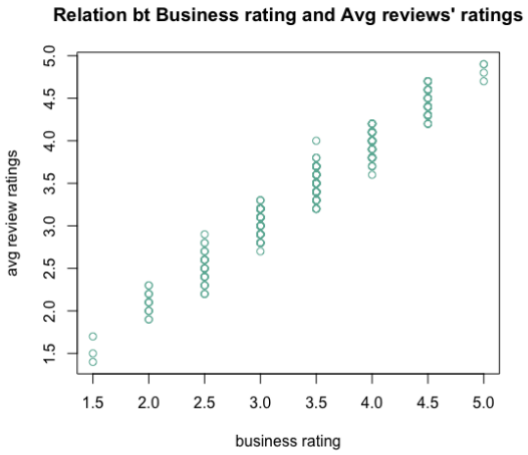
HW 3: Yelp Reviews – Text Mining

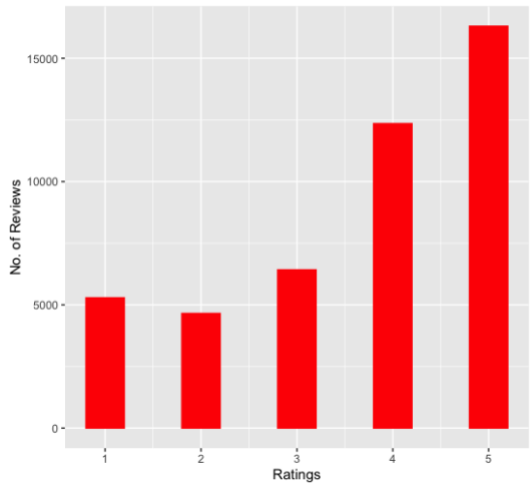
1. **Explore the data.**

   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?**

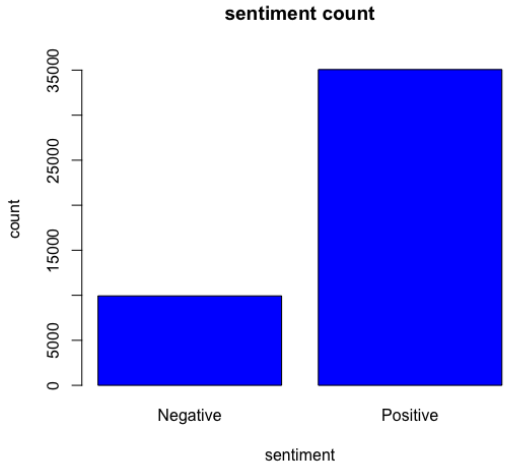


- For each business rating starting from 1.5 to 5.0 with 0.5 increment, a positive pattern can be observed. The two attributes are directly proportional
- For each business rating, the average review ratings are distributed with the mean same as business rating. E.g. for 2.5 business rating, the avg review ratings are also around 2.5
- If the business rating is given, we can guess the average review rating.

   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.?**



- More number of ratings are 4 and 5
- Considered, ratings 3, 4 and 5 as 'Positive' and 1, 2 as 'Negative' reviews.
- Business improvement can be done based on 1 and 2 ratings' reviews.
- After categorizing as above, Negative reviews are 9933 and Positive reviews are 35067.



2. What are some words in the restaurant reviews indicative of positive and negative sentiment – identify at least 20 in each category.

Our approach to determine words associated with the positive and negative sentiments:

- After tokenizing the reviews, filtered the sentiment column with 'Positive' and 'Negative' separately.
- So, the word associated with each sentiment is considered as the same sentimental word.
- In the resulted rows, calculated the frequency of each word.
- Sorted in the decreasing order.

	word	count_of_words
14340	food	26385
31404	service	13042
35658	time	10162
10359	delicious	7636
7347	chicken	7485
24099	nice	7391
21266	love	7356
29524	restaurant	7202
22511	menu	7075
14745	friendly	6171
14691	fresh	5625
21386	lunch	5424
30778	sauce	5293
7218	cheese	5180
27674	pretty	5174
3755	bar	5076
2264	amazing	5073
33483	staff	4985
30492	salad	4830
12013	eat	4588
26836	pizza	4289
22288	meal	3998

