# ML ASSIGNMENT-2

**Q.1**

**(a) Training Accuracy-** 67.576 %
**Test Accuracy-** 38.688 %

**(b) Random Guessing:**

- **Training Accuracy-** 8.892 %
- **Test Accuracy-** 8.872 %

**Majority prediction:**

- **Training Accuracy-** 20.4 %
- **Test Accuracy-** 20.088 %

The naive bayes algorithm performs approximately 3.02 times better on training data and 1.9 times better on test data. Clearly random guessing is the worst option for this data. The majority prediction has a very less accuracy as well.

**(c) Confusion matrix:**

| | | Predicted Class | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 7 | 8 | 9 | 10 |
| Actual Class | 1 | 4490 | 17 | 67 | 184 | 33 | 59 | 5 | 167 |
| | 2 | 1774 | 19 | 100 | 200 | 43 | 49 | 1 | 116 |
| | 3 | 1575 | 21 | 124 | 402 | 98 | 138 | 4 | 179 |
| | 4 | 1221 | 16 | 118 | 571 | 168 | 262 | 11 | 268 |
| | 7 | 485 | 4 | 33 | 208 | 323 | 546 | 17 | 691 |
| | 8 | 477 | 7 | 32 | 126 | 214 | 699 | 40 | 1255 |
| | 9 | 362 | 2 | 13 | 60 | 87 | 435 | 41 | 1344 |
| | 10 | 850 | 3 | 20 | 65 | 111 | 497 | 48 | 3405 |

We can clearly observe that the original class and predicted class match maximum only for labels 1 and 10. This means that the algorithm is able to predict extremely good and extremely bad rating but it gets confused otherwise. For low ratings 2,3 and 4, it confuses them with 1 while for high ratings 7,8,9 it confuses them with 10. This is the possible reason for the accuracy on test data being less than 50%.
The class ‘1’ has the highest value of the diagonal-entry in the confusion matrix. Followed by second highest ‘10’. This is again because a lot of training data has 1 (9000+ examples) as a label while others have ‘10’ (4000+ examples).
The class ‘3’ is confused with ‘1’ maximum number of times (61.98%).

**(d) Stopword removal and stemming:**

Training accuracy-

- **Naive bayes-** 67.996 %
- **Random Guessing-** 9.028 %
- **Majority prediction-** 20.4%

Test accuracy-

- **Naive bayes-** 38.425 %

- **Random Guessing-** 9.024 %
- **Majority prediction-** 20.088%

Confusion matrix for test data:

| | | Predicted Class | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 7 | 8 | 9 | 10 |
| Actual Class | 1 | 4156 | 128 | 193 | 248 | 43 | 70 | 37 | 147 |
| | 2 | 1502 | 78 | 204 | 283 | 64 | 44 | 13 | 114 |
| | 3 | 1267 | 106 | 274 | 502 | 115 | 106 | 22 | 149 |
| | 4 | 925 | 76 | 294 | 666 | 215 | 206 | 39 | 214 |
| | 7 | 340 | 25 | 111 | 289 | 407 | 497 | 121 | 517 |
| | 8 | 356 | 30 | 93 | 208 | 322 | 692 | 211 | 938 |
| | 9 | 281 | 18 | 48 | 116 | 163 | 432 | 165 | 1121 |
| | 10 | 596 | 34 | 69 | 139 | 207 | 534 | 245 | 3175 |

We see that after stopword removal and stemming, the overall accuracy for test data does not increase but the confusion matrix tells us that less ‘2’, ‘3’ and ‘4’ labels are confused with label ‘1’ like in previous execution (without stopword removal and stemming) and the value of diagonal entries for ‘2’, ‘3’ and ‘4’ increase as compared to the previous results. Similarly less ‘7’, ‘8’ and ‘9’ labels are confused with label ‘10’ like in previous execution (without stopword removal and stemming) and the value of diagonal entries for ‘7’, ‘8’ and ‘9’ increase as compared to the previous results. Yet the overall accuracy decreases as in this case less number of ‘1’ and ‘10’ labels have been correctly classified. This clearly shows that the model has learnt to classify the reviews into the non-extreme ratings as well (like ‘2’, ‘3’, ‘4’, ‘7’, ‘8’, ‘9’).

