COL774: Machine Learning. Assignment 1

Arpan Mangal — 2016CS10321

February 2019

1 Text Classification (Naive Bayes)

Generate the vocabularies using *./run.sh v 1 2 3*.

1.1 Simple Naive Bayes

Computing the model parameters:

$$\phi_k = \frac{\mathbb{1}\{y^{(i)} = k\}}{m} \tag{1}$$

$$\theta_{wk} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n_i} \mathbb{1}\{y^{(i)} = k\}\mathbb{1}\{x^{(j)} = w\}}{\sum_{i=1}^{m} \mathbb{1}\{y^{(i)} = k\} n_i} \tag{2}$$

where, m = Size of Dataset, n_i = # Words in Document i.

- Training Accuracy: 62.89 %
- Testing Accuracy: 60.45 %
- Training dataset size: 535,000
- Testing dataset size: 134,000

1.2 Random and Majority Prediction

Majority Class: 5

- Random Prediction Accuracy: 19.93 %
- Majority Prediction Accuracy: 43.99 %

1.3 Confusion Matrix

Confusion Matrix for Simple NB are plotted in Figures 1 and 2.

Observations:

- The category (5, 5) has the highest value of the diagonal entry
- It shows which classes the model is confused distinguishing between.
- For instance, the model is mainly confused between distinguishing among 4 stars and 5 stars. This is understandable as both classes represent positive sentiment, and it's difficult to determine exact degree of (positivity which is subjective).

1.4 Stemming and Stop Words Removal

Stop words were removed and rest words stemmed using the NLTK library.