Positive words

	word	count_of_words
8657	food	10175
18197	service	4690
20834	time	3867
17139	restaurant	2854
13344	minutes	2557
4370	chicken	2386
20244	table	2049
2219	bad	2006
308	2	1830
7245	eat	1811
15137	people	1758
2330	bar	1755
13118	menu	1685
20941	told	1574
17836	sauce	1552
4308	cheese	1535
18192	server	1529
7922	experience	1481
22275	wait	1454
17689	salad	1409
12998	meal	1390
22288	waitress	1388
12708	manager	1338

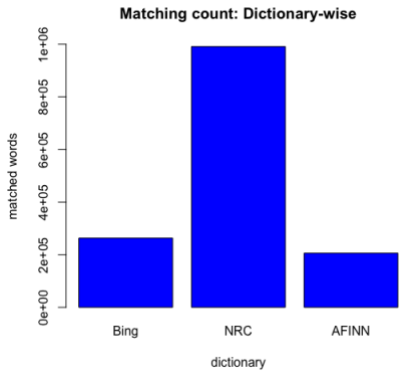
Negative words

3.

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

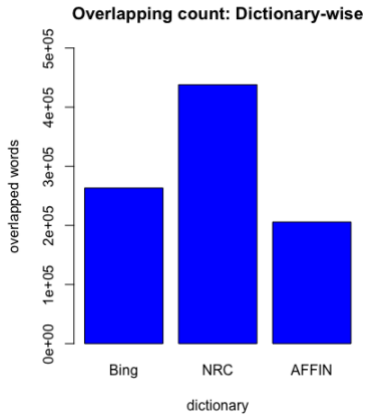
Matched number of terms in each dictionary:

- Bing: 263206
- NRC: 991180
- AFINN: 205826



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 you text of reviews?

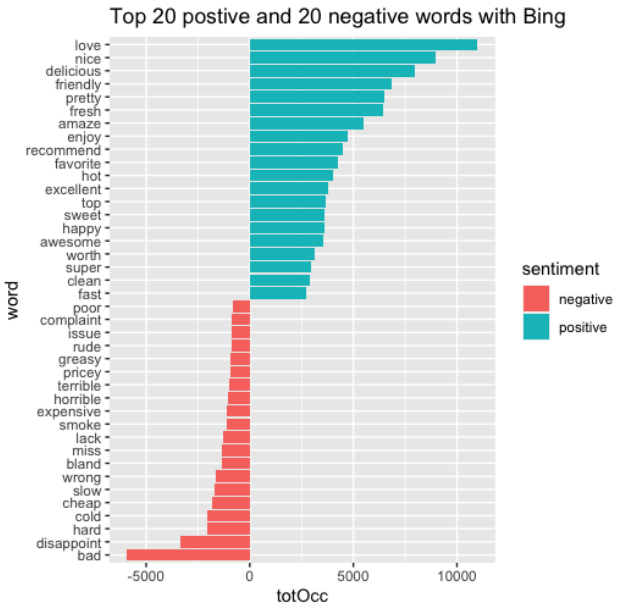
- Bing: 263205
- NRC: 437991
- AFINN: 205826



