COL774 – Machine Learning

Assignment 2

Nilaksh Agarwal
2015PH10813

Part A – Naïve Bayes

a) Implimenting NB:

Training
Accuracy – 0.69607
Macro F1 score – 0.61789

Test
Accuracy – 0.62053
Macro F1 score – 0.50157

b) Random/Majority Prediction:

Majority Prediction
Accuracy – 0.4399
Macro F1 score – 0.61
Random Prediction
Accuracy – 0.20174
Macro F1 score – 0.18406
Improvement over this baseline – 20%/40%

c) Confusion Matrix:

```
[[15605  1941  1062   972   589]
 [ 3458  1766  2757  2306   551]
 [ 1642   716  3216  7689  1268]
 [ 1059   187   862 18466  8784]
 [ 2340    49   138 12372 43923]]
```

Highest Diagonal Entry – 5/5 (Since max number of examples for 5 star)

Other Observations – Max misclassifications happen in middle ratings. (4 as 5 or 2 as 1). Since the degree of negativity isn't figured out by this model.

d) Stopwords Removal:

Test Accuracy – 60.85
Comments – Since stopwords contain words like not, didn't etc, which capture negative sentiments, removing these leads to slight reduction in accuracy.

e) Feature Engineering:

Features Selected –
- Tf-idf Matrix (to improve selection of in-domain words)
- Tri-grams (to capture longer range word dependencies)

Test Accuracy – 65.8%

f) Macro-F1 Score:

F1 Scores :

Class	F1 Score
1 Star	0.70
2 Star	0.23
3 Star	0.29
4 Star	0.52
5 Star	0.77
Macro F1	0.50

Here, Macro-F1 score is a better metric as it captures equally all the disparity among the different classes. So, in a case where one class has fewer examples, it will still show a bad F-score on predicting that class poorly.

g) Full 1M dataset:

Test Accuracy – 67.3%

Part B – Support Vector Machines

Part 1 – Binary Classification: (3 vs 4)

h) Linear Kernel – CVXOPT Package

Dual Form of the Problem:

$$\begin{aligned} \max_\alpha \quad & W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle \\ \text{s.t.} \quad & 0 \le \alpha_i \le C, \quad i = 1, \ldots, m \\ & \sum_{i=1}^{m} \alpha_i y^{(i)} = 0, \end{aligned}$$

The CVXOPT Quadratic Program solver has takes the Parameters (P,q,G,h,A,b)

$$\begin{aligned} \text{minimize} \quad & (1/2) x^T P x + q^T x \\ \text{subject to} \quad & Gx \preceq h \\ & Ax = b \end{aligned}$$

So here we obtain the values of the parameters:

- P = M x M array where each element is $y^{(i)} y^{(j)} \langle x^{(i)}, x^{(j)} \rangle$
- q = M sized array of -1 (so max becomes min)
- G = 2M x M array where first half is -1 diagonal and other is 1 diagonal
- h = 2M array where first column is 0 and next column is C
- A = $y^{(i)}$ matrix
- b = 0

Here $\langle x^{(i)}, x^{(j)} \rangle = x^{(i)}$ **dot** $x^{(j)}$

No. of Support vectors – 134
Weight Vectors: (784,1) shaped vector

$$w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}.$$

Intercept Term: -0.047

$$b^* = -\frac{\max_{i:y^{(i)}=-1} w^{*T} x^{(i)} + \min_{i:y^{(i)}=1} w^{*T} x^{(i)}}{2}.$$

Average Test accuracy = 99.5%

i) *Gaussian Kernel – CVXOPT Package*

Here Kernel = $K(x,z) = \exp\left(-\frac{||x-z||^2}{2\sigma^2}\right).$

Number of support vectors = 1386 (due to the feature based separation)

Using this formula for prediction:

$$\begin{aligned} w^T x + b &= \left(\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}\right)^T x + b \\ &= \sum_{i=1}^{m} \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b. \end{aligned}$$

