COL774 Assignment 2

Rishit Singla

October 2023

1 Text Classification

(a)

Training Accuracy: 89.90%
Validation Accuracy: 67.48%
Word Clouds:

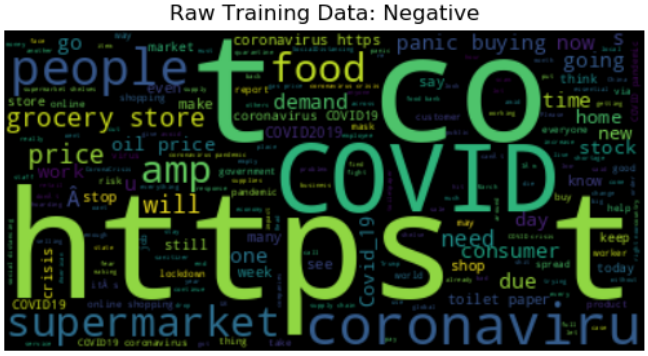


Figure 1: WordCloud (Raw Training Data, Negative Sentiment)

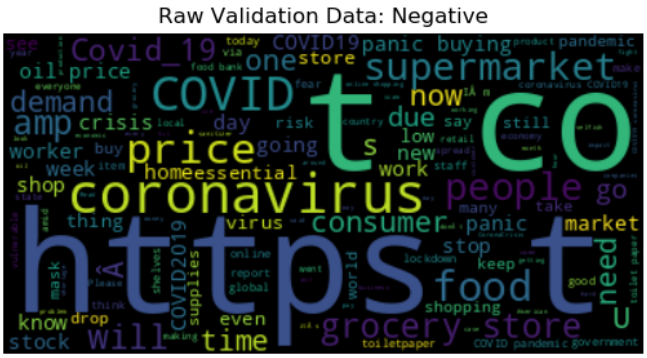


Figure 2: WordCloud (Raw Validation Data, Negative Sentiment)

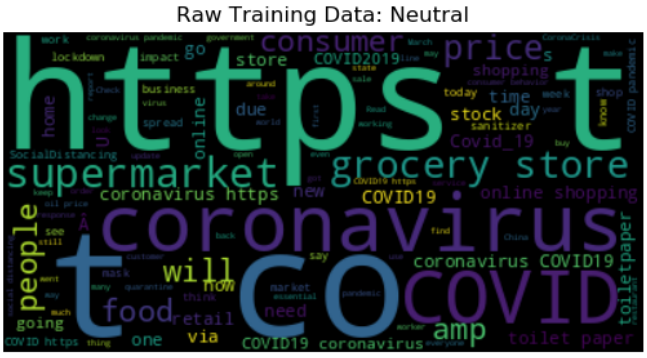


Figure 3: WordCloud (Raw Training Data, Neutral Sentiment)

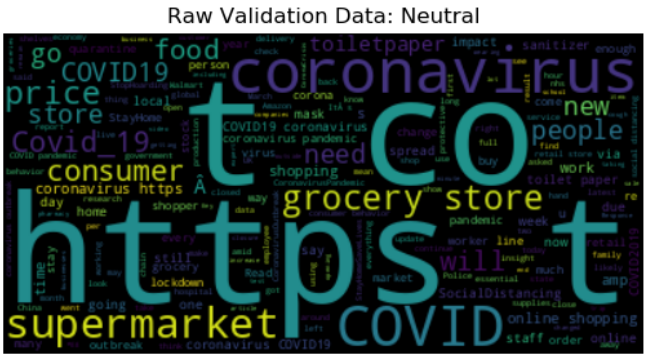


Figure 4: WordCloud (Raw Validation Data, Neutral Sentiment)

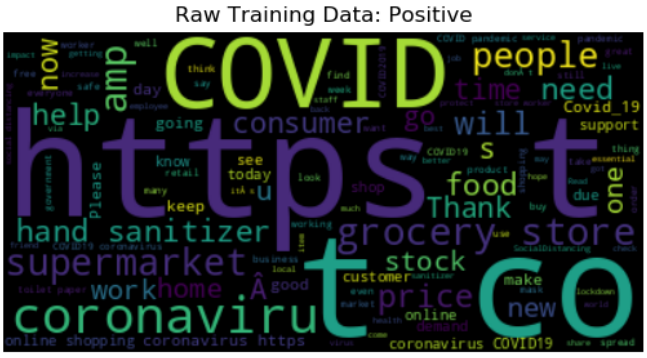


Figure 5: WordCloud (Raw Training Data, Positive Sentiment)

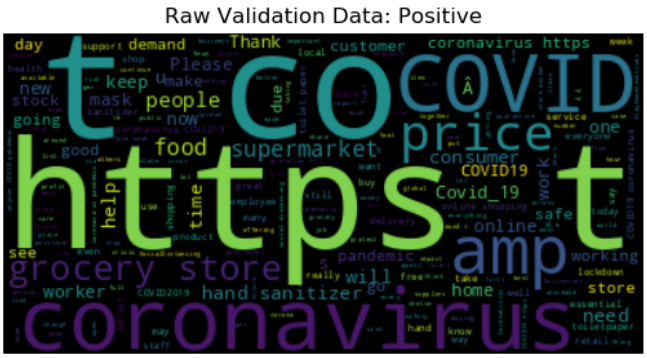


Figure 6: WordCloud (Raw Validation Data, Positive Sentiment)

(b)

Random Guessing Model Validation Accuracy: 33.37%
Positive Predicting Model Validation Accuracy: 43.85%
Our Algorithm has Validation Accuracy of 67.48% which is 34.11% more than random guessing and 23.63% more than positive predicting model. Thus our algorithm shows quite a significant improvement over these random/positive baseline.