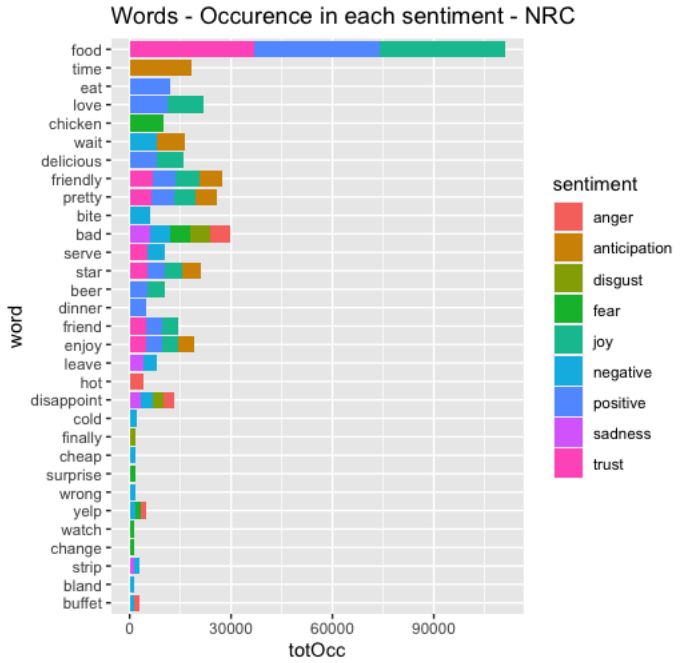
- Based on the above information, **NRC** would be better to pick up the sentiment information.

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?

- After performing inner join to retain only matching words with Bing dictionary, plotted the top words as below.



- Similarly, we counted the occurrence of each word for each sentiment and plotted the graph. However, for NRC, we need to convert these sentiments into positive and negative review. For NRC, first we converted 'anger', 'disgust', 'fear', 'sadness', 'negative', 'positive', 'joy', 'anticipation', 'trust' sentiments into positive and negative categories.



- For AFINN, we used the ‘value’ specific to the word in review. Then calculated the average value of the word in all reviews. If average_value > 0 then positive, else negative for each word.
- Most of the words don’t match with the dictionaries’ words. Some of the words are matched with NRC dictionary such as food, time, chicken, friendly, etc.

4.

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

- First, we calculated average score, by using:
  - the difference of proportion of positive words and negative words for Bing dictionary.
  - the sum of goodBad for each word in NRC Dictionary.
  - the sum of value for each review for AFINN Dictionary.

- After that we removed the reviews with 3 ratings and converted rest of the reviews into positive (>3) and negative (<3) label, which we consider as actual result (hiLo).

- To calculate prediction value, we used aggregated score and labeled positive for score>0 and vice versa for all dictionaries.

- Then we derived confusion matrix using actual and predicted hiLo label.

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

- **Matching terms:**
  - **Bing:**

actual \ predicted	-1	1
-1	7455	2201
1	4065	23873

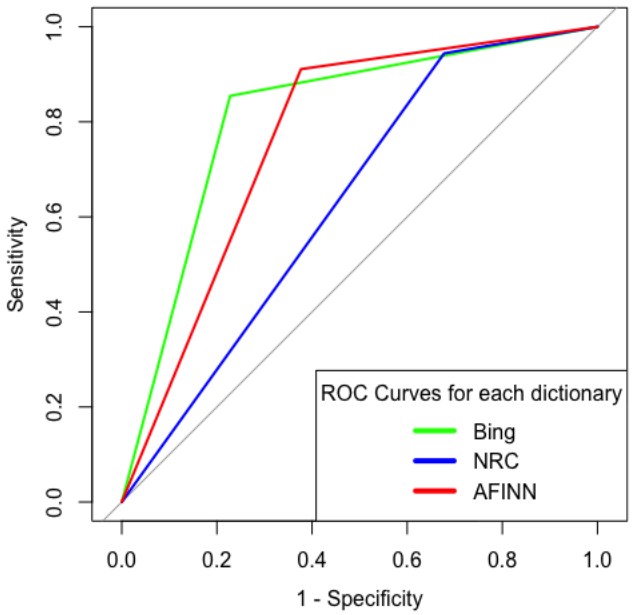
  - **NRC:**

actual \ predicted	-1	1
-1	3187	6701
1	1593	26850

  - **AFINN:**

actual \ predicted	-1	1
-1	5870	3549
1	2438	24913

- **ROC Curves from matchings:**



- **AUC for each dictionary:**
    1. Bing: 0.8133
    2. NRC: 0.6332
    3. AFINN: 0.767
- According to AUCs, **Bing dictionary** is performing better than other two dictionaries.

