IDS 572 ASSIGNMENT 5

Aditi Vishwasrao – 674849041

Mounica Sirineni - 652518967

Nishanth Reddy Konkala - 658824939

1. Read a few of the movie-reviews. If you were to classify manually, are there certain words/terms indicative of 'positive' and 'negative' that you would use? Provide lists ('dictionaries') of 'positive' and 'negative' terms. Now consider a manual classification process – you may count the number of 'positive' and 'negative' terms (matching your dictionaries) and classify based on majority count. How effective is this (accuracy?)?

Answer: We went through a few movie reviews and prepared a list of terms that indicate a positive or negative review. The complete list of terms are as follows:

positive.txt

negative.txt

A few of the positive reviews are: strong, extraordinary, interesting, good, special, strong, likeable,etc.

A few of the negative reviews are: aweful, horrific, unfunny, worst, mess, boring, slow, lame, etc.

The list of files that were considered for manual prediction is as follows:

manual_prediction.xlsx

We get the following confusion matrix from the results:

Accuracy: 55%

	Predicted Positive	Predicted Negative	Precision
Actual Positive	15	5	75%
Actual Negative	13	7	35%
Recall	70%	30%	

Conclusion: In this method of manual prediction, the accuracy is 55%. This is not very good accuracy for a model. Also, the number of negative cases that have been detected accurately are very less. We also need to take into consideration that we have taken into account a very few (20 each) positive and negative reviews for classification. We can conclude that the manual process is not very effective.

2. Using automated text-mining: perform the steps noted above to get a document-term matrix. How many terms do you have? With and without stemming?

Answer: We performed automated text mining using tokenization, filtering stop words, transforming cases and filtering tokens by length. We evaluated for stemming. The table provides a summary. Click here to view the screenshots.

Table 2.1 Automated text mining

Method	Number of Examples	Number of Special Attributes	Number of Regular Attributes
Without Stemming	2000	6	2425
With Stemming	2000	6	1040

Conclusion: From the table, we observe that stemming decreases the number of regular attributes. From the definition of stemming, we say that stemming reduces the words to its root words.

3. The sentiment polarity of a document provides the 'label'/target variables. Learn knn and naïve-Bayes models to classify for sentiment polarity. To evaluate effectiveness in prediction, we will consider Split-Validation, with 50% of the documents used for training and 50% for test (you can set a local random number seed for the Split-Validation operator to get the same split on the data across experiments).

(a) Develop and evaluate knn and naïve-Bayes models. What values of k work best? Evaluation is based on the confusion matrix – overall accuracy, accuracies on positive and negative, precision, specificity, sensitivity, etc.

Answer: We have developed KNN and Naïve Bayes models. Below table provides a summary of performance. Click here to view the screenshots of performance vector.

Table 3.1 KNN and Naïve Bayes models

Model	Dataset	Overall accuracy	Class recall (true negative)	Class recall (true positive)
KNN with K=3	Training data	80.80%	79.43%	82.12%
	Testing data	67%	61.10%	73.12%
KNN with K=4	Training data	77.40%	88.80%	66.40%
	Testing data	65.80%	73.48%	57.84%
KNN with K=5	Training data	78.10%	75.15%	80.94%
	Testing data	64.7%	57.56%	72.1%
KNN with K=10	Training data	72.30%	78%	66.80%
	Testing data	66.40%	70.92%	61.71%
KNN with K=30	Training data	73.30%	79.63%	67.19%
	Testing data	67.40%	72.30%	62.32%
Naïve Bayes	Training data	100%	100%	100%
	Testing data	64.90%	59.53%	70.47%

Conclusion: From the above table, we observe that **KNN with K = 4** is the best model. Further analysis is done using this model.

You should experiment with the following to determine how they affects performance:
(i) measures defining the document term matrix – binary term occurrence, term occurrence, term frequency, tf-idf (please define these four term as part of the assignment)

Answer:KNN with K = 4 model is evaluated using different measures. The table below provides a summary of performances. Click here to view the screenshots of performance vector.

Binary term occurrence: If a word occurs in the document, then the vector is “1” otherwise “0”.

Term occurrence: It gives the number of times the term occurs in the document.