(c)

Random Model

In Random Model, the maximum diagonal entry is for Positive Label for both Training and Validation data. Since it randomly classifies data, it classifies approximately equal proportion of available data correctly for each label. Since largest amount of data is available for positive labels, it has maximum diagonal entry in Positive label.

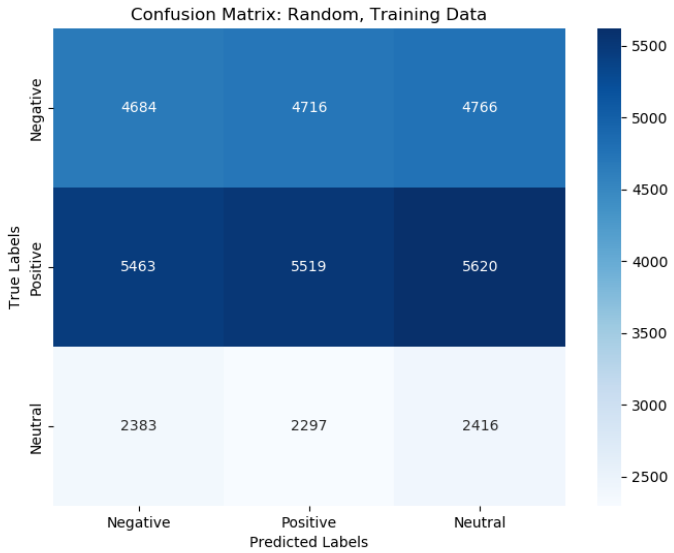


Figure 7: Confusion Matrix: Random, Training Data

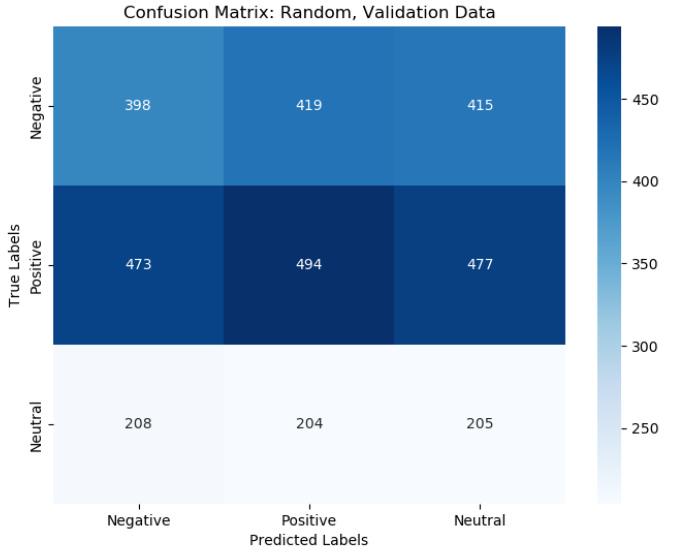


Figure 8: Confusion Matrix: Random, Validation Data

Positive Predicting Model

In Positive Predicting Model, the maximum diagonal entry is for Postive Label for both Training and Validation data. Reason is trivial, since it predicts every tweet as positive.