5. Modelling:

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

- Confusion matrix → ROC → AUC: to know how accurately the model is classifying the words.

b. Which types of models does your team choose to develop, and why?

Models chosen to develop:

- **Random Forest:** Random Forest is suitable for situations when we have a large dataset, and interpretability is not a major concern.
- **Naïve Bias:** Naive Bayes is suitable for solving multi-class prediction problems. If its assumption of the independence of features holds true, it can perform better than other models and requires much less training data.
- **SVM Classification:** SVM uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs.

Do you use term frequency, tf-idf, or other measures, and why?

- Yes, we have used term frequency (tf) and tf-idf to prune the data set and model evaluation.
- The reason we used these two terms is because they give us an idea how the document tends to be positive or negative review.

c. Develop models using only the sentiment dictionary terms – try the three different dictionaries; how do the dictionaries compare in terms of predictive performance?

- **Random Forest:**
  - **Confusion matrix before finding best thresholds:**

```
> table(actual=revDTM_sentiBing_trn$hiLo, preds=revSentiBing_predTrn[,2]>0.5)
      preds
actual FALSE  TRUE
    -1   505  4400
    1      0 13892
> table(actual=revDTM_sentiBing_tst$hiLo, preds=revSentiBing_predTst[,2]>0.5)
      preds
actual FALSE  TRUE
    -1   414  4337
    1      5 14041
> table(actual=revDTM_sentiNRC_trn$hiLo, preds=revSentiNRC_predTrn[,2]>0.5)
      preds
actual FALSE  TRUE
    -1   508  4453
    1      0 14204
> table(actual=revDTM_sentiNRC_tst$hiLo, preds=revSentiNRC_predTst[,2]>0.5)
      preds
actual FALSE  TRUE
    -1   359  4568
    1      4 14235
> table(actual=revDTM_sentiAFINN_trn$hiLo, preds=revSentiAFINN_predTrn[,2]>0.5)
      preds
actual FALSE  TRUE
    -1   877  3865
    1      9 13634
> table(actual=revDTM_sentiAFINN_tst$hiLo, preds=revSentiAFINN_predTst[,2]>0.5)
      preds
actual FALSE  TRUE
    -1   788  3889
    1     17 13691
```

  - **Best threshold:**
    - Bing: 0.7510758
    - NRC: 0.7269498
    - AFINN: 0.7525317