- Training Accuracy: 58.12 %
- Testing Accuracy: 54.34 %

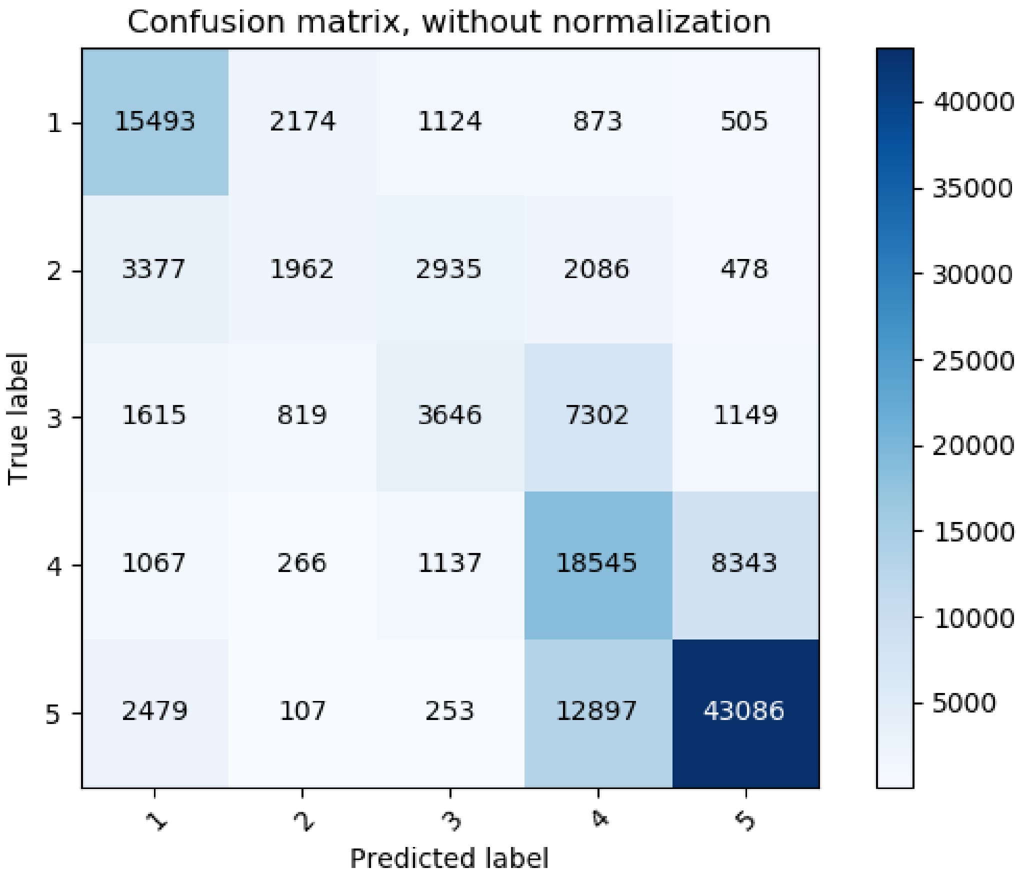


Figure 1: Confusion Matrix

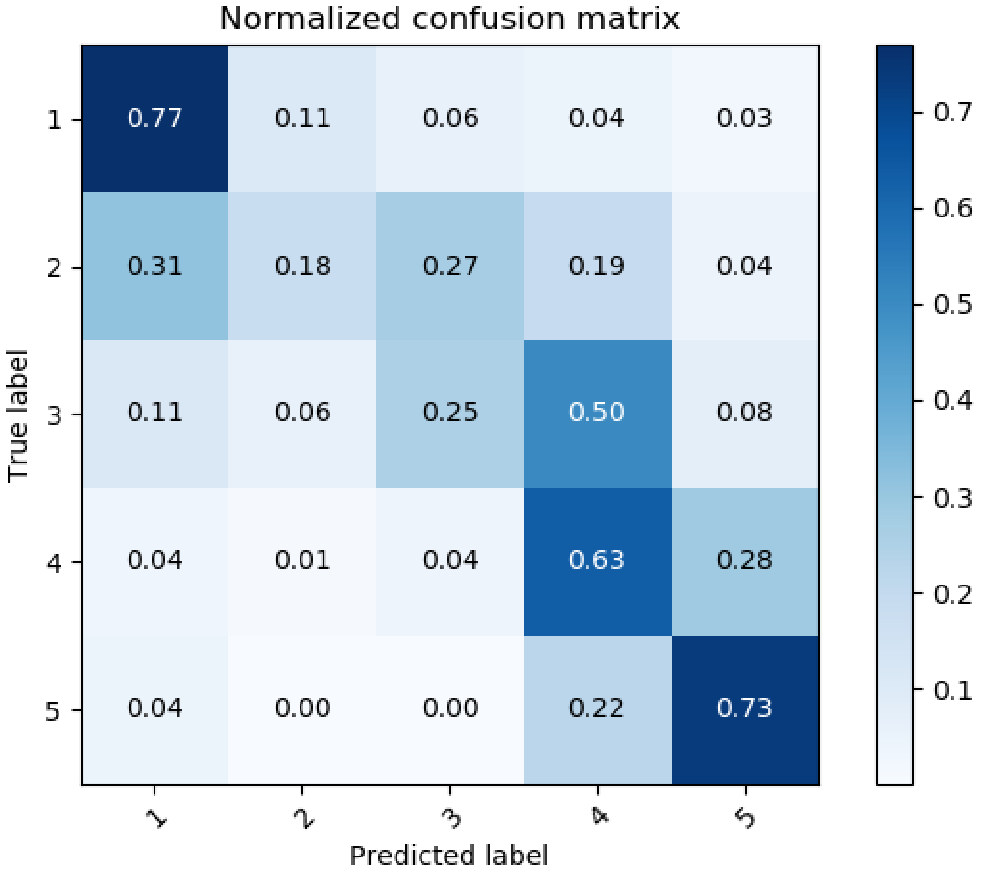


Figure 2: Normalised Confusion Matrix

- Training dataset size: 535,000
- Testing dataset size: 134,000

The Accuracy falls after doing Stemming and Stop-Words removal.

