Gaurav Shukla
2018MCS2141

Machine Learning(COL774)

Assignment 2

April 4, 2019

Problem 1 : Naive Bayes Text Classification

Answer to the problem goes here.

1. Naive bayes classifier gives following accuracies:-
   Test data accuracy = 61.87 %
   Train data accuracy = 69.60 %

2. Random and Majority Predictions:-
   Test set accuracy on Random prediction = 19.98%
   Test set accuracy on Majority prediction = 43.98%
   We got 1.41 times more accuracy than Majority Prediction and around 3.09 times more accuracy than Random Prediction.

3. Confusion Matrix on test data:

$$\begin{bmatrix} 15493 & 2174 & 1124 & 873 & 505 \\ 3377 & 1962 & 2395 & 2086 & 478 \\ 1615 & 819 & 3646 & 7302 & 1149 \\ 1067 & 266 & 1137 & 18545 & 8343 \\ 2479 & 107 & 253 & 12897 & 43086 \end{bmatrix}$$

   We can see that Class 5 has the highest diagonal value, which means that there are highest true positives for this class i.e most of the times the prediction for this class is true.
   The non diagonal entries of the confusion matrix shows wrong predictions, which means larger the non diagonal entries, less accurate will be the prediction.

   We can also observe that the entries far from the diagonal entries are lesser than those which are nearer. This implies that there is high chance of misclassification between nearer classes i.e there is a higher probability of misclassifying class 4 to class 5 than classifying it to class 1.

4. Stopwords occurs very frequently in any sentence and they actually do not contribute much in the prediction, so there removal along with some other feature helps in increasing accuracy. Stemming helps us to treat different forms of a word as same word, which helps in recognizing same words thus increasing accuracy.

   Accuracy on Stemming and stopwords removal:-
   Accuracy = 60.68 %

5. **Feature Engineering (Bi grams with Stemming and Bi gram with Lemmatization)**

Lemmatization extracts root form of the word. We will use bigrams on stemmed documents as well as lemmatized words

Test set Accuracy on stemming+bigram = 63.92333 %
Test set Accuracy on lemmatization+bigram = 64.27%

Clearly both the methods gives better accuracies than the accuracies we obtained in part (a) and part (d).

We are predicting based on the kind of words. So all the forms of same word should not contribute independently, which was obtained by using preprocessing of text like lemmatization and stemming. This is why the accuracy is much improved in later case.

6. **F1 Score**

F1 score for the best performing model(lemmatization+bigram) = $\begin{bmatrix} 0.702 \\ 0.333 \\ 0.412 \\ 0.521 \\ 0.749] \end{bmatrix}$

The Average F1 score(Macro F1 score) = 0.544

F1 score metric is a measure of a test's accuracy which takes precision and recall into account. F1 score metric is computed based on true positives predicted by our model. while test error metric is derived by true positives and true negatives.
F1 score does not suit when there are very less positive classes and lots of negative classes, which is not the case in this data set, Therefore F1 score is better suited in this case.

7. **Running on 5M training data**
It takes around 10 hours to train the model on 5M training instances.
Accuracy and F1 score on original test data:-
Accuracy = 74.4978% Macro F1 score = 0.67367 %

Problem 2 : SVM Handwritten digit binary Classification

Answer to the problem goes here.

1. **Linear Kernel** The dual problem can be expressed as:

Assignment 2

$$\min_{\alpha} \frac{1}{2}\alpha^T K\alpha - 1^T\alpha \text{ s.t.} \quad \forall i 0 \leq \alpha_i \leq C, C = 1 \text{ and } \quad \sum \alpha_i y^{(i)} = 0$$

We will solve this dual problem using Convex Optimization and will get the values of α and then we will find the support vectors (i.e vectors with $\alpha > 0$). Here due to floating point operations, none of the alphas are exactly zero, so we will use some threshold value to get support vectors. I used $\alpha > 10^{-4}$

CVXOPT solves following equation format:-

$$\min_{x} \frac{1}{2}x^T Px + q^T x \text{ s.t.} \quad Gx \preceq h \text{ and } \quad Ax = b$$

Expressing the dual in the following format $\alpha^T Q\alpha + b^T\alpha + C$:
Here
C = 0
b is a vector containing only 1's.
$Q = YX^T XY^T$
We will compute vector W(size 784×1) and intercept term b(scalar) from α's and will compute accuracy.
The average test set accuracy = 99.03%

2. **Gaussian kernel** In case of Gaussian kernel, the inner product $x_i^T x_j$ will be simply replaced with $K(x_i, x_j)$.

   The matrix Q(in the format $\alpha^T Q\alpha + b^T\alpha + C$) will be $YY^T * K$ where $K_{ij} = K(x_i, x_j)$ and $*$ is element wise matrix multiplication.
   The accuracy of Binary classification using Gaussian kernel = 99.58467928011075 %

   We get better accuracy than linear kernel because the Gaussian kernel fits curve at least as accurate as linear, as it maps data in higher dimensions.

   Since the number of features are large enough in this data set, there will not be much improvement by mapping the data into higher dimensions. We can see that accuracy's are approximately same in our dataset.