  - **Confusion matrix after finding best thresholds:**

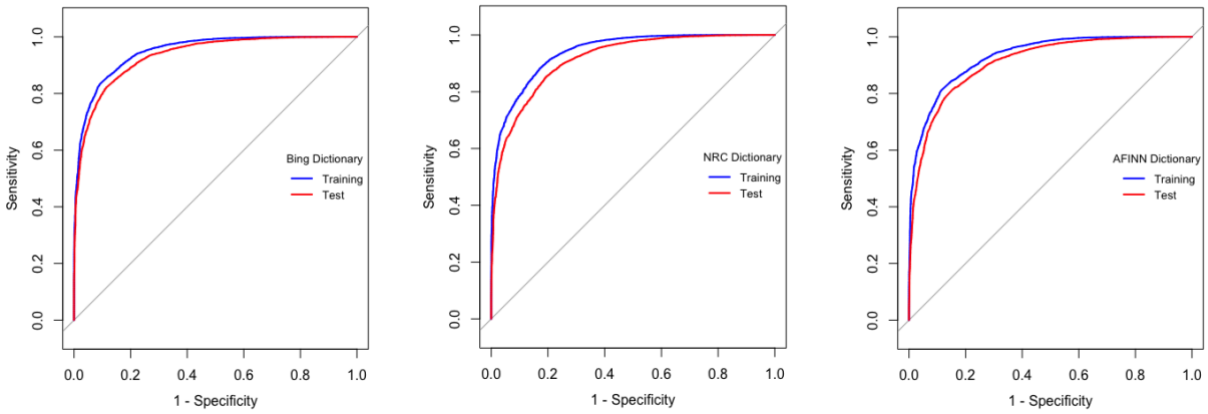
```
> table(actual=revDTM_sentiBing_trn$hiLo, preds=revSentiBing_predTrn[,2]>bThrBing[1,1])
      preds
actual FALSE  TRUE
    -1  4470   435
    1   2254 11638
> table(actual=revDTM_sentiBing_tst$hiLo, preds=revSentiBing_predTst[,2]>bThrBing[1,1])
      preds
actual FALSE  TRUE
    -1  4242   509
    1   2688 11358
> #NRC
> table(actual=revDTM_sentiNRC_trn$hiLo, preds=revSentiNRC_predTrn[,2]>bThrNRC[1,1])
      preds
actual FALSE  TRUE
    -1  4110   851
    1   1567 12637
> table(actual=revDTM_sentiNRC_tst$hiLo, preds=revSentiNRC_predTst[,2]>bThrNRC[1,1])
      preds
actual FALSE  TRUE
    -1  3900  1027
    1   1906 12333
> #AFINN
> table(actual=revDTM_sentiAFINN_trn$hiLo, preds=revSentiAFINN_predTrn[,2]>bThrAFINN[1,1])
      preds
actual FALSE  TRUE
    -1  4212   530
    1   2649 10994
> table(actual=revDTM_sentiAFINN_tst$hiLo, preds=revSentiAFINN_predTst[,2]>bThrAFINN[1,1])
      preds
actual FALSE  TRUE
    -1  4049   628
    1   2792 10916
```

