COL774 Assignment – 2

Anirudha Kulkarni

2019CS50421

Code running instructions:

1. All codes are arranged in respective folders

2. Each code is in a python file and the parameters are written in the very first line of each function which can be changed

3. Data sets need to be put in a directory in submission directory with name "dataset" which can be passed as an argument if not

5. pretrained models are stored and can be used instead

Q2: Naïve Bayes

a) Naïve Bayes with Laplace smoothing and logarithms:

Train accuracy: 51.038

Test accuracy: 0.6595714285714286

b) Random predictions, Majority Predictions:

Random Train: 0.20118

Random Test: 0.1985

Majority Train: 0.51864

Majority Test: 0.6607857142857143

d) Removal of stopwords, stemming

Some New reviews are getting classified correctly and similarly new reviews are getting classified incorrectly. Overall accuracy increases slightly.

Train accuracy: 0.519

Test accuracy: 0.661

F1 Score

[0.00000000e+00 0.00000000e+00 0.00000000e+00 6.43086817e-04

7.95784946e-01]

Macro F1 Score

0.15928560661065586

```
[[    0     0     0     0  2529]
 [    0     0     0     1  2637]
 [    0     0     0     3  5631]
 [    0     0     0    19 13248]
 [    0     0     0     1 25931]]
```

e) Feature engineering:

i. Bingram

Train accuracy: 0.86476

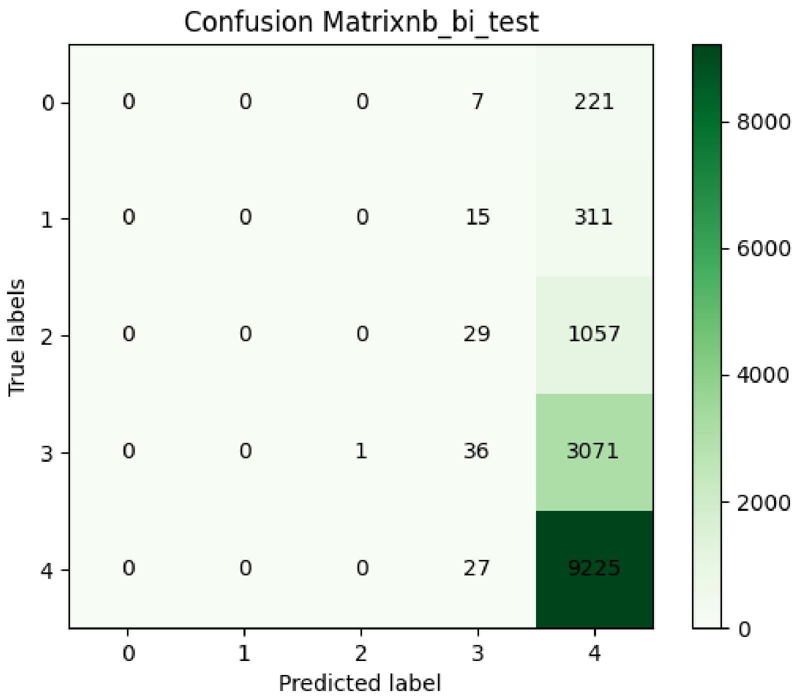
Test accuracy: 0.661428

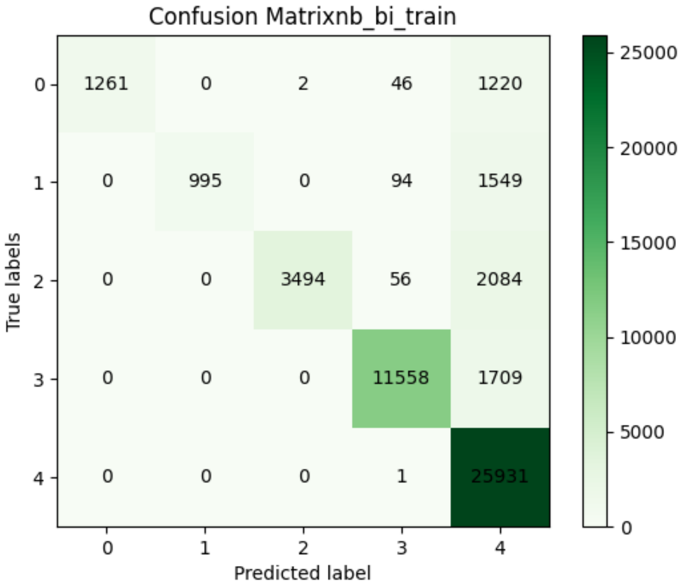
F1 Score

[0.   0.   0.   0.02234637 0.79742404]

