have combined all three dictionaries into a single dictionary and developed random forest, Naive Bias, and SVM model

Random Forest

```

> #Best threshold from ROC analyses
> bThrCOM<-coords(rocTrnCOM, "best", ret="threshold", transpose = FALSE)
> as.numeric(as.character((bThrCOM)))
[1] 0.728536
> #Confusion Matrix
> table(actual=revDTM_sentiCOM_trn$hiLo, preds=revSentiCOM_predTrn[,2]>bThrCOM[1,
1])
      preds
actual FALSE TRUE
-1    2254   286
 1     611  6849
> table(actual=revDTM_sentiCOM_tst$hiLo, preds=revSentiCOM_predTst[,2]>bThrCOM[1,
1])
      preds
actual FALSE TRUE
-1    2143   408
 1     956  6493
> #AUC
> auc(as.numeric(revDTM_sentiCOM_trn$hiLo), revSentiCOM_predTrn[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.9703
> auc(as.numeric(revDTM_sentiCOM_tst$hiLo), revSentiCOM_predTst[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.9369

```

Figure i.45

Figure i.45 shows the AUC values and Confusion Matrix for the merged dictionary using the Random Forest model.

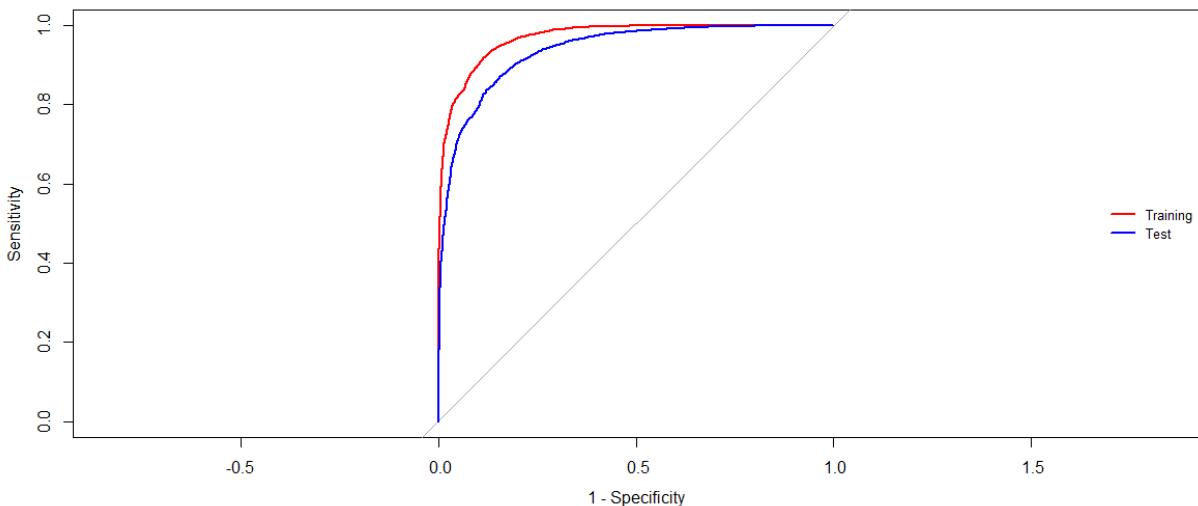


Figure i.46

The ROC Curve utilizing the Random Forest model for integrated dictionaries is displayed in Figure i.46.

Naïve Bias:

```

> #AUC
> auc(as.numeric(revDTM_sentiCOM_trn$hiLo), revSentiCOM_NBpredTrn[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.6804
> auc(as.numeric(revDTM_sentiCOM_tst$hiLo), revSentiCOM_NBpredTst[,2])
Setting levels: control = 1, case = 2
Setting direction: controls < cases
Area under the curve: 0.731
> #Confusion Matrix
> table(actual=revDTM_sentiCOM_trn$hiLo, preds=revSentiCOM_NBpredTrn[,2]>0.5)
      preds
actual FALSE TRUE
   -1   2062   478
    1   4580 2880
> table(actual=revDTM_sentiCOM_tst$hiLo, preds=revSentiCOM_NBpredTst[,2]>0.5)
      preds
actual FALSE TRUE
   -1   2196   355
    1   4507 2942

```

Figure i.47

The AUC values and Confusion Matrix for combined dictionaries employing Naive Bias are displayed in Figure i.47.