Term frequency: It gives the frequency of the term in the document.

TF-IDF: (Term Frequency – Inverse Document Frequency) It indicates the frequency of a term in the document and in the corpus of documents.

Table 3.2 KNN with K = 4 and different measures

Document-term matrix	Training accuracy	Testing accuracy
Binary term occurrence	95.10%	51.80%
Term occurrence	71.3%	52.6%
Term frequency	74.20%	63.80%
TF-IDF	77.40%	65.80%

Conclusion: From the above table, we observe that KNN with K=4 and TF-IDF measure is the best model.

(ii) which words to include
- without and with stemming

Answer: KNN model with K = 4 and TF-IDf measure is evaluated on stemming. Click here to view the screenshots of performance vector.

Table 3.3 KNN with K = 4, TF-IDf, with and without stemming

Document-term matrix	Training accuracy	Testing accuracy
With stemming	76.80%	67%
Without stemming	77.40%	65.80%

Conclusion: From the above table, we observe that KNN model with K=4, TF-IDF measure and with stemming is the best model.

- using words that match specific POS tags – you need to determine which POS tags you want to include (there are multiple websites that give the list of POS tags)

Answer: KNN model with K = 4 , TF-IDF measure and stemming is evaluated for POS tags. We have used two different combinations of POS tags. Click here to view the screenshots of performance vector.

Table 3.4 KNN with K = 4, TF-IDf, with stemming and POS tags

POS tags	Training accuracy	Testing accuracy
JJ.*|NN.*|RB.*|VB.*	76.40%	66.60%
VB.*|NN.*|RB.*	76.90%	63.70%

From the above table, we observe that the best POS tags are combination of adjective, noun, adverb and verb. Now we will evaluate whether this best POS tags are helping the model or not in terms of accuracy.

Table 3.5 KNN with K = 4, TF-IDf, with stemming, with and without POS tags

POS tags	Training accuracy	Testing accuracy
With POS tags	76.40%	66.60%
Without POS tags	76.80%	67%

From the above table, we observe that model has better accuracy without using any POS tags.

- pruning words based on min and max of the number of documents they appear in

Answer: KNN model with K = 4, TF-IDF measure, stemming and without POS tags is evaluated for pruning. We have used min of 10 and max of 100 documents for pruning. Click here to view the screenshots of performance vector.

Table 3.6 KNN with K = 4, TF-IDf, with stemming, without POS tags, with and without pruning

Pruning	Training accuracy	Testing accuracy
With pruning	75.90%	64.90%
Without pruning	76.80%	67%

From the above table, we observe that **KNN withK = 4, TF-IDf, with stemming, without POS tags and without pruning** is the best model.

(b) For any one setting from above (for example, using tf-idf, stemmed terms, and using some reasonable pruning), build a support vector machine model. Try SVM with dot-kernel. How does the performance of the SVM model compare with k-nn and naïve-Bayes models? Does using a radial basis

function kernel (nonlinear model) perform better? Also try a random forest model and compare performance. What do you conclude overall?

Answer: We have used our best model setting from previous question. TF-IDF measure with stemming, without POS tags and without pruning. We developed SVM. Random forest, KNN with K = 4 and Naïve Bayes for this model. Click here to view the screenshots of performance vectors.

Table 3.7 Different models

Model	Training accuracy	Testing accuracy
SVM with dot-kernel	100%	77.50%
SVM with radial-kernel	50.90%	49.10%
Random Forest	52.30%	49.60%
KNN with K = 4	76.80%	67%
Naïve Bayes	99.90%	64.90%

Conclusion: From the above table, we observe that SVM with dot-kernel performs better than radial kernel. When we compare all models, KNN with K = 4 is the best model.

(c) Use the Harvard-IV dictionary of positive and negative terms - Filter Tokens by Content to keep only these terms. How many terms do you get? (Note: for this, the document terms should not be stemmed, since they would otherwise not match the unstemmed dictionary terms. Does it make sense to use pruning of terms or POS tags here?). Use just the count of positive and negative terms to classify sentiment of reviews. How do you do this, and what performance do you find? Develop and evaluate a naïve-Bayes, a SVM (either with dot kernel or other kernel based on what you found to perform better in part (b) above), and a random forest model, using only the positive and negative words.