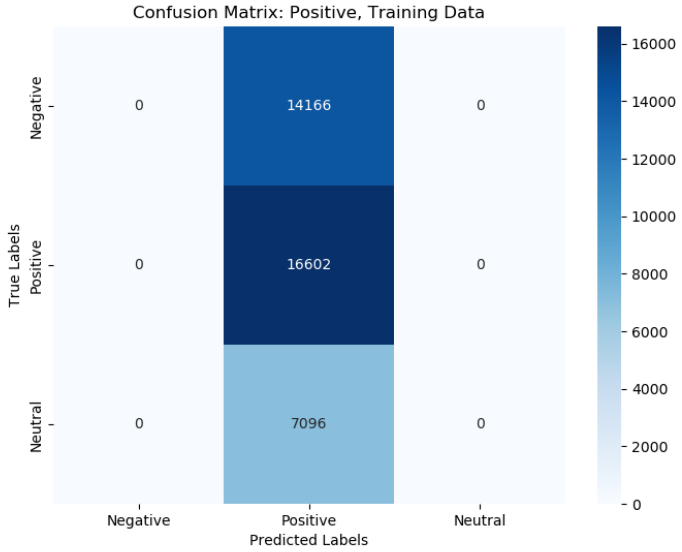


Figure 9: Confusion Matrix: Positive, Training Data

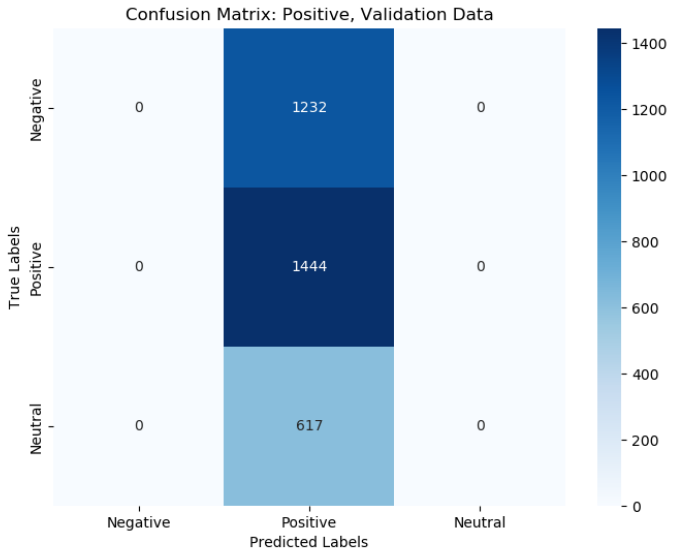


Figure 10: Confusion Matrix: Positive, Validation Data

Naive Bayes Model

In Multi Class Naive Bayes Model, the diagonal entries are quite large as compared to off diagonal entry showing high prediction accuracy. Since, largest amount of data is available for Positive label, here also max diagonal entry is for Positive label

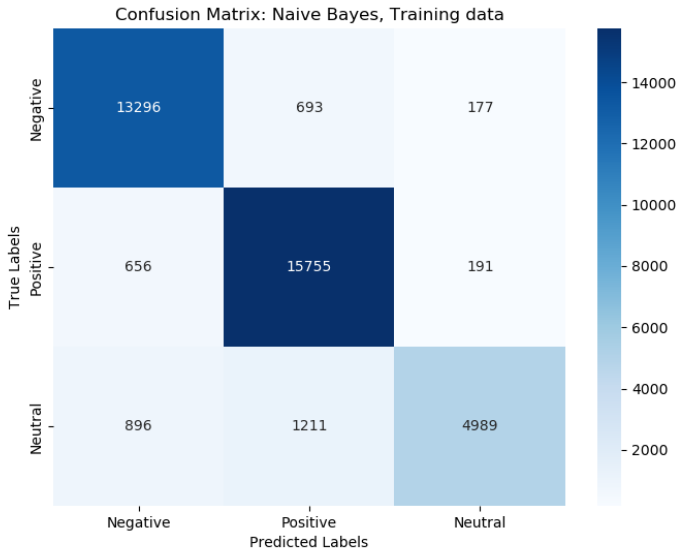


Figure 11: Confusion Matrix: Naive Bayes, Training Data

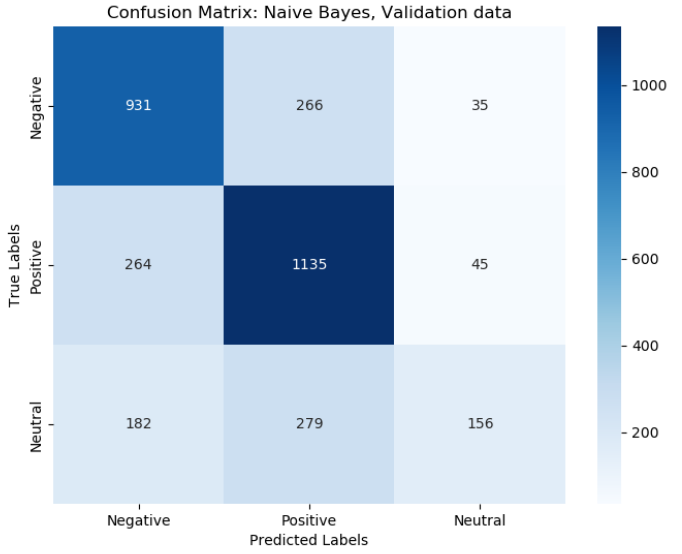


Figure 12: Confusion Matrix: Naive Bayes, Validation Data

(d)

Training Accuracy: 87.55%
Validation Accuracy: 69.54%

The accuracy over validation set increased by 2.06% on stemming and stop word removal. Thus we observe that on stemming and stop word removal, the accuracy over validation set increases.

WordClouds:

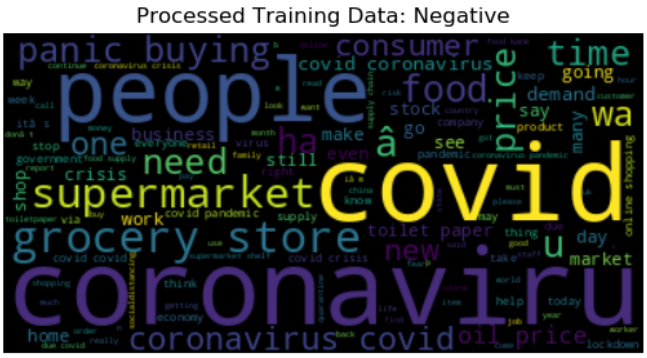


Figure 13: WordCloud (Processed Training Data, Negative Sentiment)