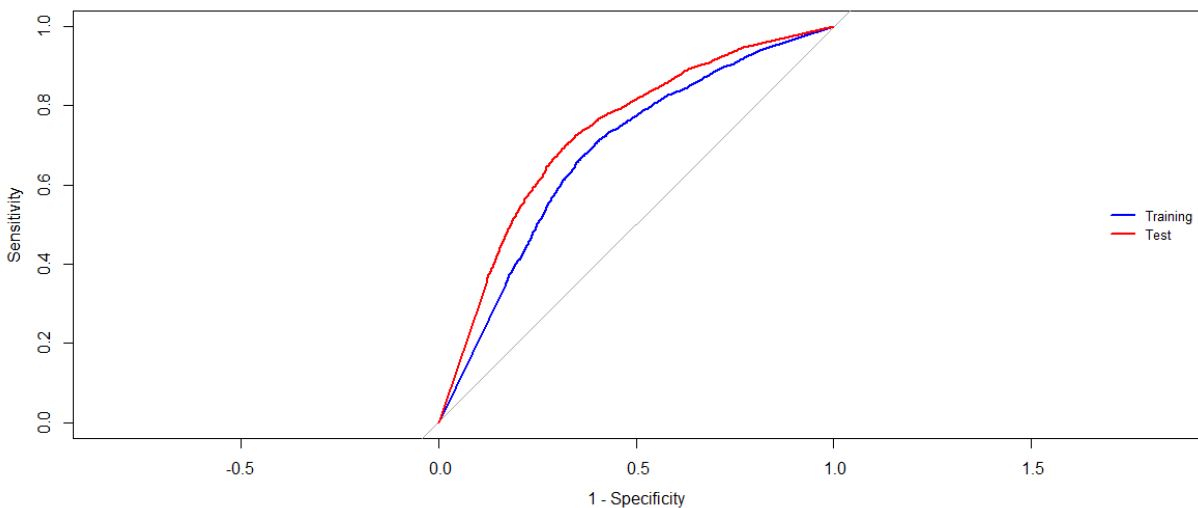


Figure i.48

The ROC Curve for a combined dictionary using the Naive model is displayed in Figure i.48.

SVM:

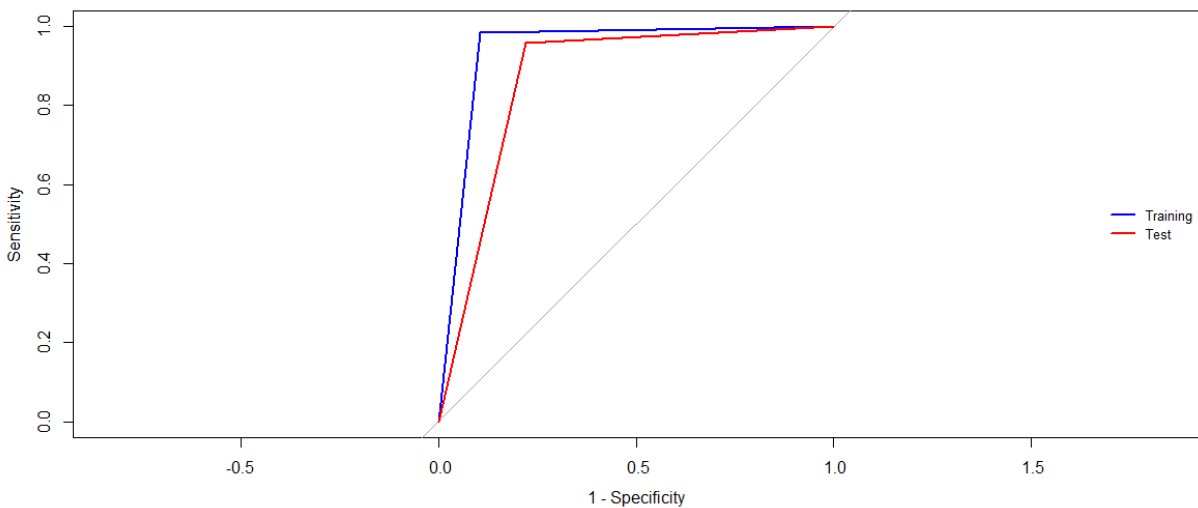


Figure i.49

The ROC Curve for combined dictionaries using SVM is displayed in Figure i.49.

6. Consider some of the attributes for restaurants – this is specified as a list of values for various attributes in the ‘attributes’ column. Extract different attributes (see note below).

(a) Consider a few interesting attributes and summarize how many restaurants there are by values of these attributes; examine if star ratings vary by these attributes.

Note: for question 6, you will consider the values in the ‘attribute’ column. This has values of multiple attributes, separated by a ‘|’. Further, some of the values, like Ambience, carry a list of True/False values (like, for example, Ambience: {'romantic': False, 'intimate': False, 'classy': False, 'hipster': False, ...}). Care must be taken to extract values for different attributes. You can consider developing a separate dataframe with review_id, attribute, and then process this further to extract values for the different attributes.

Ambience, BusinessParking, and GoodForMeal are three characteristics that we looked for.