Macro F1 Score

0.16395408162649344





ii. 3-gram

F1 Score

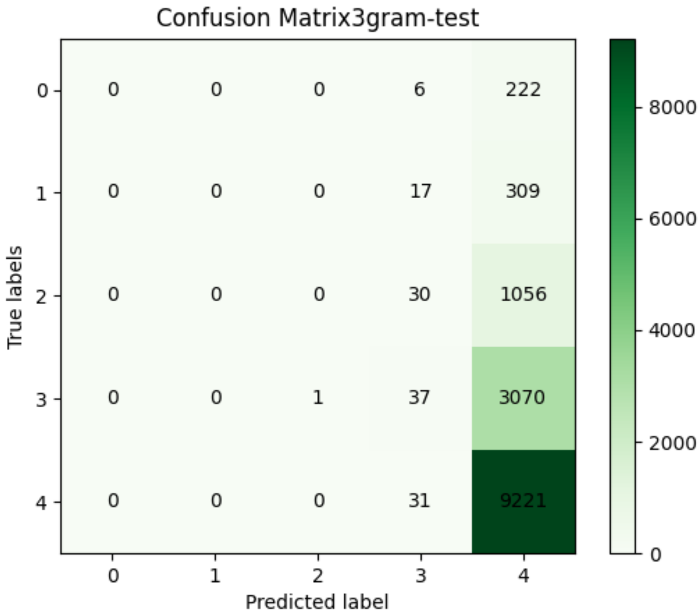
[0. 0. 0. 0.02291731 0.7973195]