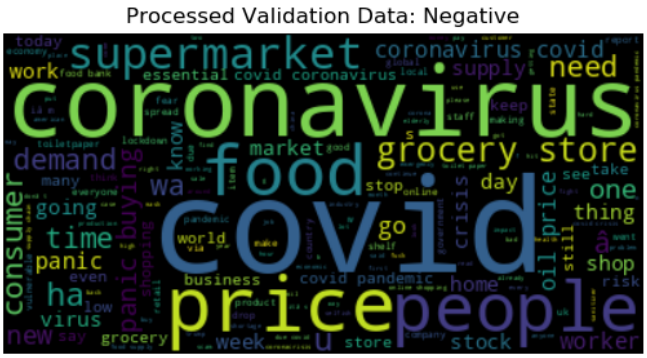


Figure 14: WordCloud (Processed Validation Data, Negative Sentiment)

Figure 15: WordCloud (Processed Training Data, Neutral Sentiment)

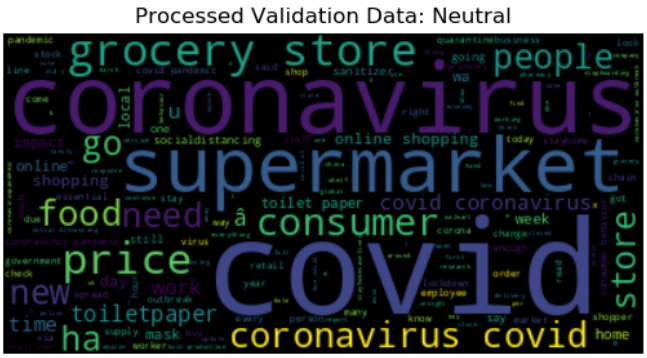


Figure 16: WordCloud (Processed Validation Data, Neutral Sentiment)

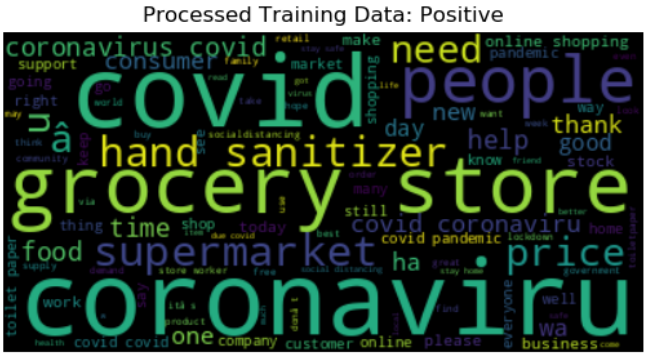


Figure 17: WordCloud (Processed Training Data, Positive Sentiment)

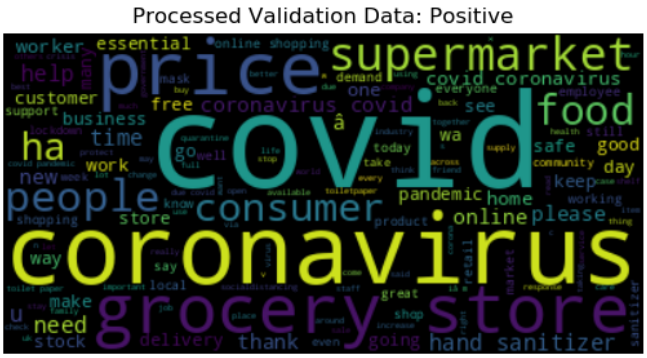


Figure 18: WordCloud (Processed Validation Data, Positive Sentiment)

(e)

Bigram Feature

Training Accuracy: 97.30%
Validation Accuracy: 70.51%

The Validaton Accuracy here is more than (a) by 3.03% and more than (d) by 0.97%. Thus the validation accuracy increased by using additional feature of bigram.

Addition Feature: Length of tweet

The additional feature used is the length of tweet. Length is used as an additional feature to improve the model.
Training Accuracy: 88.21%
Validation Accuracy: 71.54%

The Validation Accuracy here is more than (a) by 4.06% and more than (d) by 2.00%. Thus the validation accuracy increased by using additional feature of tweet length.

Bigram Feature + Tweet Length Feature

Here, both feature are incorporated together.
Training Accuracy: 96.75%
Validation Accuracy: 71.94%

The Validation Accuracy here is more than (a) by 4.46% and more than (d) by 2.40%. Thus the validation accuracy increased by using additional feature of bigram and tweet length. Also it is more than the accuracy obtained by using only one additional feature at a time.

Hence, the additional features helped to increase validation accuracy in all three cases.

(f)

1%

Validation Accuracy (Source + Target): 43.27%
Validation Accuracy (Target): 36.15%

2%

Validation Accuracy (Source + Target): 44.23%
Validation Accuracy (Target): 38.17%

5%

Validation Accuracy (Source + Target): 45.03%
Validation Accuracy (Target): 42.94%

10%

Validation Accuracy (Source + Target): 45.73%
Validation Accuracy (Target): 44.60%

25%

Validation Accuracy (Source + Target): 48.28%
Validation Accuracy (Target): 44.47%

50%

Validation Accuracy (Source + Target): 50.56%
Validation Accuracy (Target): 47.71%

100%

Validation Accuracy (Source + Target): 53.35%
Validation Accuracy (Target): 51.89%

