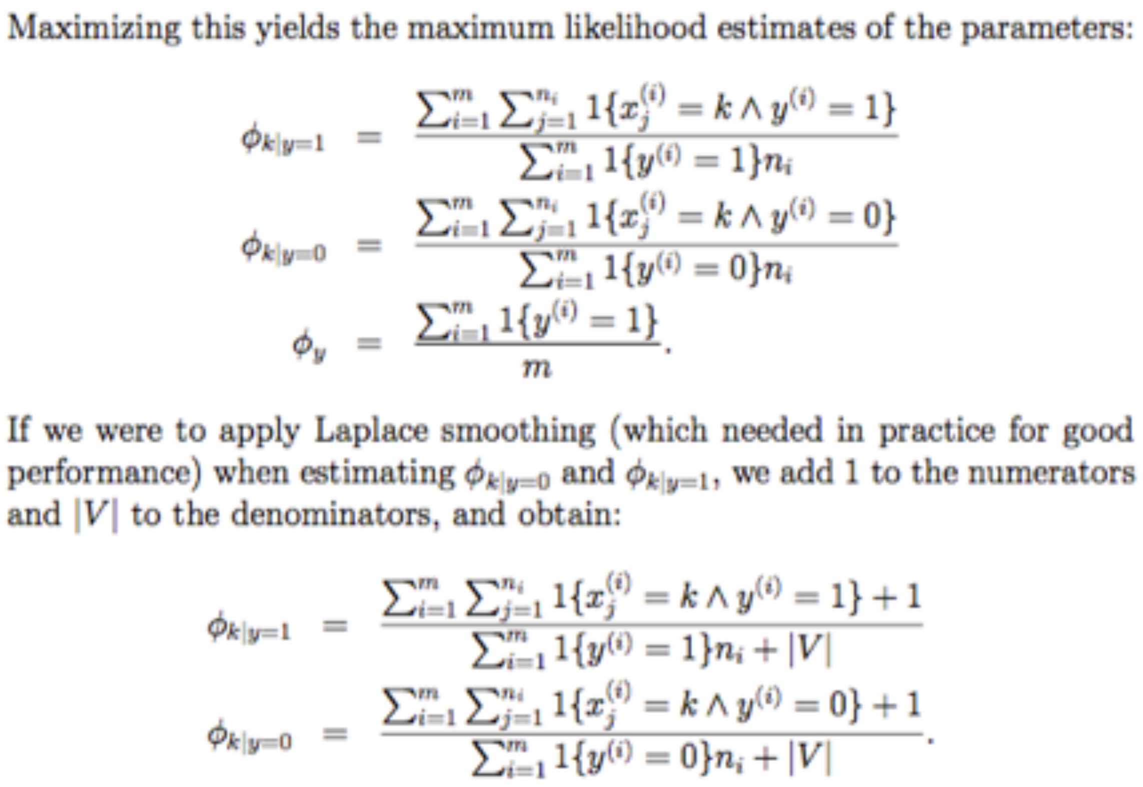
ASSIGNMENT-2

Name: Udayin Biswas
Entry No.:2014CS50296

1. TEXT CLASSIFICATION

(A) Implementing Naive Bayes Model(multinomial event model)



Maximizing this yields the maximum likelihood estimates of the parameters:

$$\begin{aligned}
\phi_{k|y=1} &= \frac{\sum_{i=1}^{m}\sum_{j=1}^{n_i} 1\{x_j^{(i)} = k \wedge y^{(i)} = 1\}}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} n_i} \\
\phi_{k|y=0} &= \frac{\sum_{i=1}^{m}\sum_{j=1}^{n_i} 1\{x_j^{(i)} = k \wedge y^{(i)} = 0\}}{\sum_{i=1}^{m} 1\{y^{(i)} = 0\} n_i} \\
\phi_y &= \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 1\}}{m}.
\end{aligned}$$

If we were to apply Laplace smoothing (which needed in practice for good performance) when estimating $\phi_{k|y=0}$ and $\phi_{k|y=1}$, we add 1 to the numerators and $|V|$ to the denominators, and obtain:

$$\begin{aligned}
\phi_{k|y=1} &= \frac{\sum_{i=1}^{m}\sum_{j=1}^{n_i} 1\{x_j^{(i)} = k \wedge y^{(i)} = 1\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} n_i + |V|} \\
\phi_{k|y=0} &= \frac{\sum_{i=1}^{m}\sum_{j=1}^{n_i} 1\{x_j^{(i)} = k \wedge y^{(i)} = 0\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = 0\} n_i + |V|}.
\end{aligned}$$

This was modified by using logarithms to avoid underflow issues. C=1 hence 1 in numerator and IVI in denominator for Laplace smoothing.
Delimiter used to separate words from sentences is space tab.
For punctuation, **. , ( ) / <br /><br /> \n -** replaced.