  - **AUC:**

Dictionary	AUC_training	AUC_testing
Bing	0.9484	0.9328
NRC	0.9402	0.9142

AFINN	0.9304	0.9094

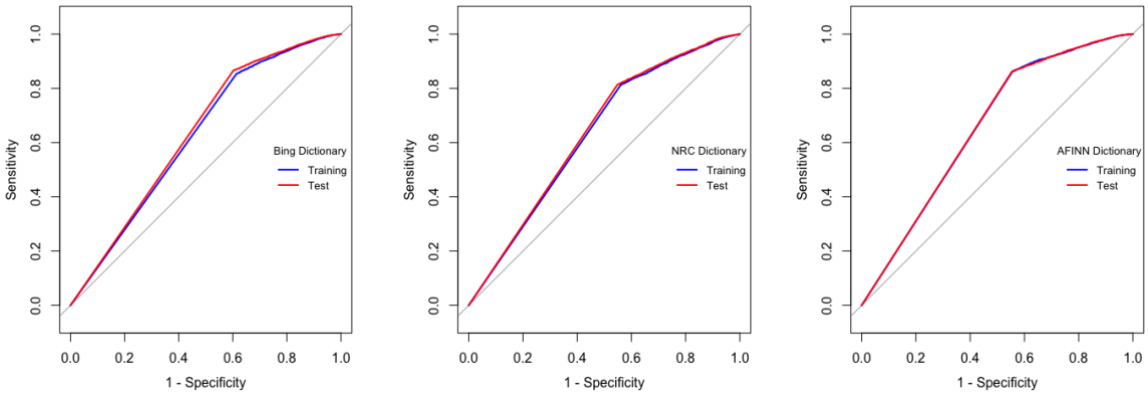
- **ROC Curves:**



- **Naïve Bias:**
  - **Confusion matrix:**

```
> #Bing
> table(actual=revDTM_sentiBing_trn$hiLo, preds=revSentiBing_NBpredTrn[,2]>0.5)
      preds
actual FALSE  TRUE
    -1  1579  3326
    1   1465 12427
> table(actual=revDTM_sentiBing_tst$hiLo, preds=revSentiBing_NBpredTst[,2]>0.5)
      preds
actual FALSE  TRUE
    -1  1494  3257
    1   1448 12598
> #NRC
> table(actual=revDTM_sentiNRC_trn$hiLo, preds=revSentiNRC_NBpredTrn[,2]>0.5)
      preds
actual FALSE  TRUE
    -1  2153  2808
    1   2611 11593
> table(actual=revDTM_sentiNRC_tst$hiLo, preds=revSentiNRC_NBpredTst[,2]>0.5)
      preds
actual FALSE  TRUE
    -1  2064  2863
    1   2611 11628
> #AFINN
> table(actual=revDTM_sentiAFINN_trn$hiLo, preds=revSentiAFINN_NBpredTrn[,2]>0.5)
      preds
actual FALSE  TRUE
    -1  1808  2934
    1   1534 12109
> table(actual=revDTM_sentiAFINN_tst$hiLo, preds=revSentiAFINN_NBpredTst[,2]>0.5)
      preds
actual FALSE  TRUE
    -1  1747  2930
    1   1445 12263
```

  - **ROC Curves:**



  - **AUC:**

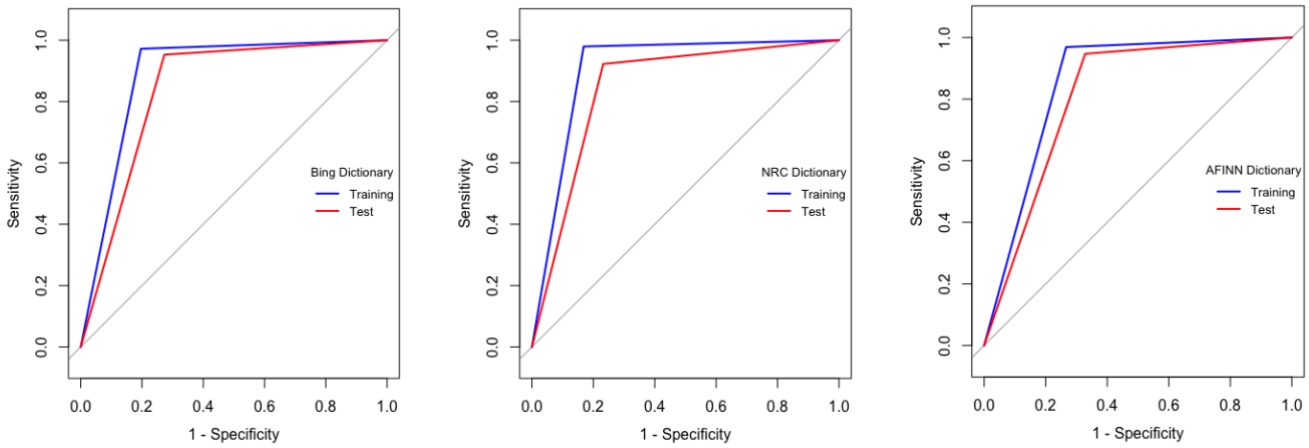
Dictionary	AUC_training	AUC_testing

Bing	0.6239	0.6352
NRC	0.6297	0.6371
AFINN	0.6573	0.6581

- **SVM Classification:**
  - **Confusion matrices:**

```
> #Bing
> table(actual= revDTM_sentiBing_trn_2$hiLo, predicted= revDTM_predTrn_svmBing)
      predicted
actual   -1    1
    -1 2088  487
    1   219 7206
> table(actual= revDTM_sentiBing_tst_2$hiLo, predicted= revDTM_predTst_svmBing)
      predicted
actual   -1    1
    -1 1845  672
    1   350 7133
> #NRC
> table(actual= revDTM_sentiNRC_trn_2$hiLo, predicted= revDTM_predTrn_svmNRC)
      predicted
actual   -1    1
    -1 2114  449
    1   160 7277
> table(actual= revDTM_sentiNRC_tst_2$hiLo, predicted= revDTM_predTst_svmNRC)
      predicted
actual   -1    1
    -1 2793  898
    1   446 5863
> #AFINN
> table(actual= revDTM_sentiAFINN_trn_2$hiLo, predicted= revDTM_predTrn_svmAFINN)
      predicted
actual   -1    1
    -1 1889  678
    1   239 7194
> table(actual= revDTM_sentiAFINN_tst_2$hiLo, predicted= revDTM_predTst_svmAFINN)
      predicted
actual   -1    1
    -1 1714  832
    1   375 7079
```