Usually stemming and stop-words removal are not good techniques for the case of Sentiment Mining. It is because some important sentiment words may be a part of the stop word list (as they are commonly occurring English words) and may be henceforth removed from the corpus. Similarly, different word forms may represent different sentiment or degree of sentiment, and stemming the words may lead to losing that sentiment information.

1.5 Feature Engineering

Following Features were tried:

- Adding Bi-grams gave good increase in accuracy.
- Adding Tri-grams gave marginal improvement over Bi-grams, at the cost of increased computation, so were removed.

- There are some words which occur too often, and don't contribute to the model. Removing these words is better then removing stop words, since these words are like in domain stop-words.
- Some words just occur a few times and do not contribute to the model. Such words are often not part of actual English and may be some random word or grammatical errors made by people. These were removed.
- Finally, the size of the dictionary was capped, so as to improve computation time.

- Training Accuracy: 64.56 %
- Testing Accuracy: 61.20 %
- Training dataset size: 535,000
- Testing dataset size: 134,000

1.6 F1 Scores

Class	**F1 Score**
1	0.701
2	0.243
3	0.309
4	0.522
5	0.767

- Macro F1 score:0.508

For multi-class classification settings like handwriting recognition, Accuracy may not be a sufficient measure. It also becomes important to see the amount of data which should be in a particular class but is not, i.e. focus on Recall along with the precision.

1.7 Full Data Set

- Training Accuracy: 79.48 %
- Testing Accuracy: 72.34 %
- Training dataset size: 5,350,000
- Testing dataset size: 1,340,000

2 Support Vector Machines

2.1 Binary Classification

For my entry number *2016CS10321*, $d = 1$ and $d + 1 = 2$.

2.1.1 Linear Kernel

The SVM dual objective is written as:

$$maxW(\alpha) = \sum_i \alpha(i) - \frac{1}{2}\sum_{i,j} y^{(i)}y^{(j)}\alpha^{(i)}\alpha^{(j)}\langle x^{(i)}, x^{(j)}\rangle \tag{3}$$

which can be written in the form:

$$minW(\alpha) = \alpha^T Q\alpha + b^T\alpha \tag{4}$$

where, $Q_{ij} = y^{(i)}y^{(j)}\langle x^{(i)}, x^{(j)}\rangle$ and $b = (-1, ..., -1)_{1\times m}$
Other Constraints:

$$\sum_i \alpha^{(i)}y^{(i)} = 0 \tag{5}$$

$$0 < \alpha^{(i)} < 1 \tag{6}$$

- Number of support vectors: 157 (stored in *Q2/support-vectors/linear.vectors*)
- Training Accuracy: 97.4 %
- Test Accuracy: 97.8 %
- Time Taken: 73.51 secs

2.1.2 Gaussian Kernel

- Number of support vectors: 827 (stored in *Q2/support-vectors/gaussian.vectors*)
- Training Accuracy: 99.997 %
- Test Accuracy: 99.538 %
- Time Taken: 59.96 secs

2.2 LIBSVM

Linear

- Number of support vectors: 158
- Training Accuracy: 100 %
- Test Accuracy: 99.03 %

- Time Taken: 1.5 secs

Gaussian

- Number of support vectors: 846
- Training Accuracy: 99.97 %
- Test Accuracy: 99.58 %
- Time Taken: 3.6 secs

Time taken is much less than compared to CVXOPT. Accuracy is same for gaussian kernel, while it is slightly more for linear kernel as compared to CVXOPT.

2.3 Multi-Class Classification

2.3.1 CVXOPT

First we train C_2^{10} classifiers to distinguish between all pairs of numbers, and during prediction time , we output the classifier which has the maximum number of wins among all the C_2^{10} classifiers.