Observations:
(a) Vocabulary length=121669
(b) Training Accuracy: 0.68496 or 68.49%
(c) Testing Accuracy: 0.38624 or 38.66%

(B) Test set accuracy

1.When randomly guessing one of the categories as the target class for each of the articles:
Testing accuracy= 0.12348 or 12.35% (Changes every time)

2.When making a majority prediction:
Training label occurring in majority = 1
Testing accuracy = 0.20088 or 20.09%

Improvement of multinomial event model of Naive Bayes theorem over random prediction:
Testing accuracy(NBT)=38.62%

Testing accuracy(Random)=12.35%
Improvement over random prediction = (38.66-12.35)/12.35 = 213.03% improvement on 25000 labels
Testing accuracy(Majority)=20.09%
Improvement over majority prediction = (38.66-20.09)/20.09 = 92.4% improvement on 25000 labels.

(C) Confusion matrix

	1	**10**	**2**	**3**	**4**	**7**	**8**	**9**
1	**4458**	813	1749	1553	1205	470	455	341
10	159	**3381**	119	188	249	675	1250	1319
2	23	8	**22**	21	9	2	3	2
3	77	14	87	**107**	112	31	31	13
4	203	73	224	452	**612**	202	137	61
7	29	118	43	93	190	**320**	211	103
8	65	529	55	121	245	584	**722**	462
9	8	63	3	6	13	23	41	**43**

Class label/Category 1 has the highest value of the diagonal entry.

Observations:

1. Class label 1 and 10 are detected correctly with a higher accuracy.(Diagonal elements)
2. Other class labels have less accuracies in being detected correctly.
3. Class label 1 is predicted most number of times by our algorithm.Even for other class labels, it is dominating by predicting class label 1 and by class label 10 in some instances.

(D) Removing stopwords and Stemming

Vocabulary length = 51265
Training accuracy = 0.6798 or 67.98%
Test accuracy = 0.38448 or 38.45%

Change of accuracy is visible, but not significant.
In fact accuracy decreases a bit surprisingly over both training and test data - a possible reason can be that after filtering by stemming and stopwords, more common words appear across the training text of different class labels and hence less distinction made.

(E) Feature Engineering

Features added:

(1) Reducing the significance of common words across many classes: Suppose a word ‘great’ appears in more than 5 classes out of 8 with each class label containing a significant portion of the word in its corresponding text(>14% of the total occurrences of the word ‘great’ across all classes), then it will be treated as a frequent word and should not contribute to our Naive Bayes theorem. The logic is that it doesn’t help us distinguish significantly across the classes

and infact is confusing for it to be used for NBT. We need to remove these common words from vocabulary.

(2) Treating words as bigrams: Treating two words as a consecutive feature can be effective here as a single word sometimes is not enough to distinguish between features, and combinations of 2 words are effective for dependent words.

<u>Features added on top of stemmed and stopword removal model:</u>

Training accuracy: 99.51%
Test accuracy: 39.38%

<u>Comparison with part (a) and part (d):</u>

<u>Part (a)</u> : Test Accuracy: 38.66%
Improvement of 39.38-38.66=<u>0.74%</u>

<u>Part (d)</u> : Test accuracy = 38.45%
Improvement of 39.38-38.45=<u>0.93%</u>

So it can be observed adding the two features helps improve the model. Bigram feature is more effective when tested individually as compared to removing the common words feature. An improvement of 0.93% over stemmed- stopword removed data is observe.

2. TEXT CLASSIFICATION

(a) <u>Implementing mini-batch version of Pegasus algorithm</u>

Convergence criterion: Setting the number of iterations=1000
Batch size, k=100
Setting w and b to 0 initially.
The function 'pegasos' takes the class labels of two digits being compared and assigns y=-1 to smaller digit and y=1 to larger digit.

<u>(b)SVM extension to multi-class</u>

Total number of classifiers= kC2 i.e.10C2 = 45 classifiers
Each classifier for comparing between 2 digits out of 0 to 9.
Multiclass SVM (C=1,K=100)

=> Output the classifier which has the maximum number of wins among all the 45 classifiers. (In case of a tie between two digits, I take the one which is less in value.)