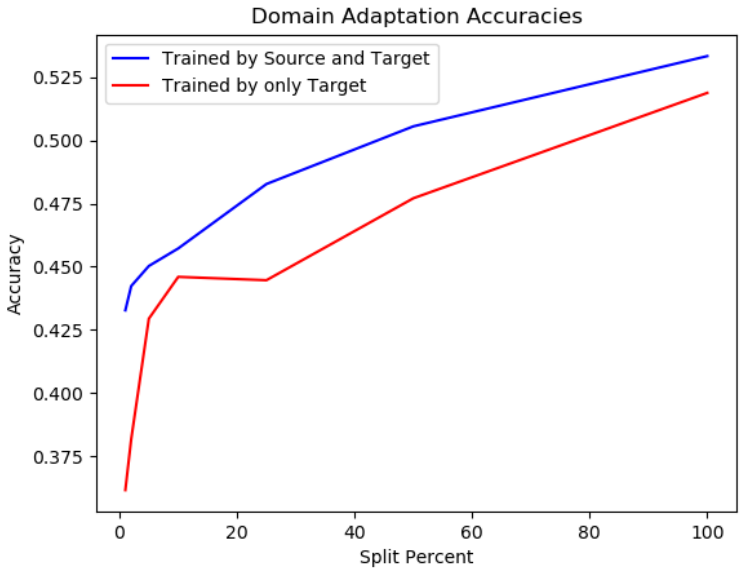


Figure 19: Domain Adaptation Accuracy

We observe that as the amount of target data increases, the accuracy of both the algorithm increases. This is simply because of increase in amount of training data. The accuracy for training with source + target is more than the accuracy for training only with target. This is also because of more training data in source + target. Even if the source data is not similar to target but it is related (both are tweets), hence inclusion of this data effects the accuracy greatly. As the split percentage increases, model trained only by target data comes closer to the model trained by source + target in terms of accuracy. The reason being that now we have substantial and comparable amount of target data. Due to large amount of target data, even comparable amount of source data is only able to increase the accuracy by small fraction for large split percentage as compared to small split percentage.

2 Image Classification

d = 7

2.1 Binary Classification

For support vector $x : w^T x + b = y$. Hence $b = y - w^T x$. We can compute b in for different support vector and take the average.

$$b = \frac{1}{nSV} \sum_{x^{(i)} \in SV} (y^{(i)} - w^T x^{(i)})$$

(a)

Input Form:

$$\frac{1}{2}\alpha^T P \alpha + q^T \alpha$$
$$G\alpha \preccurlyeq H$$
$$A\alpha = B$$

Dual:

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

Therefore,

$$P_{ij} = y^{(i)}y^{(j)} < x^{(i)}, x^{(j)} >$$

$$q = \begin{pmatrix} -1 \\ -1 \\ . \\ . \\ -1 \end{pmatrix}$$

Condition due to soft SVM is $\forall i : 0 \leq \alpha_i \leq C$. G is $(2mxm)$ and H is $(2mx1)$ matrix.

$$G = \begin{pmatrix} 1 & 0 & ... & 0 \\ 0 & 1 & ... & 0 \\ . & . & ... & . \\ . & . & ... & . \\ 0 & 0 & ... & 1 \\ -1 & 0 & ... & 0 \\ 0 & -1 & ... & 0 \\ . & . & ... & . \\ . & . & ... & . \\ 0 & 0 & ... & -1 \end{pmatrix}$$

$$H = \begin{pmatrix} 1 \\ 1 \\ . \\ . \\ 1 \end{pmatrix}$$

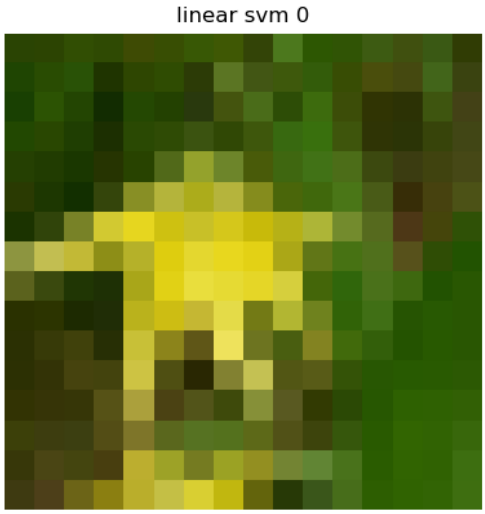
$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$$

Therefore,

$$A = \begin{pmatrix} y^{(1)} & y^{(2)} & ... & y^{(m)} \end{pmatrix}$$

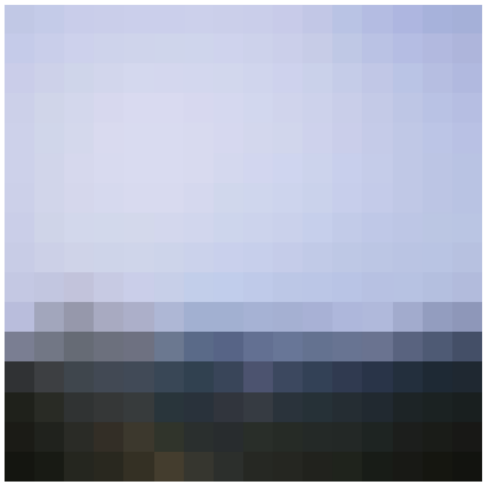
Number of Support Vectors = 297
Percentage of Training Sample constituting support vectors = 6.24%
b = 4.175
Training Accuracy = 97.18%
Validation Accuracy = 94.75%
Time Taken for training = 75.4 s