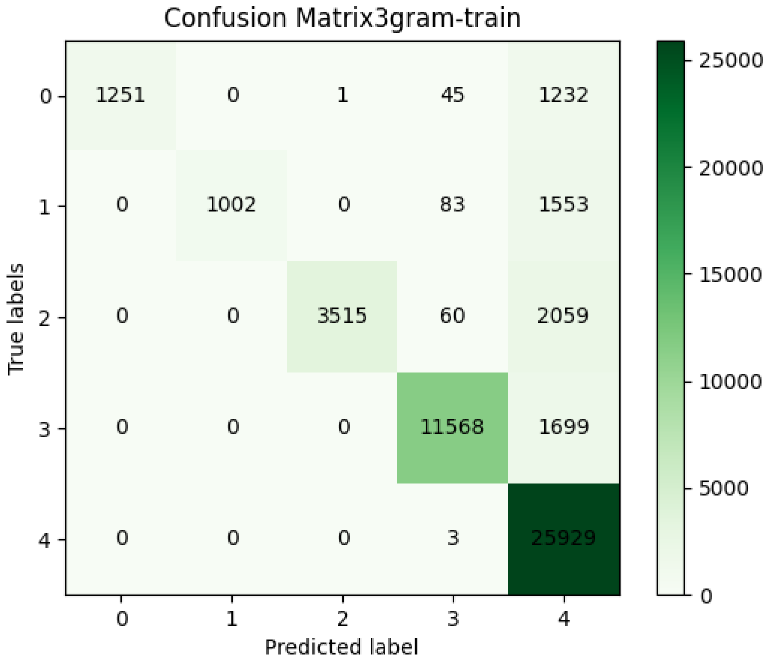
Macro F1 Score

0.16404736206961418

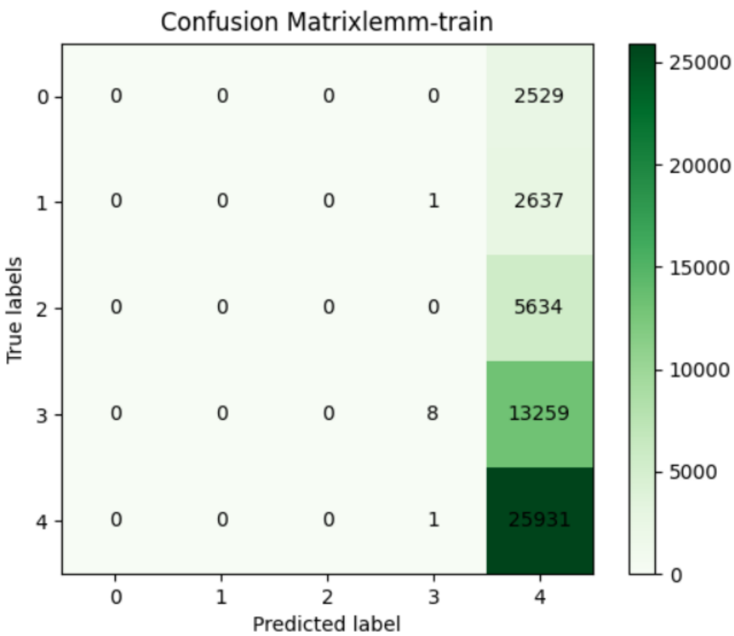
Training accuracy: 0.86528

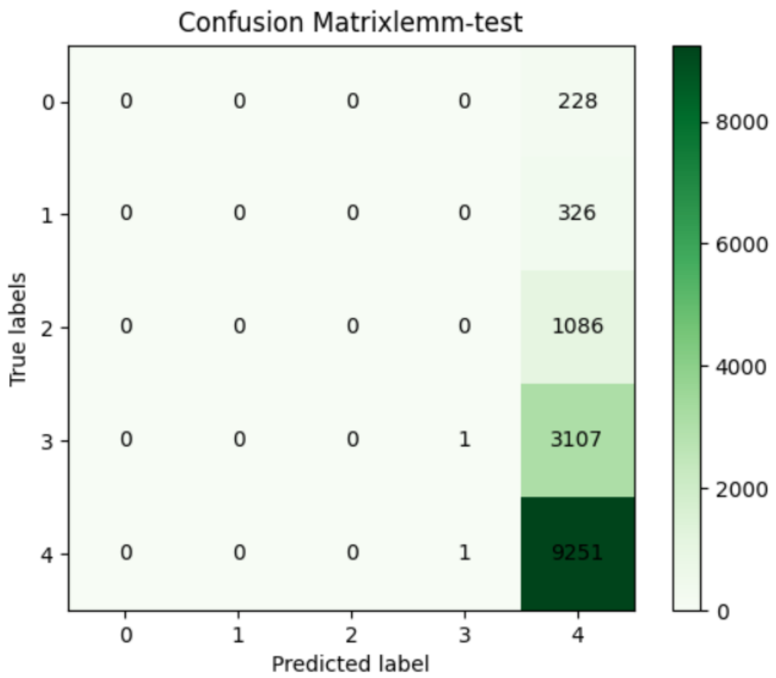
Testing accuracy: 0.6612142857142858





iii. Lemmatization





F1 Score

[0.00000000e+00 0.00000000e+00 0.00000000e+00 6.43086817e-04

7.95784946e-01]