- Test Accuracy: 96.89
- Time Taken: 9,390 secs

2.3.2 LIBSVM

- Test Accuracy: 97.24 %
- Time Taken: 1,683 secs

2.4 Confusion Matrix

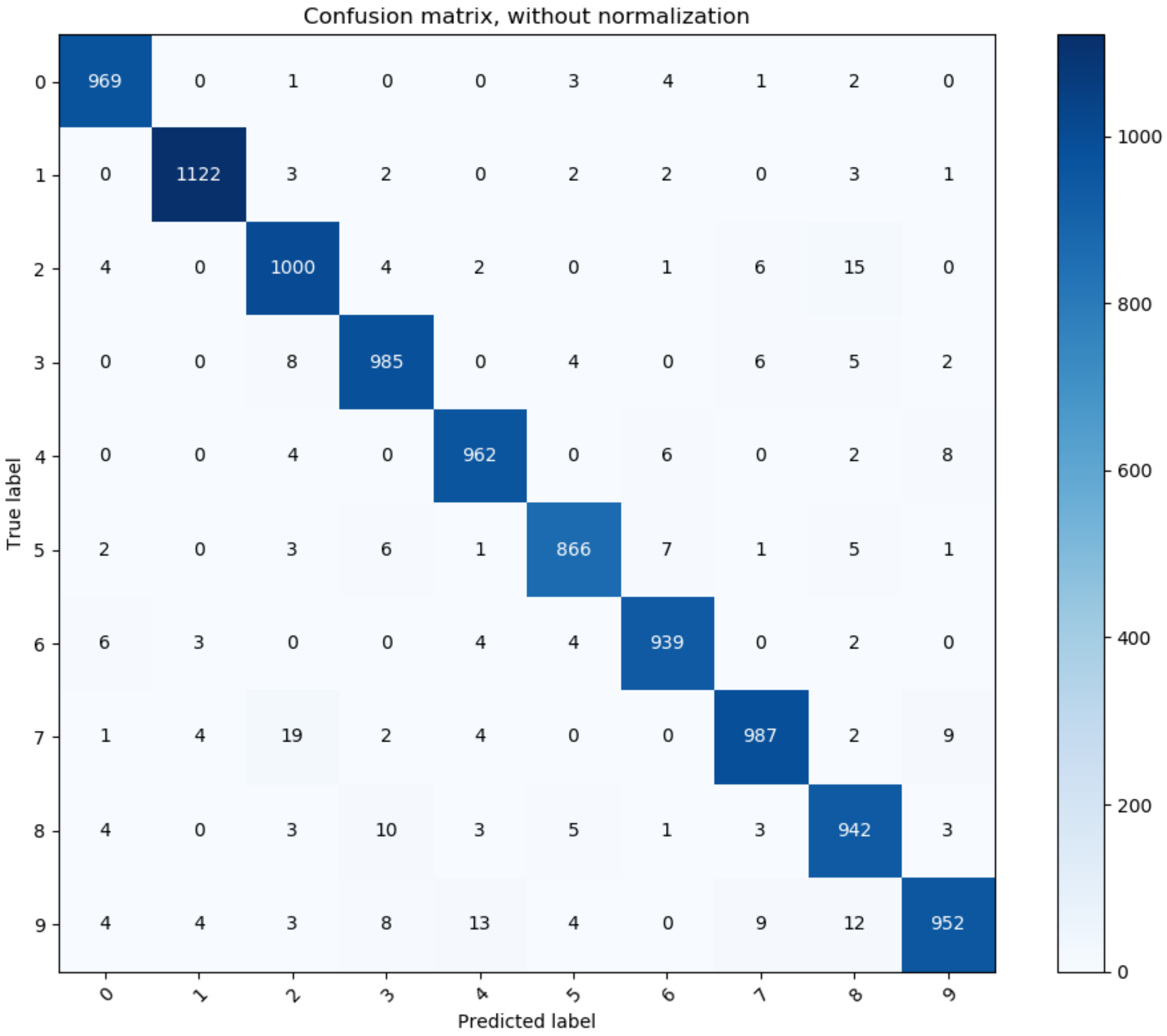
The Confusion Matrix is shown in Figures 3 and 4

Observations:

- Most digits are correctly classified.
- Some instances of mis-classification include 9 being classified as 4 or vice versa, or 8 being classified as 3.
- Yes, the results make sense. Most of them are correct, and the most of the errors like 9 vs. 4 are acceptable.

