Machine learning assignment 2

(Gulshan Jangid, 2014CS50285)

PART(A) (Naive bayes):

(A)

Accuracy on test = 60.07%

Accuracy on train(20% data) = 63.48%

(B)

Random test accuracy = 20.0803 %

Random train accuracy(20% data) = 19.897 %

Majority test accuracy = 43.9895 %

Majority train accuracy(20% data) = 43.8214 %

(C) Confusion matrix:

	1	2	3	4	5
1	14346	2800	1356	1091	3177
2	3818	3282	1682	723	328
3	1183	3330	5209	2553	651
4	465	1075	5377	18030	15207
5	357	351	907	6961	39459

Five stars has the highest value in the diagonal entry, also out of 58,822 five stars, algorithm predicted 39459 (67%) of them correctly.

Out of 20169 one stars, algorithm predicted 14346(71.1%) of them correct.

This shows that 1 star and 5 star have pretty distinct feature words which have a lot of effect in determination of these two classes.

Algorithm is poor in performing prediction for 2 and 3 star classes.

(D)

Stemming and stopwords removal:

Test accuracy = 59.517 %

Train accuracy(20% data) = 62.534 %

Stemming reduces accuracy in this case.

(E)

Two alternative features we can use are:

(1) Bigrams
(2) Part of speech tagging(POS)

POS accuracy = 60.54% , not a significant change

Bigrams Test accuracy = 63.888 %

Bigrams Train accuracy(20%) = 82.666 %

Bigrams with stemming and stopwords:

Accuracy on test = 63.000 %

Accuracy on train(20%) = 89.959 %

Bigrams without stemming helps improve the overall accuracy.

(F)

F1 score for each class in the test set:

1	2	3	4	5
74.51%	16.175%	22.856 %	52.166%	79.16%

Average = 49.10%

F1 score is more suited because it takes care of the distribution rather than blindly going for the number of correct samples. For example in a dataset with 99.1% healthy and 0.09% cancer patients ,A model which says healthy always has the accuracy of 99.1% but will have poor f score.

So accuracy doesn’t make sense here rather we focus on f score.

(G)

Bigrams worked better when trained with full_train data.

Using bigrams train_full.json and testing on test.json

Test accuracy = 68.387 %

Test macro f score = 63.75 %

PART(B) (SVM):

(1) Binary classification-

(A) (cvxopt) (linear)

Total training points = 4000

With d = 5

Training time = 27.8 seconds

accuracy = 97.29%

(Correct/Total) for 5 = 864/892

(Correct/Total) for 6 = 936/958

b = -1.624

numSV = 233, taking cutoff of 10^{-5}

(B) (cvxopt) (gaussian)

Training time = 21.3 seconds

Accuracy = 99.19%

Correct/Total for 5 = 884/892

Correct/Total for 6 = 951/958

b = -0.1511

numSV = 1520, taking cutoff of 10^{-5}

Gaussian kernel performs better than the linear one.

(C) (LIBSVM) (linear)
training time = 1.93 seconds
b = -1.6244
numSV = 233
Accuracy = 97.2972972973 %
correct 5 = 864, total 5 = 892
correct 6 = 936, total 6 = 958
Pretty similar compared to cvxopt linear

(LIBSVM) (gaussian)
training time = 6.47399997711 seconds
b = -0.140648214741
numSV = 1478
Accuracy = 99.1891891892 %
correct 5 = 884, total 5 = 892
correct 6 = 951, total 6 = 958
Cvxopt has somewhat extra support vectors compared to libsvm and also the difference in b value, accuracy is similar.

(2) Multiclass classification-

Cvxopt:
Test Accuracy = 97.15%
Train accuracy = 99.92%
Libsvm:
Test Accuracy = 97.24 %

Confusion matrix (for cvxopt):

	0	1	2	3	4	5	6	7	8	9	
0	[969	0	4	0	0	2	6	1	4	4	]
1	[0	1120	0	0	0	0	3	4	0	4	]
2	[1	4	1006	12	4	4	0	23	3	5	]
3	[0	3	1	977	0	5	0	2	7	7	]
4	[0	0	1	0	961	1	4	6	2	14	]
5	[3	2	0	5	0	868	4	0	5	6	]
6	[4	2	1	0	7	6	938	0	1	0	]
7	[1	0	6	6	0	1	0	979	2	6	]
8	[2	3	13	9	2	5	3	3	947	13	]
9	[0	1	0	1	8	0	0	10	3	950]	

Digit 9 is worst classified with 5.84%(52 out of 1009) of the examples classified as some other digit