Macro F1 Score

0.15928560661065586

Training accuracy: 0.51878

Testing accuracy: 0.6607857142857143

iv. TF-IDF

Test accuracy: 0.66086

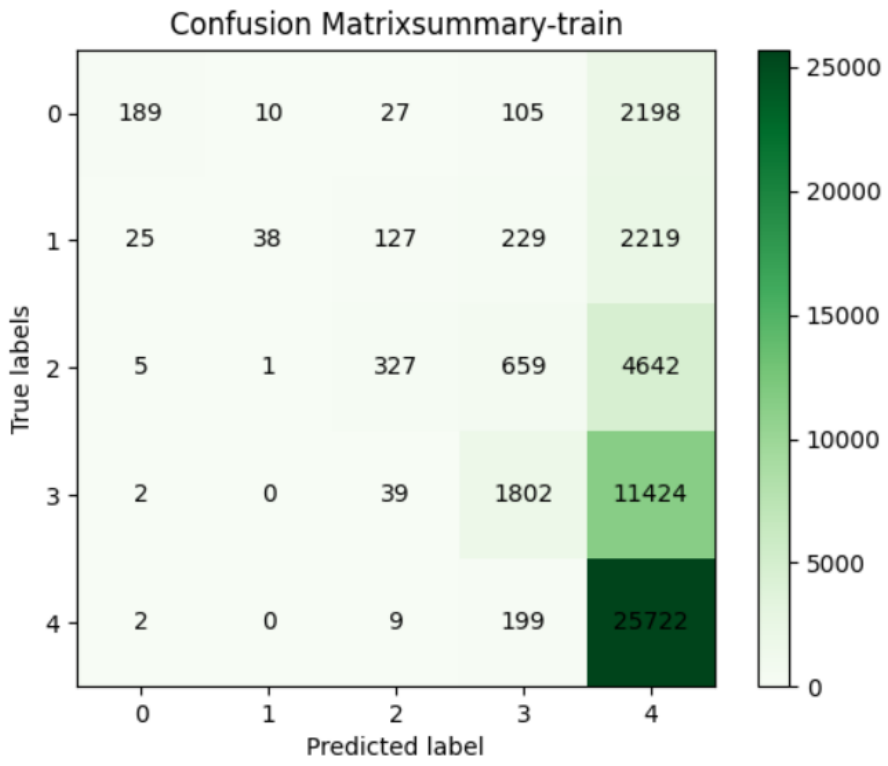
Macro F1 score : 0.16079297464750816

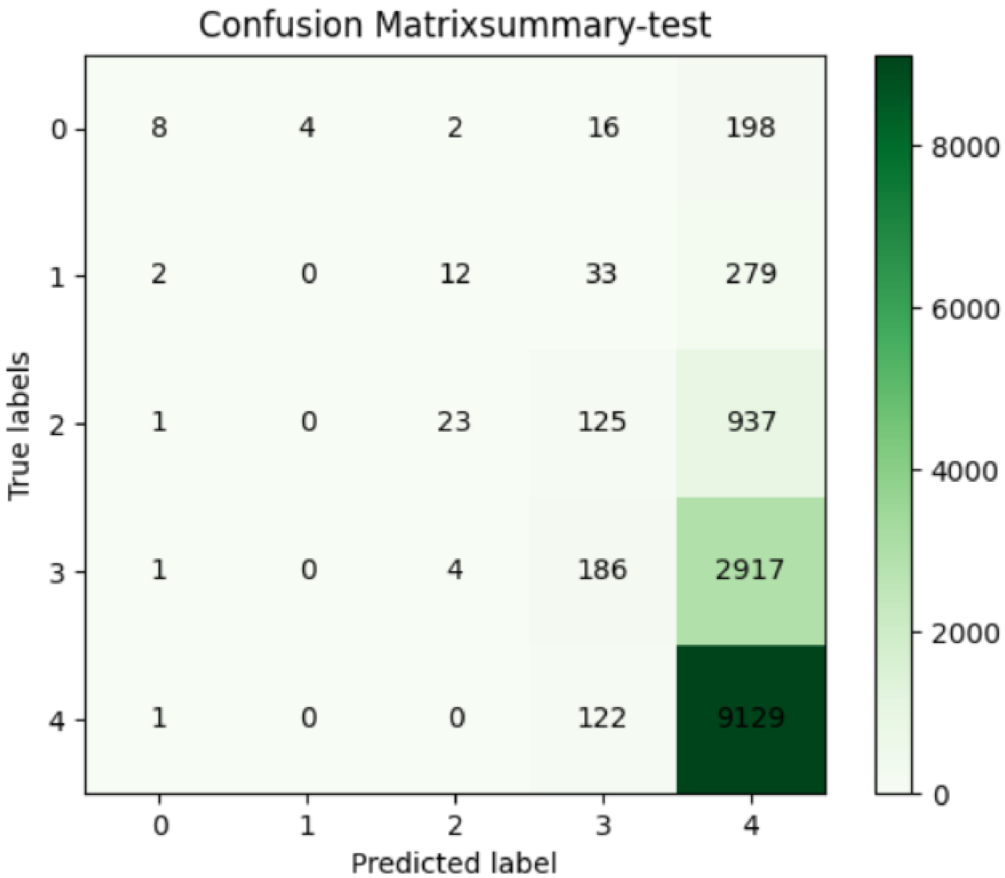
Among all feature engineering 2-gram fits training data most accurately but performs average on test data. 3-gram performs the best on test data. TF-IDF performs similar to majority predictions. Lemmatization reduces the accuracy when done with stemming. Without stemming and only lemmatization also reduces the accuracy.

f) F1- score: Done in each part

F1 score is better parameter as it takes the false positive and false negative into account. Will be critical where false positive and false negative matter. Ex- classifying terrorist as non-terrorist or important mail as spam.

g) With summary removal of stopwords, stemming





Confusion Matrix

```
[[   8    4    2   16  198]
 [   2    0   12   33  279]
 [   1    0   23  125  937]
 [   1    0    4  186 2917]
 [   1    0    0  122 9129]]
```

F1 Score

[0.06639004 0.   0.04081633 0.10362117 0.80389222]

Macro F1 Score

0.202943950701937

Summaries are generally short conclusion by a human which is more accurate than ML model. Hence accuracy is more than review text

---

Q2: SVM

a) Binary classification:

Dual problem with noise:

Dual:-

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} x^{(i)T} x^{(j)}$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \qquad \underbrace{0 \le \alpha_i \le C}_{\text{Box constraints}}$$

CVXOPT Format:

$$\begin{aligned} \min_{x} \quad & \tfrac{1}{2} x^{\top} P x + q^{\top} x \\ \text{subject to} \quad & Gx \preceq h \\ & Ax = b \end{aligned}$$

Max can be converted to min using negative sign,

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} x^{(i)T} x^{(j)} - \sum_{\alpha} \alpha$$

i) Linear Kernel:

Finding P,q,G,h,A,b:

$q^T = [-1, -1, -1, \ldots, -1]$

$A = Y^T$

$b = [0]$

$P_{i,j} = y^{(i)} y^{(j)} x^{(j)T} x^{(j)}$

$h^T = [0, 0, \ldots, 0, C, C, \ldots, C]$

$$G = \begin{bmatrix} -1 & 0 & 0 & 0 & \ldots & 0 \\ 0 & -1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & -1 & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & \ldots & 0 & -1 \\ 1 & 0 & 0 & 0 & \ldots & 1 \\ 0 & 1 & 0 & 0 & \ldots & 1 \\ 0 & 0 & 1 & 0 & \ldots & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The condition $0 \leq \alpha_i \leq C$ is modelled using G. First m rows cover the $0 \leq \alpha_i$ part and second m rows cover $\alpha_i \leq C$

Parameters:

d=1, threshold = 0.0001, C=1.0

Support vectors: Indices of support vectors are saved in a file *support_vector_linear.txt*.