Average Test accuracy = 99.8% (Better than linear kernel)

j) *Linear & Gaussian Kernel – LIBSVM Package*

Accuracy –
- Linear – 99.5%
- Gaussian – 99.8%

Train Time –
- Linear – 1.75s (37s for CVXOPT)
- Gaussian – 8.5s (4m for CVXOPT)

So we see, the accuracy of LIBSVM and our CVXOPT implementation is similar, but the training time is significantly better in LIBSVM
This is due to a highly optimised and parallelised code running in LIBSVM

Part 2 – Multi-class Classification:

a) *Gaussian Kernel – CVXOPT Package*

Train set accuracy – 99.765%
Test set accuracy – 94.95%

b) *Gaussian Kernel – LIBSVM Package*

Train set accuracy – 99.92%
Test set accuracy – 97.23%

Computation Time – 10 mins (Train + Prediction) (As compared to 6+ hours in CVXOPT)

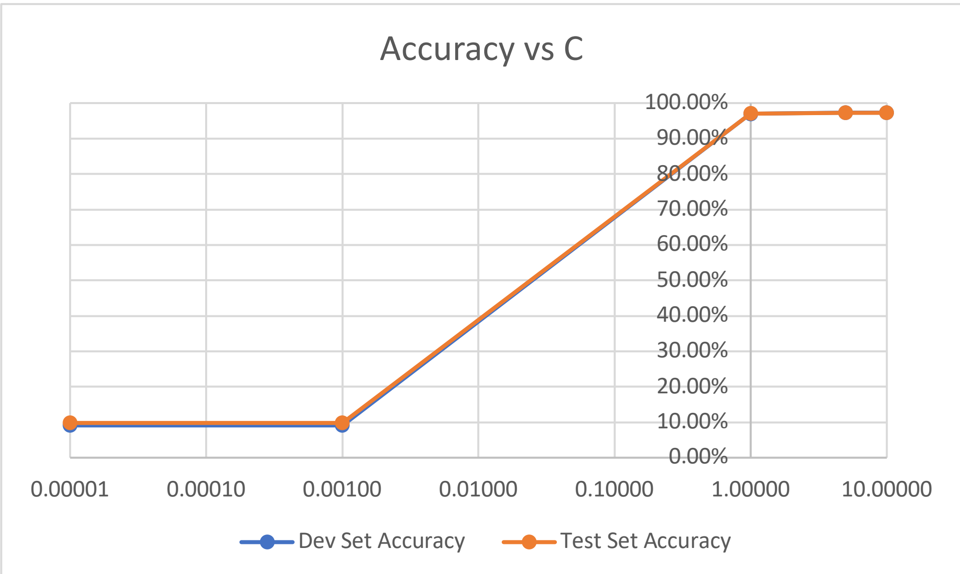
c) *Confusion Matrix*

(Rows – Gold, Columns – Predicted)

	0	1	2	3	4	5	6	7	8	9
0	969	0	4	0	0	2	6	1	4	4
1	0	1121	0	0	0	0	3	4	0	4
2	1	3	1000	8	4	3	0	19	3	3
3	0	2	4	985	0	6	0	2	10	8
4	0	1	2	0	962	1	4	4	1	13
5	3	2	0	4	0	866	4	0	5	4
6	4	2	1	0	6	7	939	0	3	0
7	1	0	6	6	0	1	0	987	3	9
8	2	3	15	5	2	5	2	2	942	12
9	0	1	0	2	8	1	0	9	3	952

Most misclassifications –
2 and 7, 2 and 8, 8 and 9, 4 and 9
Makes sense since these digits look similar to each other

d) *Hyper-Parameter Tuning*



Here, accuracy is almost constant for C between 1 and 10. So, this is the optimal value ofr this parameter. Too low a value of C results in underfitting and too high a value results in overfitting