Images with Top 6 Coefficient:



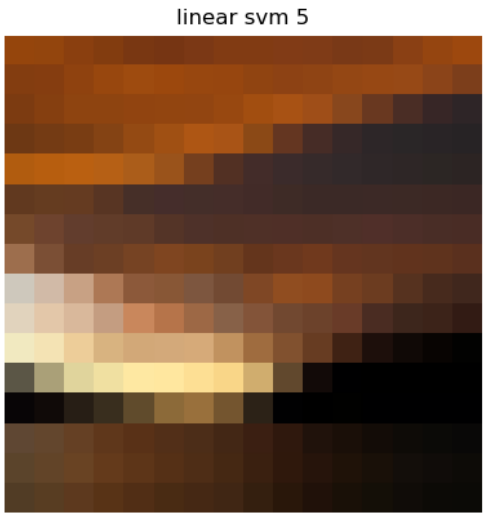
linear svm 1

linear svm 2



linear svm 3

linear svm 4



Plot of w:

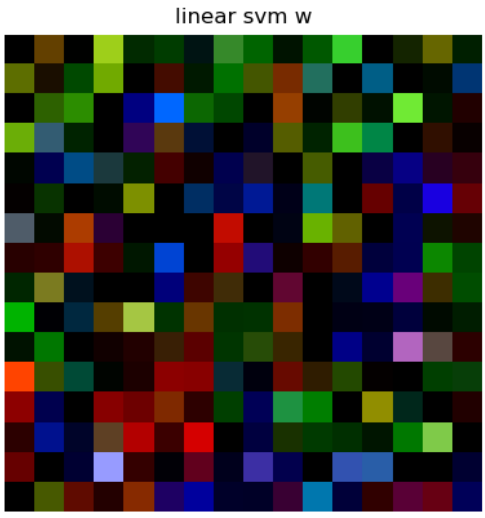


Figure 20: Plot of w for linear kernel

(b)

Input Form:

$$\frac{1}{2}\alpha^T P\alpha + q^T\alpha$$

$$G\alpha \preccurlyeq H$$

$$A\alpha = B$$

Dual:

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

Therefore,

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

where

$$K(x^{(i)},x^{(j)}) = e^{-\gamma(x^{(i)}-x^{(j)})^T(x^{(i)}-x^{(j)})}$$

This matrix P has kernel value times product of class label, hence it is used during prediction for calculation of training data accuracy to calculate kernel value effeciently.

$$q = \begin{pmatrix} -1 \\ -1 \\ . \\ . \\ -1 \end{pmatrix}$$

Condition due to soft SVM is $\forall i : 0 \leq \alpha_i \leq C$. G is $(2mxm)$ and H is $(2mx1)$ matrix.

$$G = \begin{pmatrix} 1 & 0 & ... & 0 \\ 0 & 1 & ... & 0 \\ . & . & ... & . \\ . & . & ... & . \\ 0 & 0 & ... & 1 \\ -1 & 0 & ... & 0 \\ 0 & -1 & ... & 0 \\ . & . & ... & . \\ . & . & ... & . \\ 0 & 0 & ... & -1 \end{pmatrix}$$

$$H = \begin{pmatrix} 1 \\ 1 \\ . \\ . \\ 1 \end{pmatrix}$$

$$\sum_{i=1}^{m}\alpha_i y^{(i)} = 0$$

Therefore,

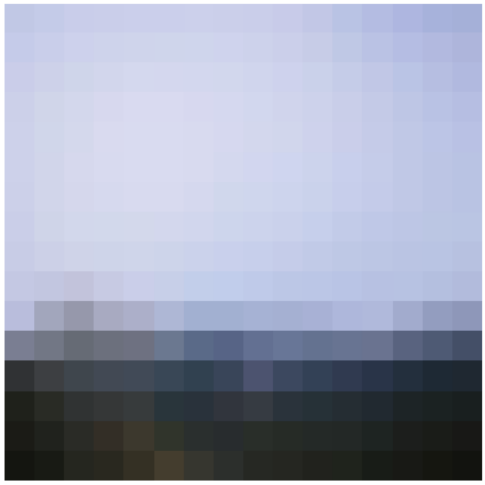
$$A = \begin{pmatrix} y^{(1)} & y^{(2)} & ... & y^{(m)} \end{pmatrix}$$

Number of Support Vectors = 394 Percentage of Training Sample constituting support vectors = 8.28%
Number of Support Vectors matching with (a) = 68
b = -4.725
Training Accuracy = 95.17%
Validation Accuracy = 93.75%
Time taken for training model = 98.32 s

Validation Accuracy is less than for gaussian kernel as compared to linear kernel. Lower accuracy for gaussian kernel is maybe due to $\gamma = 0.001$, a better value of gamma might increase the accuracy. Also it may be possible that the classes 1 and 2 which we are dealing with in binary classifier are better classified by a linear separator. Even though accuracy of gaussian kernel is lower, but the difference in accuracy of gaussian kernel and linear kernel is very less.