Weight = stored in the file

bias = 0.795

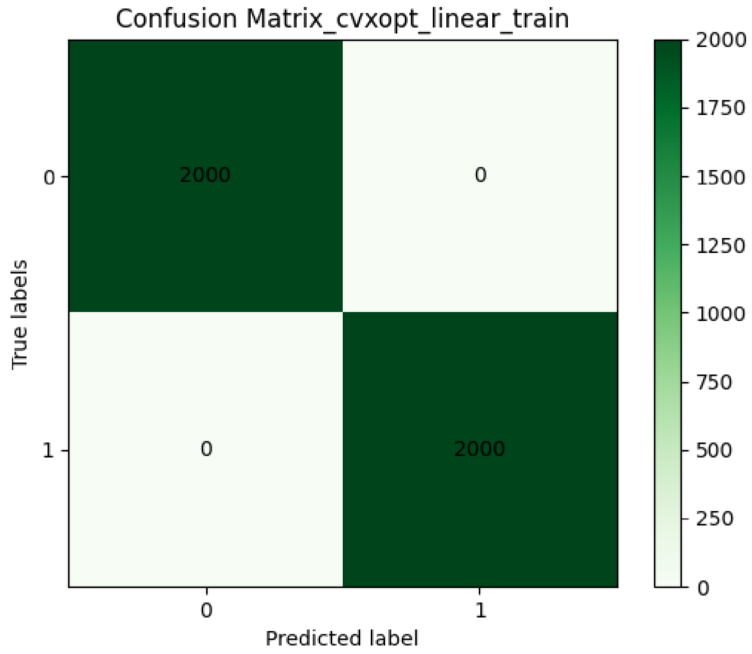
nSV = 95

Validation accuracy: 100%

Test accuracy: 98.89%

Computational cost: 8.87 seconds

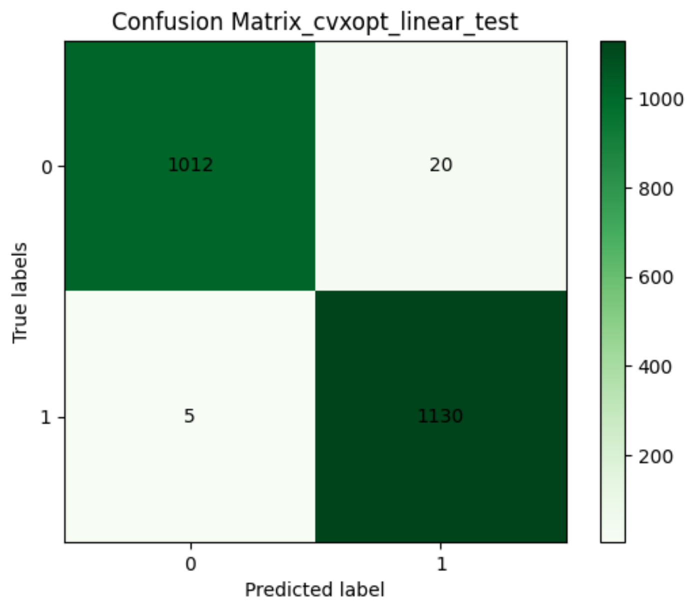
Confusion Matrix Train:



2000 0

0 2000

Confusion Matrix test:



1012 20

5 1130

---

ii) Gaussian Kernel:

Everything same as linear kernel except P

$P_{i,j} = y^{(i)} y^{(j)} K(x^{(j)}, x^{(j)})$

W is given by,

$$\sum_{i=1}^{m} \alpha_i y^i K\left(x^i, x\right)$$

b is given by,

$$\frac{1}{m}\sum_{i=1}^{m}\left(y^i - \sum_{j=1}^{m}\alpha_j y^j K\left(x^{(i)}, x^{(j)}\right)\right)$$

Parameters:

d=1, threshold = 0.0001, Gamma = 0.05, C=1.0

<u>Support vectors:</u> Indices of support vectors are saved in a file *support_vector_linear.txt*.

nSV = 867

Validation accuracy: 99.825%

Test accuracy: 95.846%

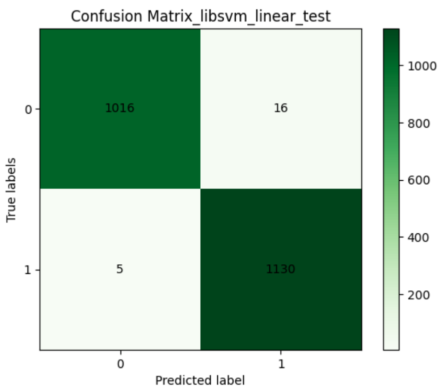
Computational cost: 28.100 sec

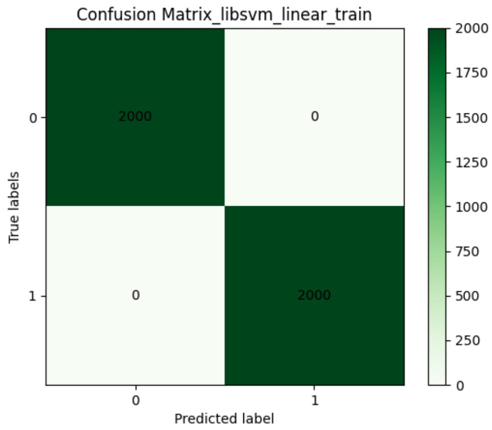
<u>iii) LIBSVM:</u>

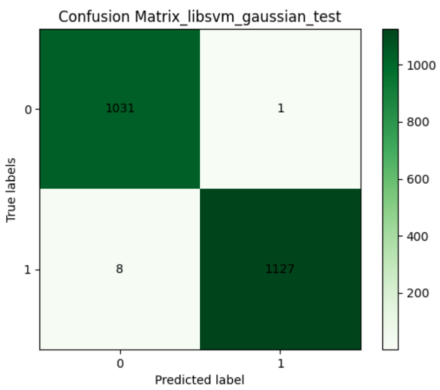
	Linear		Gaussian	
	CVXPOT	LIBSVM	CVXPOT	LIBSVM
Validation accuracy	100%	100%	99.825%	99.975%
Test accuracy	98.84%	99.03%	95.846%	99.58%
nSV	158	158	845	847
Bias	1.377	-1.219	-0.90	0.890
Computational cost	37.28	0.50	28.100	1.988