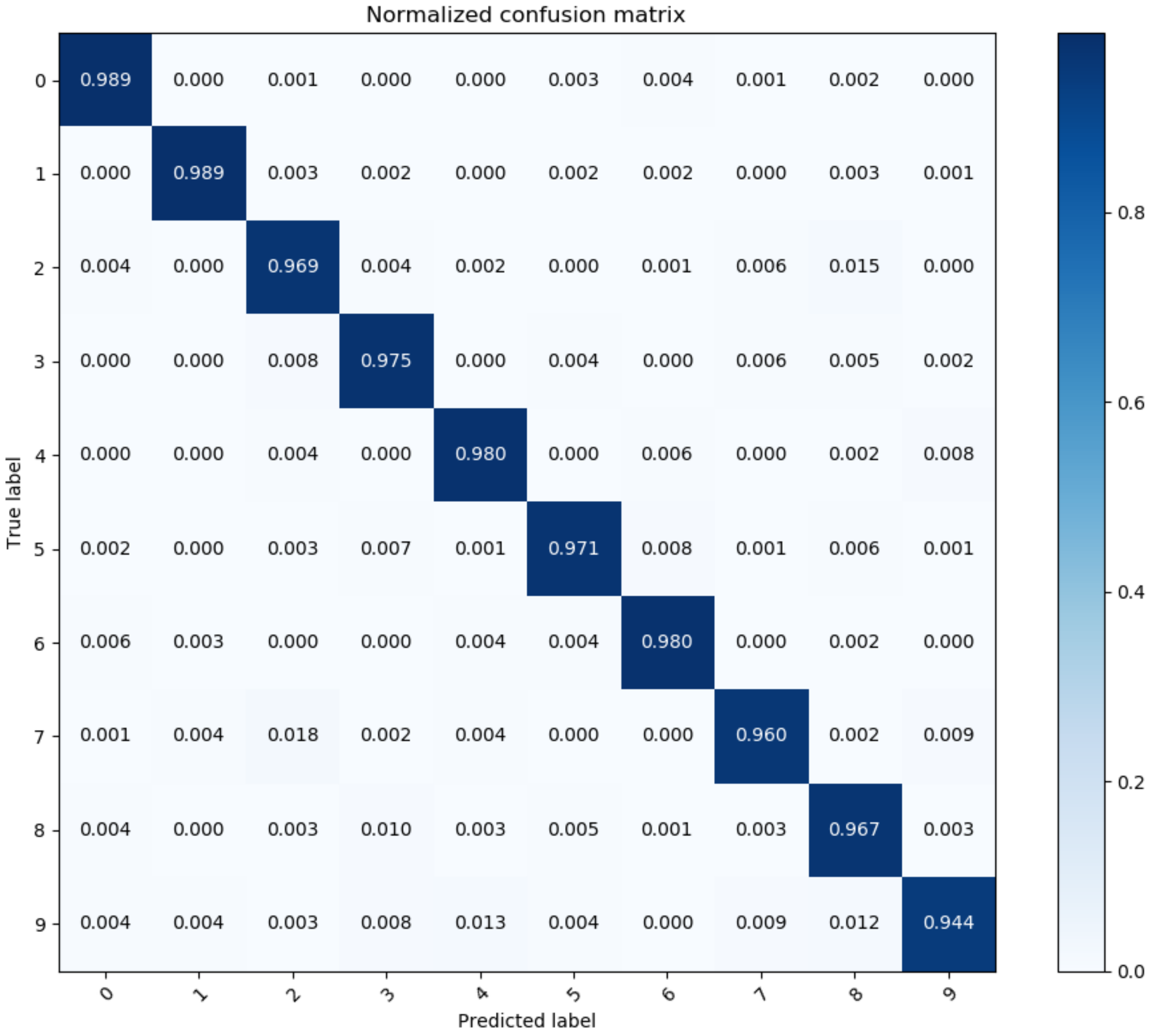
2.5 Validation

The LIBSVM model of *2b* was used with different values of C in the set $[\{10^{-5}, 10^{-3}, 1, 5, 10]$. The results are shown in Figures 5 and 6.