Images with top 6 coefficients:

gaussian svm 1



gaussian svm 2

gaussian svm 3

gaussian svm 4

(c)

Number of Support Vectors for linear kernel = 669
Number of Support Vectors for gaussian kernel = 1055
This is similar to the trend in part(a) and (b) that gaussian kernel has more support vectors as compared to linear kernel in our model with given data.
Number of Support Vectors for linear and gaussian kernel which are matching = 564
Number of Support Vectors matching for linear kernel with (a) = 286
Number of Support Vectors matching for gaussian kernel with (b) = 207

b for linear kernel = 4.174
Norm of difference in normalized w of (a) and normalized w for linear kernel found in this part = 0.0037
Clearly b for part (a) and (c) are almost same. Since norm of difference of w is very small, we can conclude w for part(a) and part(c) are also almost same.

Training Accuracy for linear kernel = 97.16%
Validation Accuracy for linear kernel = 94.75%
Training Accuracy for gaussian kernel = 95.17%
Validation Accuracy for gaussian kernel = 93.75%
Validation accuracy for linear kernel is same as obtained in part (a). Similarly validation accuracy for gaussian kernel is same as obtained in part (b).

Time taken for training linear kernel = 7.6 s

Time taken for training gaussian kernel = 7.4 s
Time taken for linear kernel by LIBSVM is 10 times less than that using CVXOPT. For gaussian kernel it is around 13 times lesser. Hence, implementation via LIBSVM is much faster than CVXOPT.

2.2 Multi-Class Image Classification

One vs One Classifier

(a)

Validation Accuracy = 55.00%
Training Time = 1869 s

(b)

Validation Accuracy = 56.08%
Training Time = 325 s
Validation Accuracy for LIBSVM is slightly more that for CVXOPT. Also the training time for LIBSVM is around 33 times less than that for CVXOPT. Hence implementation via LIBSVM is much faster than CVXOPT for multi class classification as well.

(c)

Confusion Matrix:

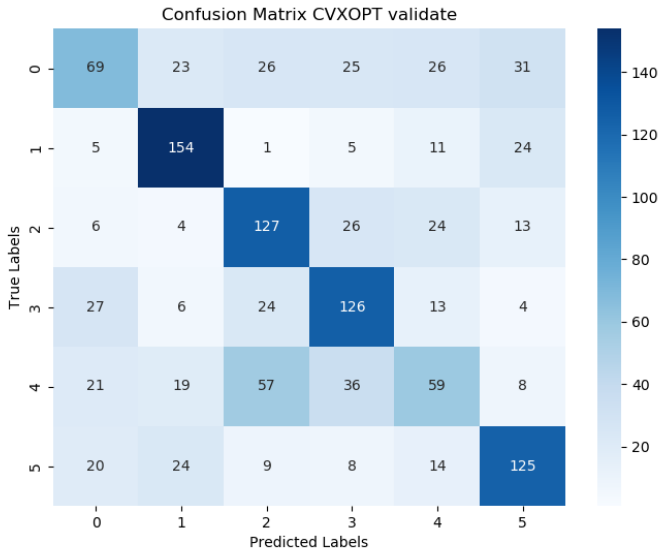


Figure 21: Confusion Matrix for CVXOPT Validate

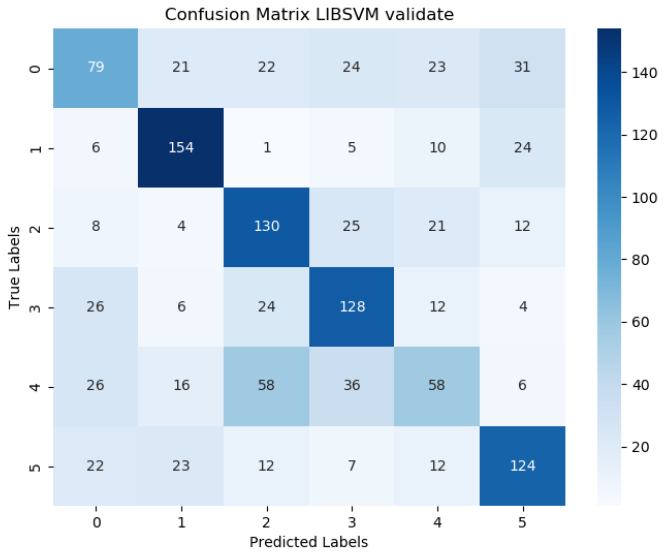
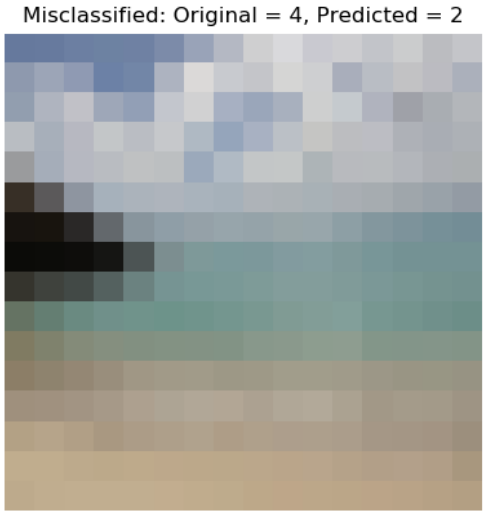