**(e) Feature engineering:**

- **Bigram features (best)-**

Treating 2 consecutive words as features. The test accuracy I get is 39.336 % which is greater than in both the previous cases. The confusion matrix for this case is:

| | | Predicted Class | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 7 | 8 | 9 | 10 |
| Actual Class | 1 | 4397 | 134 | 123 | 110 | 34 | 39 | 18 | 167 |
| | 2 | 1729 | 123 | 118 | 124 | 24 | 32 | 18 | 134 |
| | 3 | 1576 | 122 | 255 | 246 | 80 | 68 | 23 | 171 |
| | 4 | 1268 | 94 | 224 | 481 | 148 | 128 | 47 | 245 |
| | 7 | 454 | 35 | 93 | 168 | 396 | 340 | 94 | 727 |
| | 8 | 411 | 42 | 72 | 136 | 269 | 495 | 161 | 1264 |
| | 9 | 297 | 26 | 35 | 73 | 126 | 273 | 167 | 1347 |
| | 10 | 620 | 52 | 50 | 66 | 158 | 298 | 235 | 3520 |

We see fairly more number of ‘2’, ‘3’, ‘4’, ‘7’, ‘8’, ‘9’ labels correctly classified without compromising the classification of reviews ‘1’ and ‘10’ correctly, thus an overall increase in accuracy.

- **Assigning weights to good features-**

Finding some good words from the dataset associating more weights to them and removing very common and non deciding words from the vocabulary. This increased the training accuracy to 70.608 % as expected but the training accuracy was reduced to 34.704 %. This was because in the test dataset the good words from training might not be significant to determining the rating of the movie without context.

**Q.2**

**(b) Training accuracy-** 95.075%
**Test accuracy-** 93.96

**(c) Linear-**

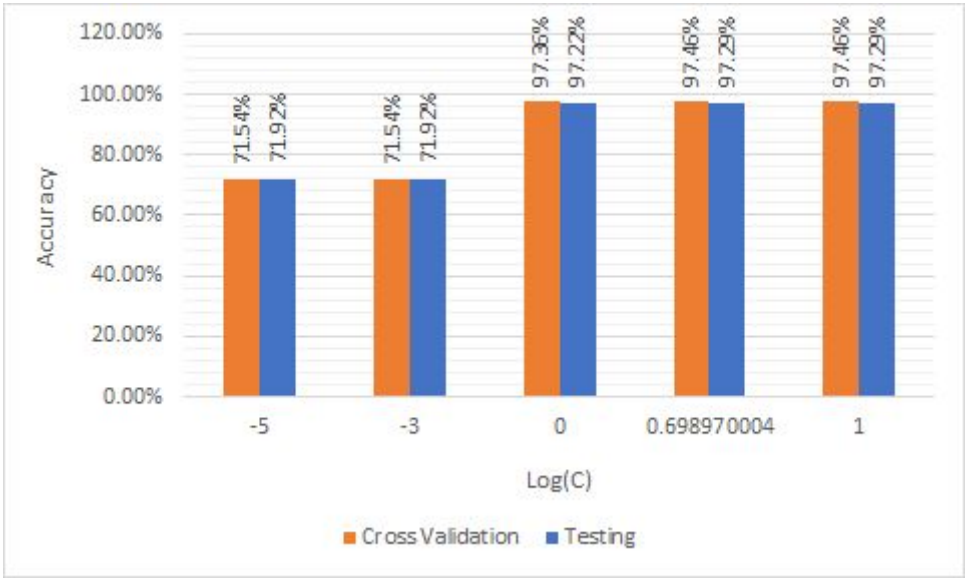
- **Training accuracy-** 98.785%
- **Test accuracy-** 92.76%

**Gaussian-**

- **Training accuracy-** 99.92%
- **Test accuracy-** 97.22%

We see that the accuracies obtained using libsvm are close to that obtained from pegasos and hence solving the primal objective is essentially the same as solving the dual objective.

**(d)**




We observe that the cross validation accuracies and testing accuracies are same for C=0.00001 and C=0.001 (71.54% and 71.92% respectively) and for C= 10 and C=5 as well.

C=10 and C=5 give the best cross-validation accuracy (97.355%) and the same Cs give the best test accuracies (97.29%). Essentially the accuracy drastically increases with the value of C. This shows that cross validation is a good method for optimising the model parameters. This is because as the value of C increases the outliers decrease.

**(e) Confusion matrix-**

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

Since all the diagonal entries are in the range 800-1100 we observe that the gaussian model corresponding to C=10 serves as a robust classifier because it classifies majority of the data precisely. Looking at the off diagonal entries, we find that class 7 is most difficult to classify with 20 incorrect classifications as 2. One reason is because 2 and 7 have maximum similarity among other digits.
Other misclassified examples are-
3 misclassifed as 2

2 misclassified as 8

4 misclassified as 6