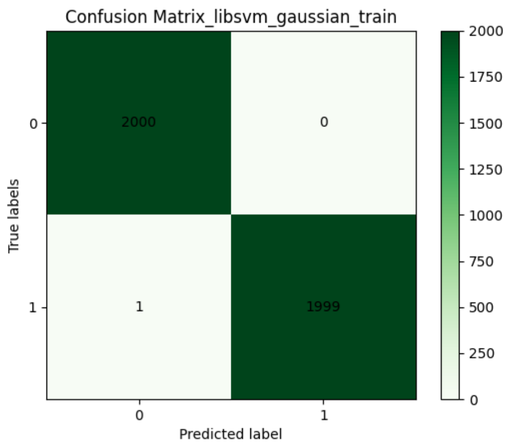
B value is almost negative due to representation difference.

W is vector and stored in the file provided









a) Multi-class classification:

i) CVXOPT:

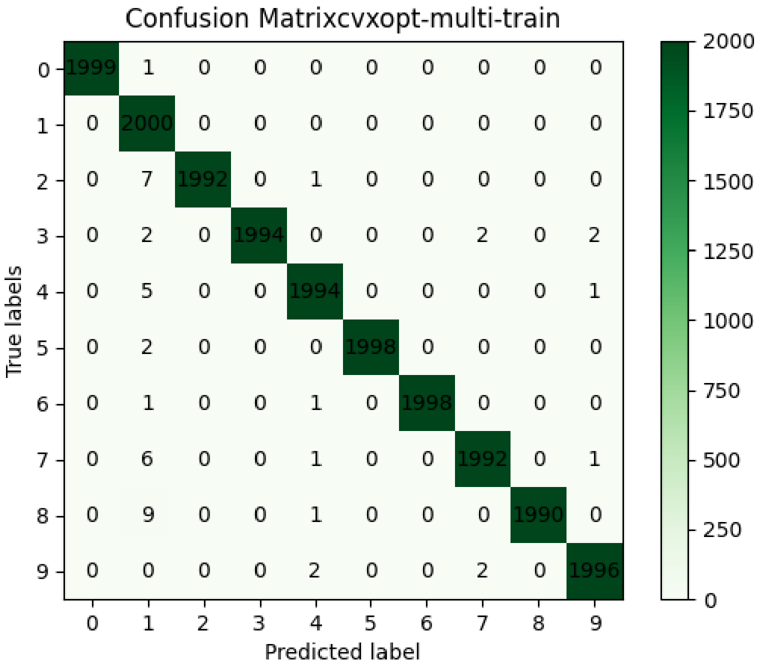
Total training time: 1198.8615338802338

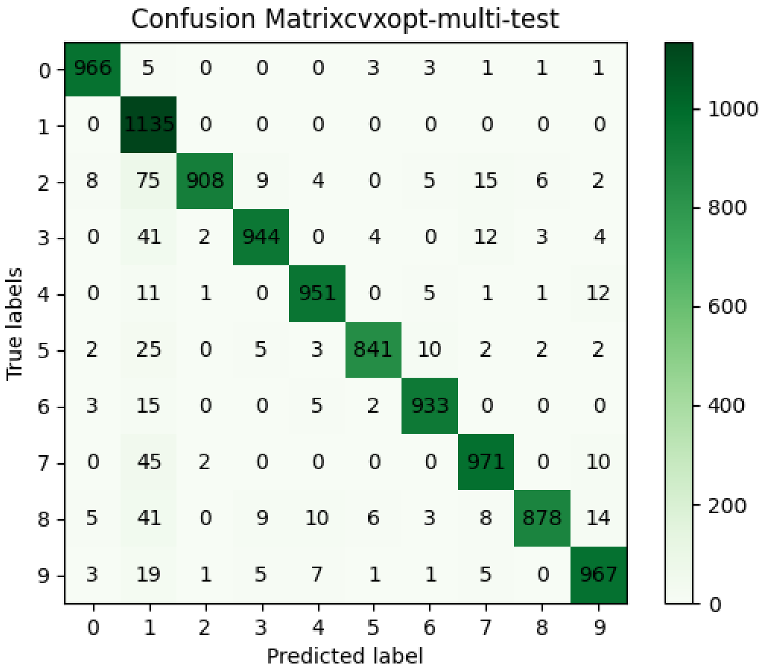
Prediction Time: 3 Hrs

Multiclass LIBSVM Training accuracy: 0.99765

Multiclass LIBSVM Test accuracy: 0.9494

Total nSV: 9866





ii) LIBSVM:

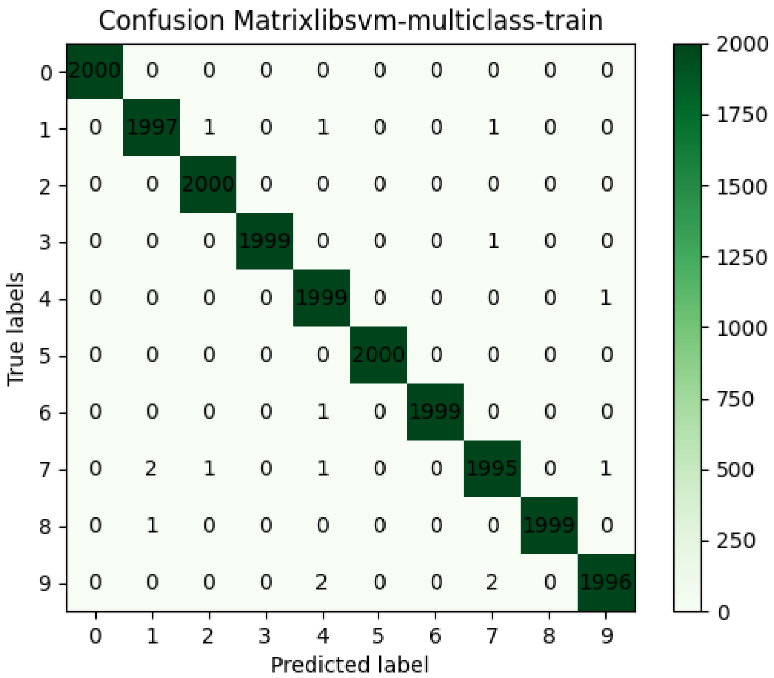
Training time: 170.986590385437

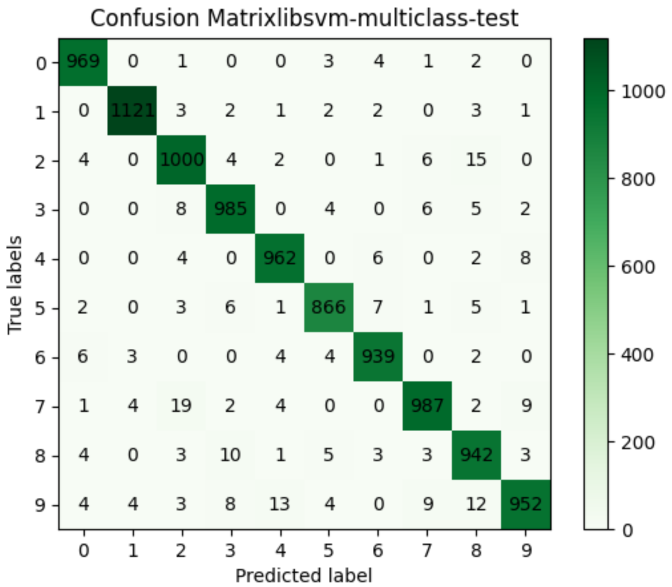
Total nSV: 10493

Train accuracy: 99.92% (19984/20000) (classification)

Test Accuracy: 97.23% (9723/10000) (classification)

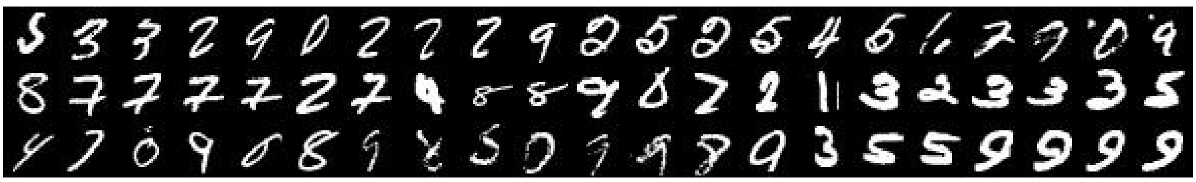
iii) Confusion matrices:



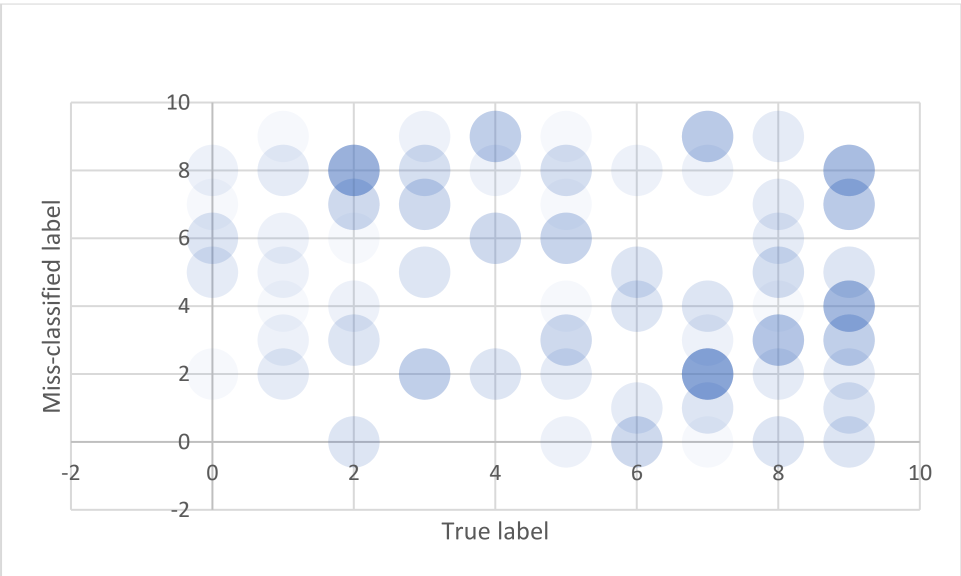


Missed train examples:

Missed test examples:



Miss-classified digits:



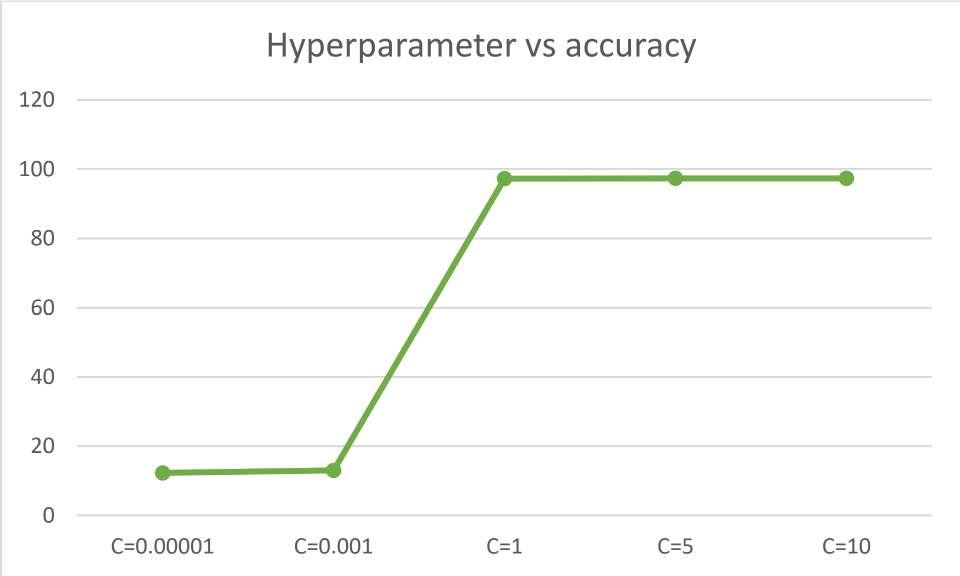
Most miss-classified are (7,2), (2,7), (8,3), (2,8), (9,4), (9,8), (9,4), (5,6), (6,0), (7,9)

Most of the results are intuitive as these are confusing to humans too.

Cross validation accuracy

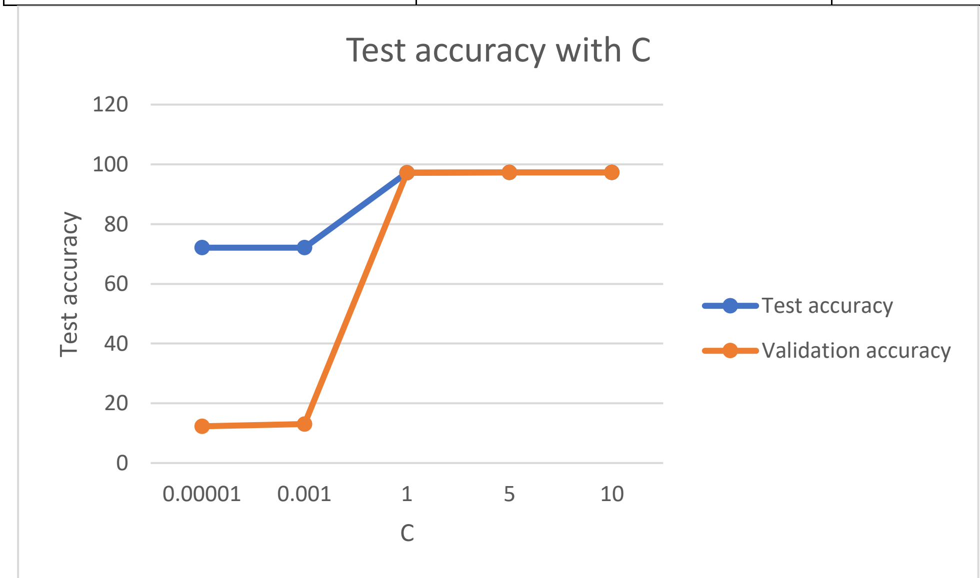
	C=0.00001	C=0.001	C=1	C=5	C=10
K1	9.325	9.325	97.125	97.2	97.2
K2	17.075	17.075	97.25	97.45	97.45
K3	9.275	9.275	97.625	97.65	97.65
K4	16.525	16.525	97.05	97.125	97.125

K5	9.275	9.275	97.1	97.175	97.175
Average	**12.295**	**13.05**	**97.23**	**97.32**	**97.32**



Test accuracy

C	Test accuracy	Validation accuracy
0.00001	12	12.295
0.001	72.1	13.05
1	97.23	97.23
5	97.29	97.32
10	97.29	97.32



Optimal parameters: 1,5,10

Low C leads to less weight to error and model is soft SVM hence underfitting

High C leads to more weight to error and hence overfitting with hard SVM