For the **Ambience**:

'Casual' and 'trendy' have both appeared roughly 32036 and 1775 times, respectively.

	amb	n
1	'casual'	32306
2	c(" 'classy'", " 'upscale'")	490
3	character(0)	5092
4	'trendy'	1775
5	c(" 'classy'", " 'trendy'", " 'upscale'")	283
6	'touristy'	420
7	c(" 'touristy'", " 'casual'")	158
8	'intimate'	499
9	'classy'	757
10	{'romantic'	300
11	'divey'	1421
12	c(" 'hipster'", " 'trendy'")	237
13	'upscale'	223
14	c(" {'romantic'", " 'intimate'", " 'classy'", " 'c [...]	88
15	c(" 'divey'", " 'casual'")	136
16	c(" 'trendy'", " 'casual'")	159
17	c(" {'romantic'", " 'intimate'", " 'casual'")	42
18	c(" 'classy'", " 'trendy'")	131
19	c(" 'hipster'", " 'divey'", " 'touristy'", " 'casu [...]	52
20	'hipster'	121
21	c(" {'romantic'", " 'classy'")	34
22	c(" {'romantic'", " 'intimate'", " 'classy'")	36
23	c(" {'romantic'", " 'classy'", " 'trendy'", " 'ups [...]	34
24	c(" 'intimate'", " 'classy'", " 'trendy'", " 'casu [...]	56
25	c(" 'hipster'", " 'casual'")	34

Figure i.50

For the **Music**:

Words like 'background music' and 'live' have been used 1460 and 918 times, respectively.

	Music	n
1	character(0)	41427
2	'live'	918
3	'background_music'	1460
4	'jukebox'	514
5	{'dj'}	491
6	c(" 'background_music'", " 'live'")	74

For the **BusinessParking**:

Words like 'lot' and 'street' have been used 26365 and 6152 times, respectively.

	bsnsPrk	n
1	'lot'	26365
2	'valet'	806
3	c(" 'street'", " 'lot'")	1822
4	'street'	6152
5	character(0)	3858
6	c(" {'garage'", " 'street'", " 'lot'")	129
7	c(" {'garage'", " 'street'", " 'validated'", " 'va [...]	219
8	c(" {'garage'", " 'valet'")	860
9	c(" {'garage'", " 'street'")	366
10	{'garage'}	2631
11	c(" {'garage'", " 'lot'", " 'valet'")	384
12	c(" 'street'", " 'valet'")	587
13	c(" {'garage'", " 'street'", " 'validated'")	67
14	c(" 'street'", " 'lot'", " 'valet'")	133
15	c(" {'garage'", " 'street'", " 'valet'")	44
16	c(" 'lot'", " 'valet'")	221
17	'validated'	39
18	c(" {'garage'", " 'lot'")	114
19	c(" {'garage'", " 'validated'")	87

Figure i.51

For **GoodForMeal**:

Words like 'lunch' and 'dinner' have been used 8289 and 7359 times, respectively.

	GdFrMI	n
1	c(" 'lunch'", " 'dinner'")	15004
2	'dinner'	7359
3	'lunch'	8289
4	c(" 'dinner'", " 'breakfast'", " 'brunch'")	138
5	character(0)	2174
6	c(" 'breakfast'", " 'brunch'")	1266
7	c(" 'lunch'", " 'dinner'", " 'breakfast'")	750
8	'latenight'	605
9	'breakfast'	928
10	c(" ('dessert'", " 'lunch'", " 'dinner'")	706
11	c(" 'dinner'", " 'brunch'")	701
12	c(" 'lunch'", " 'breakfast'", " 'brunch'")	1516
13	c(" 'lunch'", " 'dinner'", " 'brunch'")	99
14	c(" 'lunch'", " 'breakfast'")	486
15	c(" ('dessert'", " 'breakfast'")	283
16	c(" 'latenight'", " 'lunch'", " 'dinner'", " 'brea [...]"	126
17	c(" 'latenight'", " 'breakfast'", " 'brunch'")	66
18	c(" 'latenight'", " 'lunch'")	138
19	('dessert'	731
20	c(" 'latenight'", " 'lunch'", " 'dinner'")	1080
21	'brunch'	662
22	c(" ('dessert'", " 'brunch'")	44
23	c(" ('dessert'", " 'lunch'")	228
24	c(" ('dessert'", " 'latenight'", " 'breakfast'", " [...]"	121
25	c(" 'latenight'", " 'dinner'")	668
26	c(" 'latenight'", " 'lunch'", " 'breakfast'", " 'b [...]"	38
27	c(" 'lunch'", " 'brunch'")	261
28	c(" ('dessert'", " 'dinner'")	117
29	c(" 'latenight'", " 'lunch'", " 'breakfast'")	132
30	c(" 'latenight'", " 'breakfast'")	30
31	c(" ('dessert'", " 'lunch'", " 'breakfast'")	109
32	c(" 'lunch'", " 'dinner'", " 'breakfast'", " 'brun [...]"	29

Figure i.52

Yes, as shown in Figures i.50 to i.52, the star ratings vary depending on these factors.

(b) For one of your models (choose your 'best' model from above), does prediction accuracy vary by certain restaurant attributes? You do not need to look into all attributes; choose a few which you think may be interesting, and examine these.

We have chosen the Random Forest Model for best accuracy compared to the other models. We can observe that prediction accuracy varies by certain attributes.