Answer:
By filtering tokens by content of Harvard-IV dictionary of positive and negative terms, we get 2885 terms. Pruning or using POS tags may result in different token from the review documents which will not match the Harvard IV dictionary terms. Thus, pruning or using POS tags does not make sense.

Click here to view the screenshots of performance vector.

Table 3.8 Harvard dictionary with different models

Model	Training accuracy	Testing accuracy
Naïve Bayes	93.20%	57.60%
SVM with dot-kernel	97.20%	67.10%
Random Forest	55.90%	53.40%
KNN with K = 4	75.90%	61.00%

We used only the SVM with dot kernel model here as this was the better of the two models used for part (b). From the above table, we can see that KNN with K=4 is clearly the best model.

(d) Next, use Wordnet and Senti-Wordnet to determine the sentiment polarity of the reviews. Provide a description of how this process works – i.e. how WordNet and Senti-Wordnet operates. Use the Senti-WordNet sentiment scores to classify the reviews. What performance do you find?

Answer:

"Word net is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Sysnets are interlinked by means of conceptual semantic and lexical relations" ~ **Princeton University WordNet**.

Whereas Senti-WordNet is a lexical resource for opinion mining. It assigns each synset of Word Net three sentiment scores:Positivity, negativity, and objectivity.

Click here to view the screenshots of performance vector.

Table 3.9 Wordnet dictionary with different models

Model	**Training accuracy**	**Testing accuracy**
Naïve Bayes	100%	63%
SVM with dot-kernel	100%	75.60%
Random Forest	50.90%	49.10%
KNN with K = 4	78%	64.90%

Conclusion: KNN with K = 4 is the best model.

(e) How does performance in (a) - (d) above compare? What can you conclude? Does the smaller list of 'sentiment' words provide similar performance levels as models built on a broader set of terms? How does the performance of Senti-WordNet compare? Does it help to include Senti-WordNet scores in a model with sentiment-words or with the broader set of terms?Which is your 'best' model?

KNN with k=4 yields the best model in all cases from (a) to (d). Comparing the training and validation accuracies in all four cases, we do not find any significant differences in the performance of the KNN model in each of the four cases.

However, considering only the validation accuracy, the smaller list of sentiment words looks to be giving a slightly better performance.

The inclusion of Senti-Wordnet scores does not seem to help to a great extent.

Our best model is KNN with k=4 with stemming, but without using POS tags or pruning.

4. Appendix:

4.1 Automated text mining [Back to the top]

4.1.1 Without stemming

ExampleSet (2000 examples, 6 special attributes, 2425 regular attributes) Filter (2,000 / 2,000 examples): all

Row No.	label	metadata_...	metadata_...	metadata_...	NEG_token...	POS_token...	abandon	abandonm...	ability	abject	able
1	neg	cv000_294	/Users/Nish	Jan 23, 201	15	8	0	0	0	0	0
2	neg	cv001_195	/Users/Nish	Jan 23, 201	2	1	0	0	0	0	0
3	neg	cv002_174	/Users/Nish	Jan 23, 201	9	6	0	0	0	0	0
4	neg	cv003_126	/Users/Nish	Jan 23, 201	11	8	0	0	0	0	0.111
5	neg	cv004_126	/Users/Nish	Jan 23, 201	13	10	0	0	0	0	0
6	neg	cv005_293	/Users/Nish	Jan 23, 201	20	6	0	0	0	0	0
7	neg	cv006_170	/Users/Nish	Jan 23, 201	12	8	0	0	0	0	0
8	neg	cv007_499	/Users/Nish	Jan 23, 201	11	12	0	0	0	0	0
9	neg	cv008_293	/Users/Nish	Jan 23, 201	21	16	0	0	0	0	0.070
10	neg	cv009_294	/Users/Nish	Jan 23, 201	10	10	0	0	0	0	0
11	neg	cv010_290	/Users/Nish	Jan 23, 201	17	9	0	0	0	0	0
12	neg	cv011_130	/Users/Nish	Jan 23, 201	6	14	0	0	0	0	0
13	neg	cv012_294	/Users/Nish	Jan 23, 201	12	1	0	0	0	0	0
14	neg	cv013_104	/Users/Nish	Jan 23, 201	21	10	0	0	0	0	0
15	neg	cv014_156	/Users/Nish	Jan 23, 201	12	3	0	0	0	0	0
16	neg	cv015_293	/Users/Nish	Jan 23, 201	17	8	0	0	0	0	0
17	neg	cv016_434	/Users/Nish	Jan 23, 201	9	8	0	0	0	0	0
18	neg	cv017_234	/Users/Nish	Jan 23, 201	13	11	0	0	0	0	0.098
19	neg	cv018_216	/Users/Nish	Jan 23, 201	5	8	0	0	0	0	0
20	neg	cv019_161	/Users/Nish	Jan 23, 201	10	18	0	0	0	0	0
21	neg	cv020_923	/Users/Nish	Jan 23, 201	24	14	0	0	0	0	0
22	neg	cv021_173	/Users/Nish	Jan 23, 201	13	11	0	0	0	0	0
23	neg	cv022_142	/Users/Nish	Jan 23, 201	16	5	0	0	0	0	0
24	neg	cv023_138	/Users/Nish	Jan 23, 201	16	14	0	0	0.114	0	0.079
25	neg	cv024_703	/Users/Nish	Jan 23, 201	15	24	0	0	0	0	0