  - **ROC Curves:**



  - **AUC:**

Dictionary	AUC_training	AUC_testing
Bing	0.8877	0.8398
NRC	0.9055	0.8453
AFINN	0.8511	0.8094

Then with a combination of the three dictionaries, i.e., combine all dictionary terms. What is the size of the document-term matrix? Should you use stemming or lemmatization when using the dictionaries? Why?

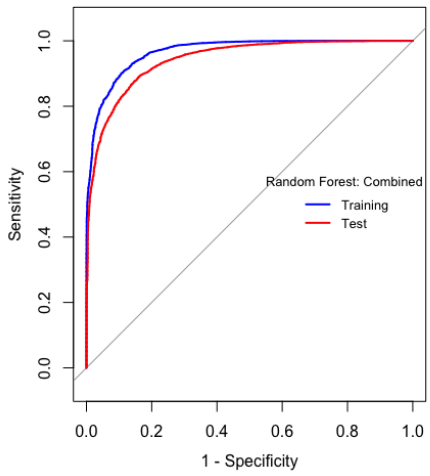
- Stemming is a good idea as it provides less attributes to train, but we did not use stemming here.
- However, using stemming would have resulted in fewer features than what we had previously, and it could have decreased the time to fit the model.
- Size of doc-term matrix: 36421 x 5472

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?

- By 'merge' function, we can combine different lists.
- Here, we won't separately stem or lemmatize because we have already done it in previous steps.
- **Models:**
  - **Random Forest:**
    - Best threshold: 0.732927
    - Confusion matrix:

```
> #Confusion Matrix
> table(actual=revDTM_se
      preds
actual FALSE TRUE
    -1  2295  276
    1    735 6694
> table(actual=revDTM_se
      preds
actual FALSE TRUE
    -1  2198  366
    1    967 6469
```

    - ROC:



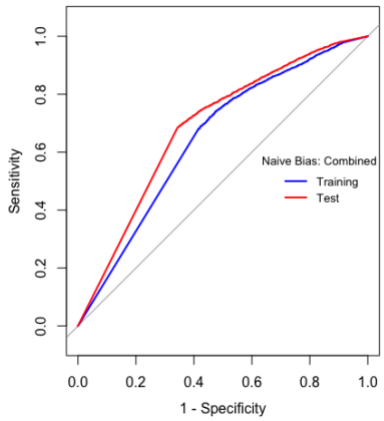
    - AUC for training: 0.9703
    - AUC for testing: 0.9427

  - **Naïve Bias:**

- Confusion matrix:

```
> #Confusion Matrix
> table(actual=revDTM_
      preds
actual FALSE TRUE
    -1  1406 1165
    1   2101 5328
> table(actual=revDTM_
      preds
actual FALSE TRUE
    -1  1544 1020
    1   2039 5397
```

- ROC:

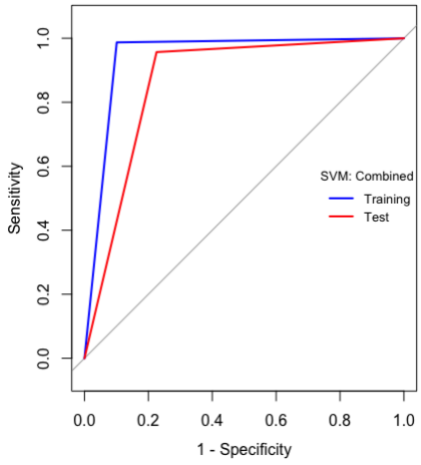


- AUC: 0.6489 for training and 0.6874 for testing

- **SVM:**
  - Confusion matrix:

```
> #Confusion Matrix
> table(actual= revDT
      predicted
actual   -1    1
    -1 2311  260
    1    96 7333
> table(actual= revDT
      predicted
actual   -1    1
    -1 1985  579
    1   321 7115
```

  - ROC:



- AUC: 0.943 for training and 0.8655 for testing set.