3. **Libsvm**

   Accuracy in linear kernel = 99.35 %
   Accuracy in Gaussian kernel = 99.5847%

Due to linear boundary, number of points at unit distance will be less in linear kernel, consequently the number of support vectors are comparatively less in linear kernel than in Gaussian kernel. This is the reason Gaussian kernel performs better than linear model.

Now The number of support vectors and the value of parameters are same as obtained from libsvm but the libsvm is around 4 times faster than my implementation.

4. **Multi-class classification** For the given MNIST data set the accuracy are following :-
   Accuracy on test data = 97.23 %
   Accuracy on training data = 99.88 %

5. **Using libsvm -**
   Accuracy on test data = 97.23 %
   Accuracy on training data = 99.92 %
   We can see that that accuracy obtained by SVM module and the modal we have developed are approximately same. The little difference is because of the large floating point operation and the data structure that we use while implementation. The points that are differing in our output and the output obtained by LIBSVM are those points that are very near to the separating boundary.
   computational cost of the model implemented in (i) was around 40 minutes in my machine and LIBSVM takes around 25 minutes on the same data set. The difference is CVXOPT may use different algorithm for computing the optimal α (other than CVXOPT) and relax some parameter in order to make the model fast.

6. **Confusion matrix** We got the following confusion matrix while running the multi-class classification SVM model

```
    [[ 969    0    1    0    0    3    4    1    2    0]
     [   0 1121    3    2    1    2    2    0    3    1]
     [   4    0 1000    4    2    0    1    6   15    0]
     [   0    0    8  984    0    4    0    6    5    3]
C=   [   0    0    4    0  962    0    6    0    2    8]
     [   2    0    3    6    1  866    7    1    5    1]
     [   6    3    0    0    4    4  939    0    2    0]
     [   1    4   19    2    4    0    0  987    2    9]
     [   4    0    3   10    2    5    1    3  943    3]
     [   5    4    3    8   13    3    0    7   14  952]]
```

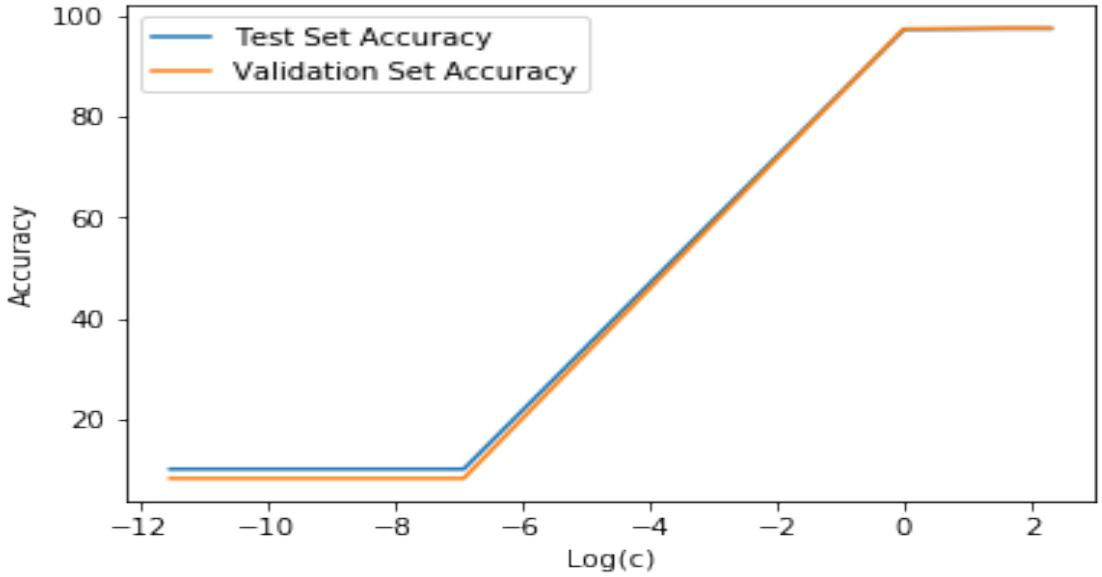
The diagonal entry of confusion matrix shows the true positives meaning the no of examples predicted correctly. From the above confusion matrix we can see that $C_{7,2} = 19$ that is highest among the non diagonal values. So we can say that 2^{nd} class is

predicted as 7^{th} class 19 times.
From the above confusion matrix we can see that the diagonal values are very large compared to non diagonal values hence accuracy is very good.

7. **Cross Validation-**

```
1e-05
Accuracy = 8.25% (165/2000) (classification)
Accuracy = 10.1% (1010/10000) (classification)
10.100000000000001
0.001
Accuracy = 8.25% (165/2000) (classification)
Accuracy = 10.1% (1010/10000) (classification)
10.100000000000001
1
Accuracy = 97.3% (1946/2000) (classification)
Accuracy = 97.19% (9719/10000) (classification)
97.19
5
Accuracy = 97.45% (1949/2000) (classification)
Accuracy = 97.34% (9734/10000) (classification)
97.34
10
Accuracy = 97.45% (1949/2000) (classification)
Accuracy = 97.34% (9734/10000) (classification)
97.34
```

While plotting the graph of accuracy against the value of C we get the following:-



figureValidation vs test accuracy in Multi-class SVM