Training accuracy : 91.225%
Test accuracy : 91.32%

<u>(c)Multiclass SVM training using LIBSVM</u>

Using LIBSVM library,
First scaling of data done: `$ svm-scale -l 0 -u 1 ../train_libsvm.txt > scaled_train.txt`

For linear kernels:
C=1.0

```
$svm-train -t 0 -c 1 scaled_train.txt q22model.model
$svm-predict scaled_test.txt  q22model.model train_output_linear.csv
```

Testing accuracy:
Accuracy = 92.78% (9278/10000) (classification)

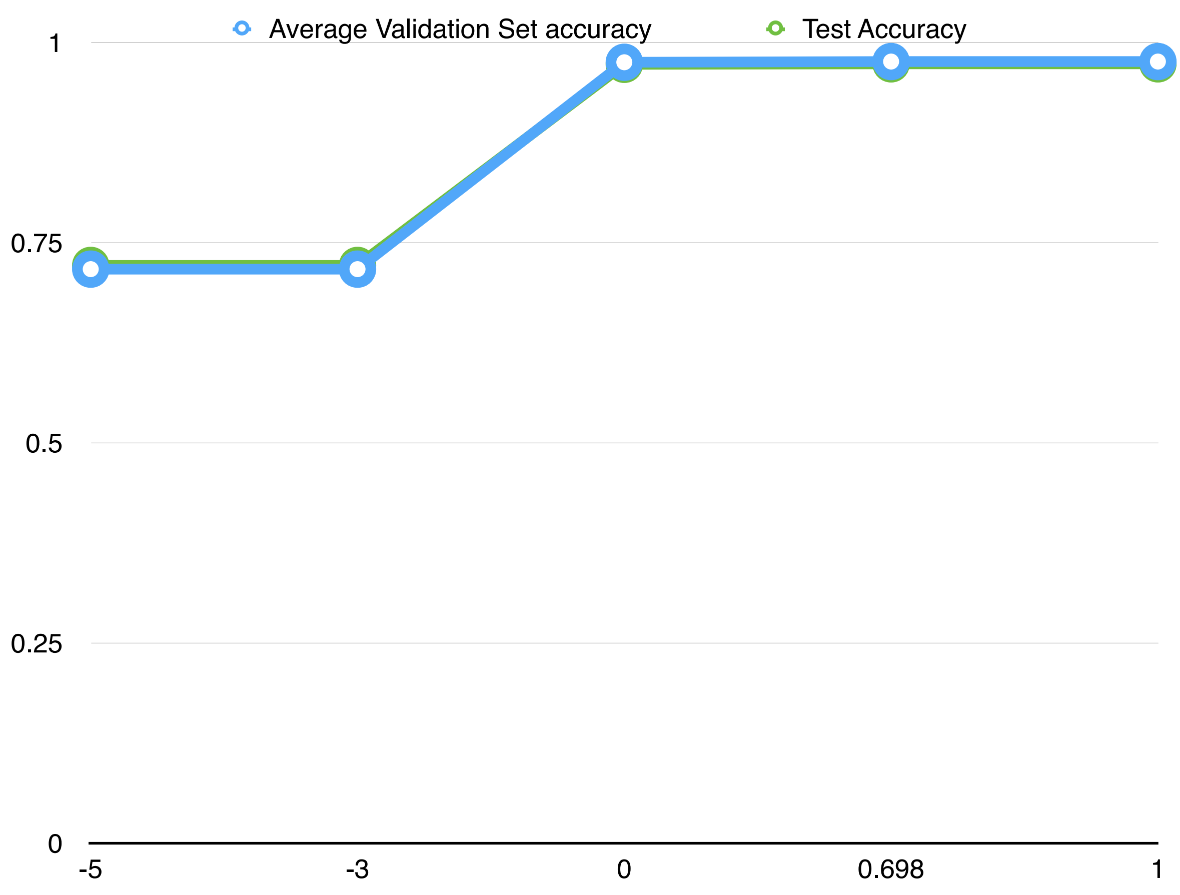
For gaussian kernel:
γ= 0.05, C=1.0

```
$svm-train -t 2 -c 1 -g 0.05 scaled_train.txt q23model.model
$svm-predict .scaled_test.txt q23model.model train_output_gaussian.csv
```