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).

Random forest – compare AUCs with separate dictionaries:

Dictionary	Training	Testing
Bing	0.9484	0.9328
NRC	0.9402	0.9142
AFINN	0.9304	0.9094
COMBINED	0.9703	0.9427

Naïve Bias – compare:

Dictionary	Training	Testing
Bing	0.6239	0.6352
NRC	0.6297	0.6371
AFINN	0.6573	0.6581
COMBINED	0.6489	0.6874

SVM – compare:

Dictionary	Training	Testing
Bing	0.8877	0.8398
NRC	0.9055	0.8453
AFINN	0.8511	0.8094
COMBINED	0.943	0.8655

Clearly, every model is performing much better with the combined dictionaries rather than with each dictionary alone. Because different words might be matched in different dictionaries. But with the combined dictionaries, all matched words are under one roof. So, more accuracy.

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.

- **Selected attributes:**
  1. **Ambiance:**

	amb	n
1	'casual'	32306
2	character(0)	5092
3	'trendy'	1775
4	'divey'	1421
5	'classy'	757
6	'intimate'	499
7	c(" 'classy'", " 'upscale'")	490
8	'touristy'	420
9	{'romantic'	300
10	c(" 'classy'", " 'trendy'", " 'upscale'")	283
11	c(" 'hipster'", " 'trendy'")	237
12	'upscale'	223
13	c(" 'trendy'", " 'casual'")	159
14	c(" 'touristy'", " 'casual'")	158
15	c(" 'divey'", " 'casual'")	136

2. **Parking:**

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

3. **Good for meal:**

	GdFrMl	n
1	c(" 'lunch'", " 'dinner'")	15004
2	'lunch'	8289
3	'dinner'	7359
4	character(0)	2174
5	c(" 'lunch'", " 'breakfast'", " 'brunch'")	1516
6	c(" 'breakfast'", " 'brunch'")	1266
7	c(" 'latenight'", " 'lunch'", " 'dinner'")	1080
8	'breakfast'	928
9	c(" 'lunch'", " 'dinner'", " 'breakfast'")	750
10	{'dessert'	731
11	c(" {'dessert'", " 'lunch'", " 'dinner'")	706
12	c(" 'dinner'", " 'brunch'")	701
13	c(" 'latenight'", " 'dinner'")	668
14	'brunch'	662
15	'latenight'	605
16	c(" 'lunch'", " 'breakfast'")	486
17	c(" {'dessert'", " 'breakfast'")	283
18	c(" 'lunch'", " 'brunch'")	261
19	c(" {'dessert'", " 'lunch'")	228
20	c(" 'dinner'", " 'breakfast'", " 'brunch'")	138
21	c(" 'latenight'", " 'lunch'")	138
22	c(" 'latenight'", " 'lunch'", " 'breakfast'")	132
23	c(" 'latenight'", " 'lunch'", " 'dinner'", " 'breakfast'")	126
24	c(" {'dessert'", " 'latenight'", " 'breakfast'", " 'brunch'")	121
25	c(" {'dessert'", " 'dinner'")	117

- Yes, star ratings varied with respect to these three attributes selected.

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 investigate all attributes; choose a few which you think may be interesting and examine these.

We have chosen the **Random Forest Model** as it has given us the best accuracy compared to the other models. And looking at all the attributes, we can observe thar prediction accuracy does vary by certain restaurant attributes.