4.1.2 With stemming

ExampleSet (2000 examples, 6 special attributes, 1040 regular attributes) Filter (2,000 / 2,000 examples): all

Row No.	label	metadata_...	metadata_...	metadata_...	NEG_token...	POS_token...	abandon	abject	abolish	abound	abrupt
1	neg	cv000_294	/Users/Nish	Jan 23, 201	11	8	0	0	0	0	0
2	neg	cv001_195	/Users/Nish	Jan 23, 201	3	0	0	0	0	0	0
3	neg	cv002_174	/Users/Nish	Jan 23, 201	6	4	0	0	0	0	0
4	neg	cv003_126	/Users/Nish	Jan 23, 201	10	4	0	0	0	0	0
5	neg	cv004_126	/Users/Nish	Jan 23, 201	7	4	0	0	0	0	0
6	neg	cv005_293	/Users/Nish	Jan 23, 201	12	7	0	0	0	0	0
7	neg	cv006_170	/Users/Nish	Jan 23, 201	7	6	0	0	0	0	0
8	neg	cv007_499	/Users/Nish	Jan 23, 201	12	7	0	0	0	0	0
9	neg	cv008_293	/Users/Nish	Jan 23, 201	23	6	0	0	0	0	0
10	neg	cv009_294	/Users/Nish	Jan 23, 201	11	6	0	0	0	0	0
11	neg	cv010_290	/Users/Nish	Jan 23, 201	17	9	0	0	0	0	0
12	neg	cv011_130	/Users/Nish	Jan 23, 201	5	10	0	0	0	0	0
13	neg	cv012_294	/Users/Nish	Jan 23, 201	11	2	0	0	0	0	0
14	neg	cv013_104	/Users/Nish	Jan 23, 201	14	10	0	0	0	0	0
15	neg	cv014_156	/Users/Nish	Jan 23, 201	10	1	0	0	0	0	0
16	neg	cv015_293	/Users/Nish	Jan 23, 201	18	8	0	0	0	0	0
17	neg	cv016_434	/Users/Nish	Jan 23, 201	10	5	0	0	0	0	0
18	neg	cv017_234	/Users/Nish	Jan 23, 201	12	8	0	0	0	0	0
19	neg	cv018_216	/Users/Nish	Jan 23, 201	3	7	0	0	0	0	0
20	neg	cv019_161	/Users/Nish	Jan 23, 201	8	10	0	0	0	0	0
21	neg	cv020_923	/Users/Nish	Jan 23, 201	20	10	0	0	0	0	0
22	neg	cv021_173	/Users/Nish	Jan 23, 201	14	8	0	0	0	0	0
23	neg	cv022_142	/Users/Nish	Jan 23, 201	14	10	0	0	0	0	0
24	neg	cv023_138	/Users/Nish	Jan 23, 201	19	8	0	0	0	0	0