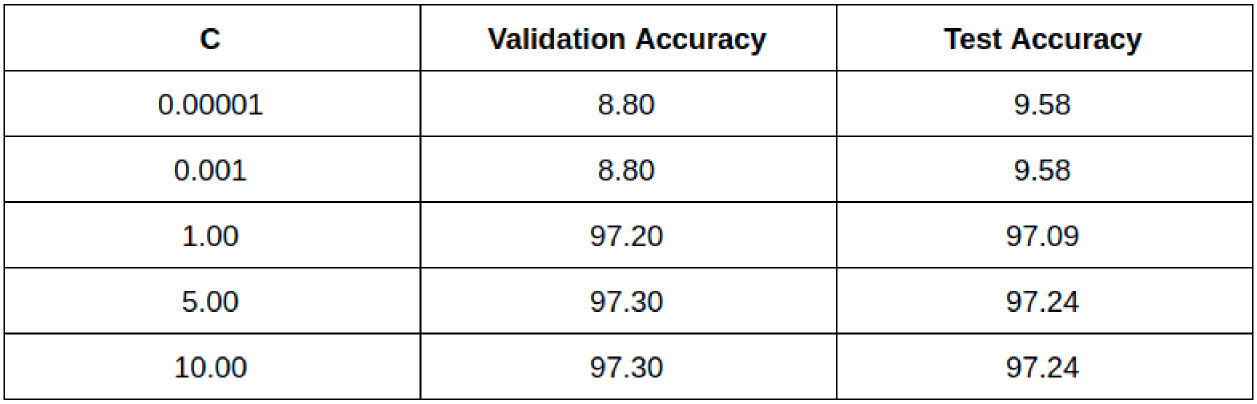
	0	1	2	3	4	5	6	7	8	9
0	969	0	1	0	0	3	4	1	2	0
1	0	1122	3	2	0	2	2	0	3	1
2	4	0	1000	4	2	0	1	6	15	0
3	0	0	8	985	0	4	0	6	5	2
4	0	0	4	0	962	0	6	0	2	8
5	2	0	3	6	1	866	7	1	5	1
6	6	3	0	0	4	4	939	0	2	0
7	1	4	19	2	4	0	0	987	2	9
8	4	0	3	10	3	5	1	3	942	3
9	4	4	3	8	13	4	0	9	12	952

Figure 3: Confusion Matrix



	0	1	2	3	4	5	6	7	8	9
0	0.989	0.000	0.001	0.000	0.000	0.003	0.004	0.001	0.002	0.000
1	0.000	0.989	0.003	0.002	0.000	0.002	0.002	0.000	0.003	0.001
2	0.004	0.000	0.969	0.004	0.002	0.000	0.001	0.006	0.015	0.000
3	0.000	0.000	0.008	0.975	0.000	0.004	0.000	0.006	0.005	0.002
4	0.000	0.000	0.004	0.000	0.980	0.000	0.006	0.000	0.002	0.008
5	0.002	0.000	0.003	0.007	0.001	0.971	0.008	0.001	0.006	0.001
6	0.006	0.003	0.000	0.000	0.004	0.004	0.980	0.000	0.002	0.000
7	0.001	0.004	0.018	0.002	0.004	0.000	0.000	0.960	0.002	0.009
8	0.004	0.000	0.003	0.010	0.003	0.005	0.001	0.003	0.967	0.003
9	0.004	0.004	0.003	0.008	0.013	0.004	0.000	0.009	0.012	0.944

Figure 4: Normalised Confusion Matrix



C	Validation Accuracy	Test Accuracy
0.00001	8.80	9.58
0.001	8.80	9.58
1.00	97.20	97.09
5.00	97.30	97.24
10.00	97.30	97.24