Testing accuracy:
Accuracy = 97.23% (9723/10000) (classification)

(d) Crossfold validation:

(1) C=0.00001, γ=0.05 : Cross validation accuracy = 71.645%, Test set accuracy = 72.11%
(1) C=0.001, γ=0.05 : Cross validation accuracy = 71.645%, Test set accuracy = 72.11%
(1) C=1, γ=0.05 : Cross validation accuracy = 97.485%, Test set accuracy = 97.23%
(1) C=5, γ=0.05 : Cross validation accuracy = 97.575%, Test set accuracy = 97.29%
(1) C=10, γ=0.05 : Cross validation accuracy = 97.575%, Test set accuracy = 97.29%



Average Validation Set accuracy and Test accuracy vs log(C) to base 10

From the observations, it is observed that C=5 and C=10 give the best cross validation accuracy of 97.575%.
Yes, they also yield the best test accuracy of 97.29%.

From the plot, it can be seen that average validation set accuracy and test accuracy are almost the same i.e. overlap w.r.t value of log(C).

(e) Confusion matrix:

Best Libsvm model (rbf kernel) corresponding to part d: C=10(Choose over C=5 because had to choose one of them as both give same accuracy)

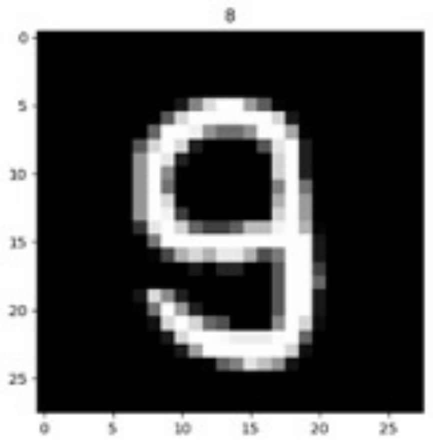
```
$svm-train -t 2 -c 10 -g 0.05 scaled_train.txt q23model.model
$svm-predict scaled_test.txt q23model.model q23output.txt
```

	0	1	2	3	4	5	6	7	8	9
0	**969**	0	4	0	1	2	5	1	4	4
1	0	**1122**	0	0	0	0	4	4	0	4
2	1	3	**1000**	8	4	3	0	20	3	3
3	0	2	4	**985**	0	6	0	2	10	8
4	0	1	2	0	**962**	1	3	3	1	9
5	3	2	0	4	0	**866**	4	0	5	4
6	4	2	1	0	5	7	**940**	0	3	0
7	1	0	6	7	0	1	0	**986**	3	9
8	2	2	15	5	2	5	2	2	**942**	11
9	0	1	0	1	8	1	0	10	3	**957**

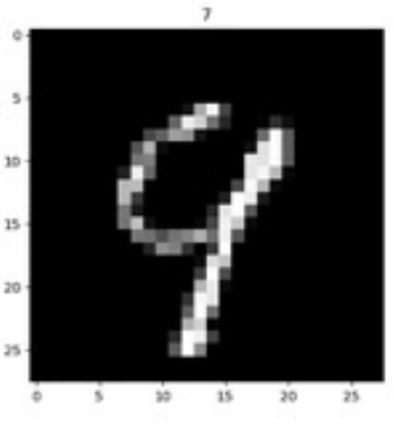
Class 9 seems to be the most difficult to classify, as observed from the confusion matrix. As it confuses the 9 with some other digit in the most number of instances relatively, it has the highest chance of being misrepresented by another digit.

Three misclassified examples:

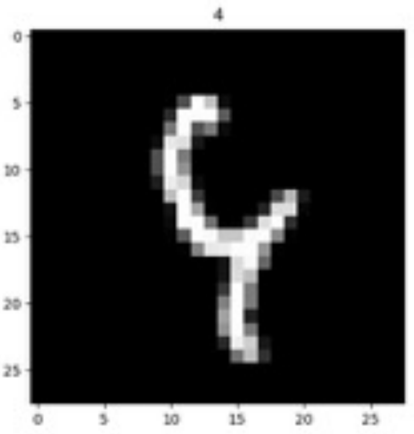
(1) 9(actual) being confused with 8(predicted)



(2) 9(actual) being confused with 7(predicted)



(3) 9(actual) being confused with 4(predicted)



Comment: The example can be misclassified because of the fact that the 28*28 representation of the bit image being similar as the digits are very close to each other in resemblance. The values in the confusion matrix prove this.