4.2 KNN and Naïve Bayes models

[Back to the top]

4.2.1 K = 3

Training data

accuracy: 80.80%

	true neg	true pos	class precision
pred. neg	390	91	81.08%
pred. pos	101	418	80.54%
class recall	79.43%	82.12%	

Testing data

accuracy: 67.00%

	true neg	true pos	class precision
pred. neg	311	132	70.20%
pred. pos	198	359	64.45%
class recall	61.10%	73.12%	

4.2.2 K = 4

Training data

accuracy: 77.40%			
	true neg	true pos	class precision
pred. neg	436	171	71.83%
pred. pos	55	338	86.01%
class recall	88.80%	66.40%	

Testing data

accuracy: 65.80%			
	true neg	true pos	class precision
pred. neg	374	207	64.37%
pred. pos	135	284	67.78%
class recall	73.48%	57.84%	

4.2.3 K = 5

Training data:

accuracy: 78.10%			
	true neg	true pos	class precision
pred. neg	369	97	79.18%
pred. pos	122	412	77.15%
class recall	75.15%	80.94%	

Testing data:

accuracy: 64.70%			
	true neg	true pos	class precision
pred. neg	293	137	68.14%
pred. pos	216	354	62.11%
class recall	57.56%	72.10%	

4.2.4 K = 10

Training data

accuracy: 72.30%			
	true neg	true pos	class precision
pred. neg	383	169	69.38%
pred. pos	108	340	75.89%
class recall	78.00%	66.80%	

Testing data

accuracy: 66.40%			
	true neg	true pos	class precision
pred. neg	361	188	65.76%
pred. pos	148	303	67.18%
class recall	70.92%	61.71%	

4.2.5 K = 30

Training data

Table View Plot View

accuracy: 73.30%			
	true neg	true pos	class precision
pred. neg	391	167	70.07%
pred. pos	100	342	77.38%
class recall	79.63%	67.19%	

Testing data

Table View Plot View

accuracy: 67.40%			
	true neg	true pos	class precision
pred. neg	368	185	66.55%
pred. pos	141	306	68.46%
class recall	72.30%	62.32%	

4.2.6 Naïve Bayes

Training data

accuracy: 100.00%			
	true neg	true pos	class precision
pred. neg	491	0	100.00%
pred. pos	0	509	100.00%
class recall	100.00%	100.00%	

Testing data

accuracy: 64.90%			
	true neg	true pos	class precision
pred. neg	303	145	67.63%
pred. pos	206	346	62.68%
class recall	59.53%	70.47%	

4.3 Measures defining the document term matrix [Back to the top]

4.3.1 Binary term occurrence

Training data

accuracy: 95.10%			
	true neg	true pos	class precision
pred. neg	487	45	91.54%
pred. pos	4	464	99.15%
class recall	99.19%	91.16%	

Testing data

accuracy: 51.80%			
	true neg	true pos	class precision
pred. neg	115	88	56.65%
pred. pos	394	403	50.56%
class recall	22.59%	82.08%	

4.3.2 Term occurrence

Training data

accuracy: 71.30%			
	true neg	true pos	class precision
pred. neg	477	273	63.60%
pred. pos	14	236	94.40%
class recall	97.15%	46.37%	

Testing data

accuracy: 52.60%			
	true neg	true pos	class precision
pred. neg	461	426	51.97%
pred. pos	48	65	57.52%
class recall	90.57%	13.24%	

4.3.3 Term frequency

Training data

accuracy: 74.20%			
	true neg	true pos	class precision
pred. neg	457	224	67.11%
pred. pos	34	285	89.34%
class recall	93.08%	55.99%	

Testing data

accuracy: 63.80%			
	true neg	true pos	class precision
pred. neg	438	291	60.08%
pred. pos	71	200	73.80%
class recall	86.05%	40.73%	

4.3.4 TF-IDF

Training data

accuracy: 77.40%			
	true neg	true pos	class precision
pred. neg	436	171	71.83%
pred. pos	55	338	86.01%
class recall	88.80%	66.40%	

Testing data

accuracy: 65.80%			
	true neg	true pos	class precision
pred. neg	374	207	64.37%
pred. pos	135	284	67.78%
class recall	73.48%	57.84%	