Figure 5: Validation & Test Accuracies

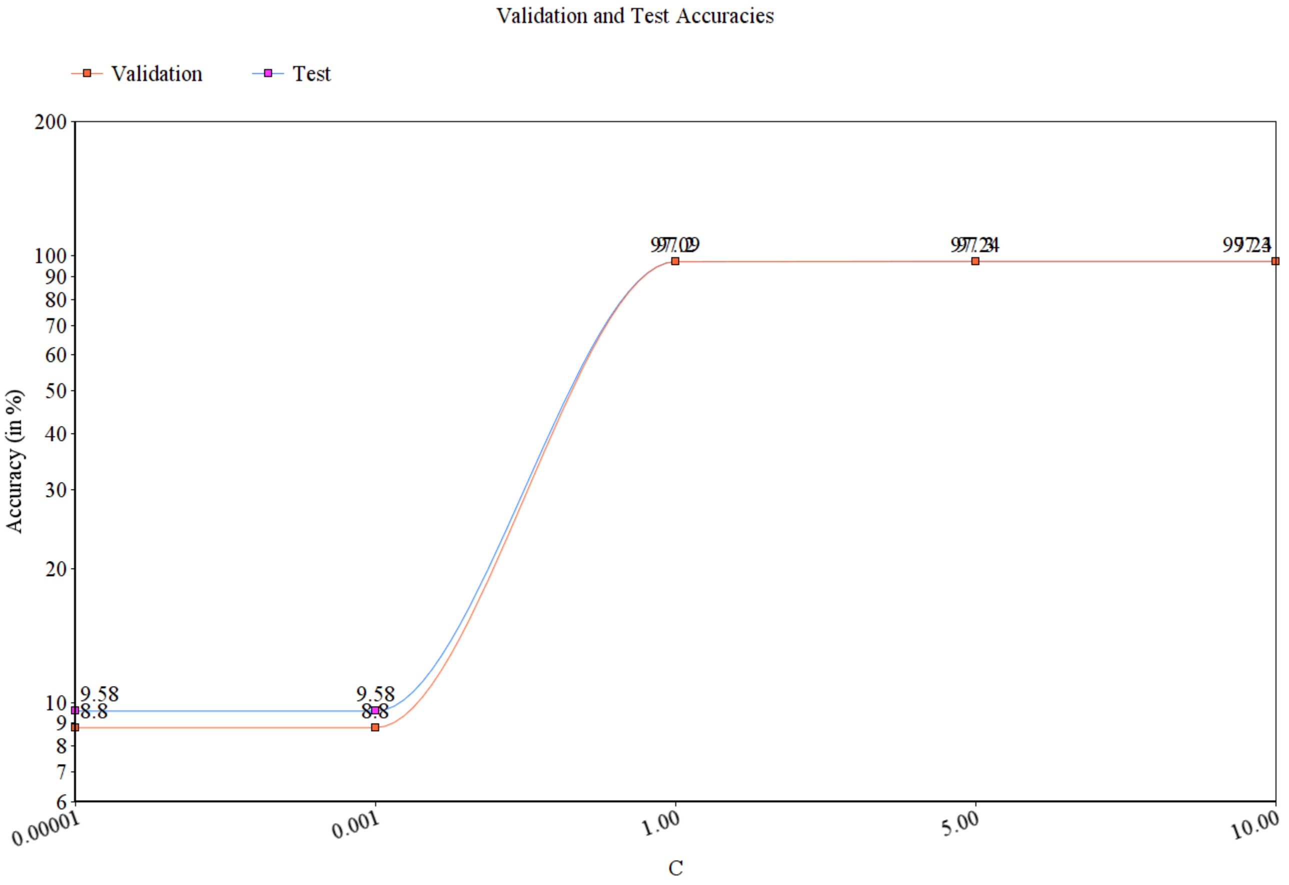


Figure 6: Validation & Test Accuracies