4.4 Without and with stemming

[Back to the top]

4.4.1 With stemming

Training data

accuracy: 76.80%			
	true neg	true pos	class precision
pred. neg	440	181	70.85%
pred. pos	51	328	86.54%
class recall	89.61%	64.44%	

Testing data

accuracy: 67.00%			
	true neg	true pos	class precision
pred. neg	384	205	65.20%
pred. pos	125	286	69.59%
class recall	75.44%	58.25%	

4.4.2 Without stemming

Training data

accuracy: 77.40%			
	true neg	true pos	class precision
pred. neg	436	171	71.83%
pred. pos	55	338	86.01%
class recall	88.80%	66.40%	

Testing data

accuracy: 65.80%			
	true neg	true pos	class precision
pred. neg	374	207	64.37%
pred. pos	135	284	67.78%
class recall	73.48%	57.84%	

4.5 POS tags

[Back to the top]

4.5.1 POS = JJ.*|NN.*|RB.*|VB.*

Training data

accuracy: 76.40%			
	true neg	true pos	class precision
pred. neg	440	185	70.40%
pred. pos	51	324	86.40%
class recall	89.61%	63.65%	

Testing data

accuracy: 66.60%			
	true neg	true pos	class precision
pred. neg	379	204	65.01%
pred. pos	130	287	68.82%
class recall	74.46%	58.45%	

4.5.2 POS = NN.*|RB.*|VB.*

Training data:

accuracy: 76.90%			
	true neg	true pos	class precision
pred. neg	437	177	71.17%
pred. pos	54	332	86.01%
class recall	89.00%	65.23%	

Testing data:

accuracy: 63.70%			
	true neg	true pos	class precision
pred. neg	375	229	62.09%
pred. pos	134	262	66.16%
class recall	73.67%	53.36%	

4.6 Pruning [Back to the top]

4.6.1 With pruning

Training data:

accuracy: 75.90%			
	true neg	true pos	class precision
pred. neg	446	196	69.47%
pred. pos	45	313	87.43%
class recall	90.84%	61.49%	

Testing data:

accuracy: 64.90%			
	true neg	true pos	class precision
pred. neg	384	226	62.95%
pred. pos	125	265	67.95%
class recall	75.44%	53.97%	

4.6.2 Without pruning

Training data:

accuracy: 76.80%			
	true neg	true pos	class precision
pred. neg	440	181	70.85%
pred. pos	51	328	86.54%
class recall	89.61%	64.44%	

Testing data:

accuracy: 67.00%			
	true neg	true pos	class precision
pred. neg	384	205	65.20%
pred. pos	125	286	69.59%
class recall	75.44%	58.25%	

4.7Comparison of different models

[Back to the top]

4.7.1 SVM with dot-kernel

Training data:

accuracy: 100.00%			
	true neg	true pos	class precision
pred. neg	491	0	100.00%
pred. pos	0	509	100.00%
class recall	100.00%	100.00%	

Testing data:

accuracy: 77.50%			
	true neg	true pos	class precision
pred. neg	363	79	82.13%
pred. pos	146	412	73.84%
class recall	71.32%	83.91%	

4.7.2 SVM with radial-kernel

Training data:

accuracy: 50.90%			
	true neg	true pos	class precision
pred. neg	0	0	0.00%
pred. pos	491	509	50.90%
class recall	0.00%	100.00%	

Testing data:

accuracy: 49.10%			
	true neg	true pos	class precision
pred. neg	0	0	0.00%
pred. pos	509	491	49.10%
class recall	0.00%	100.00%	

4.7.3 Random forest

Training data

Table View   Plot View

accuracy: 52.30%

	true neg	true pos	class precision
pred. neg	25	11	69.44%
pred. pos	466	498	51.66%
class recall	5.09%	97.84%	

Testing data

Table View   Plot View

accuracy: 49.60%

	true neg	true pos	class precision
pred. neg	18	13	58.06%
pred. pos	491	478	49.33%
class recall	3.54%	97.35%	

4.7.4 KNN with K = 4

Training data:

accuracy: 76.80%

	true neg	true pos	class precision
pred. neg	440	181	70.85%
pred. pos	51	328	86.54%
class recall	89.61%	64.44%	

Testing data:

accuracy: 67.00%

	true neg	true pos	class precision
pred. neg	384	205	65.20%
pred. pos	125	286	69.59%
class recall	75.44%	58.25%	

4.7.5 Naïve Bayes

Training data

accuracy: 99.90%

	true neg	true pos	class precision
pred. neg	491	1	99.80%
pred. pos	0	508	100.00%
class recall	100.00%	99.80%	

Testing data

accuracy: 64.90%			
	true neg	true pos	class precision
pred. neg	309	151	67.17%
pred. pos	200	340	62.96%
class recall	60.71%	69.25%	

4.8 Comparison of different models using Harvard dictionary [Back to the top]

4.8.1 Naïve Bayes

Training data

accuracy: 93.20%			
	true neg	true pos	class precision
pred. neg	487	64	88.38%
pred. pos	4	445	99.11%
class recall	99.19%	87.43%	

Testing data

accuracy: 57.60%			
	true neg	true pos	class precision
pred. neg	299	214	58.28%
pred. pos	210	277	56.88%
class recall	58.74%	56.42%	

4.8.2 SVM with dot-kernel

Training data

Table View Plot View

accuracy: 97.20%			
	true neg	true pos	class precision
pred. neg	489	26	94.95%
pred. pos	2	483	99.59%
class recall	99.59%	94.89%	

Testing data

Table View Plot View

accuracy: 67.10%			
	true neg	true pos	class precision
pred. neg	384	204	65.31%
pred. pos	125	287	69.66%
class recall	75.44%	58.45%	

4.8.3 Random forest

Training data

Table View Plot View

accuracy: 55.90%

	true neg	true pos	class precision
pred. neg	348	298	53.87%
pred. pos	143	211	59.60%
class recall	70.88%	41.45%	

Testing data

Table View Plot View

accuracy: 53.40%

	true neg	true pos	class precision
pred. neg	350	307	53.27%
pred. pos	159	184	53.64%
class recall	68.76%	37.47%	

4.8.4 KNN with K=4:

Training data

Table View Plot View

accuracy: 75.90%

	true neg	true pos	class precision
pred. neg	446	196	69.47%
pred. pos	45	313	87.43%
class recall	90.84%	61.49%	

Testing data

accuracy: 61.00%

	true neg	true pos	class precision
pred. neg	377	258	59.37%
pred. pos	132	233	63.84%
class recall	74.07%	47.45%	

4.9 Word net dictionary

[Back to the top]

4.9.1 Naïve bayes

Training data

accuracy: 100.00%

	true neg	true pos	class precision
pred. neg	491	0	100.00%
pred. pos	0	509	100.00%
class recall	100.00%	100.00%	

Testing data

accuracy: 63.00%			
	true neg	true pos	class precision
pred. neg	281	142	66.43%
pred. pos	228	349	60.49%
class recall	55.21%	71.08%	

4.9.2 SVM with dot-kernel

Training data

accuracy: 100.00%			
	true neg	true pos	class precision
pred. neg	491	0	100.00%
pred. pos	0	509	100.00%
class recall	100.00%	100.00%	

Testing data

accuracy: 75.60%			
	true neg	true pos	class precision
pred. neg	363	98	78.74%
pred. pos	146	393	72.91%
class recall	71.32%	80.04%	

4.9.3 Random forest

Training data

accuracy: 50.90%			
	true neg	true pos	class precision
pred. neg	0	0	0.00%
pred. pos	491	509	50.90%
class recall	0.00%	100.00%	

Testing data

accuracy: 49.10%			
	true neg	true pos	class precision
pred. neg	0	0	0.00%
pred. pos	509	491	49.10%
class recall	0.00%	100.00%	

4.9.4 KNN with K = 4

Training data

accuracy: 78.00%			
	true neg	true pos	class precision
pred. neg	434	163	72.70%
pred. pos	57	346	85.86%
class recall	88.39%	67.98%	

Testing data

accuracy: 64.90%			
	true neg	true pos	class precision
pred. neg	366	208	63.76%
pred. pos	143	283	66.43%
class recall	71.91%	57.64%	