Figure 22: Confusion Matrix for LIBSVM Validate

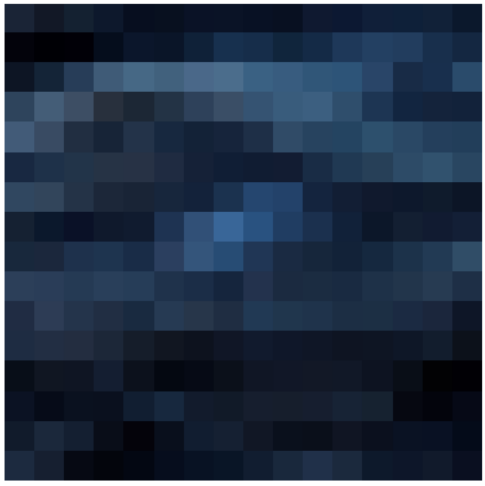
We observe that diagonal elements are larger as compared to off diagonal elements showing high accuracy of model.
Max off diagonal element in row:
Class 0 is misclassified mostly as class 5
Class 1 is misclassified mostly as class 5
Class 2 is misclassified mostly as class 3
Class 3 is misclassified mostly as class 0
Class 4 is misclassified mostly as class 2
Class 5 is misclassified mostly as class 1
Max off diagonal element in column:
Mostly Class 3 is misclassified as class 0
Mostly Class 5 is misclassified as class 1
Mostly Class 4 is misclassified as class 2
Mostly Class 4 is misclassified as class 3
Mostly Class 0 is misclassified as class 4
Mostly Class 0 is misclassified as class 5
Also, since class 4 has least value in its diagonal entry, it is misclassified most of the times.

Misclassified Figures:


Misclassified: Original = 5, Predicted = 0

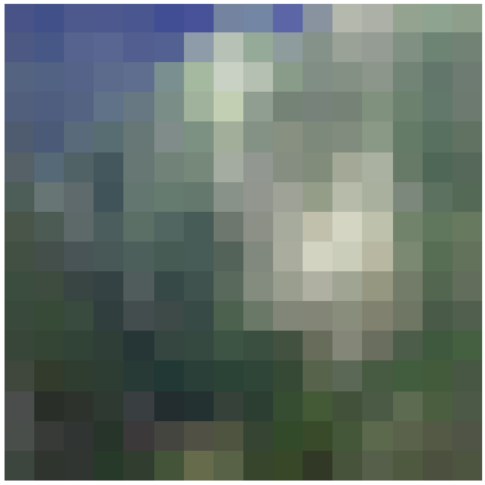

Misclassified: Original = 4, Predicted = 2

Misclassified: Original = 4, Predicted = 1

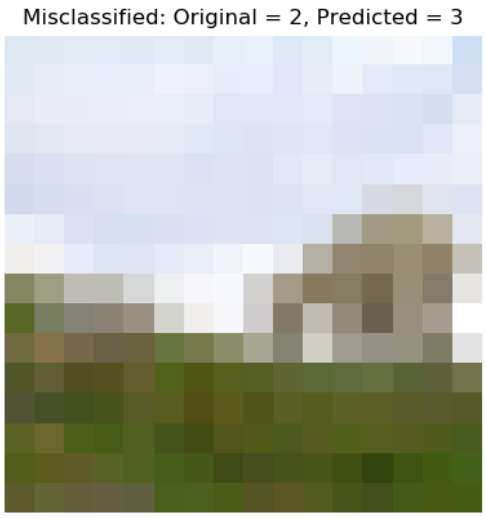


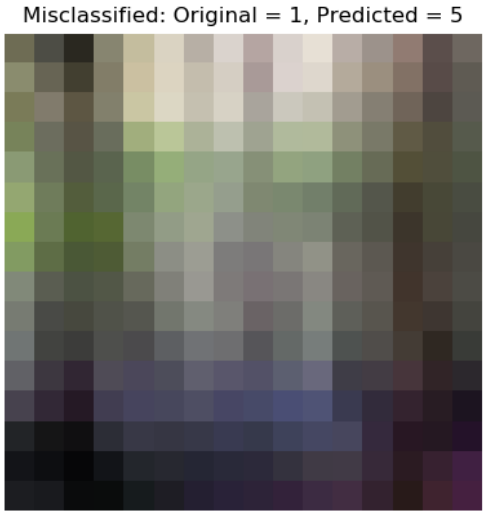
Misclassified: Original = 3, Predicted = 0

Misclassified: Original = 3, Predicted = 4



Misclassified: Original = 2, Predicted = 4


Misclassified: Original = 2, Predicted = 3


Misclassified: Original = 1, Predicted = 5

Misclassified: Original = 1, Predicted = 5

Misclassified: Original = 0, Predicted = 5

These misclassified Images looks in between the original and predicted values and hence the results make sense.

(d)

C	K Fold Cross Validation Accuracy	Validation Set Accuracy
10^{-5}	16.11	16.67
10^{-3}	16.53	16.67
1	56.34	55.5
5	57.48	58.25
10	60.47	59.0

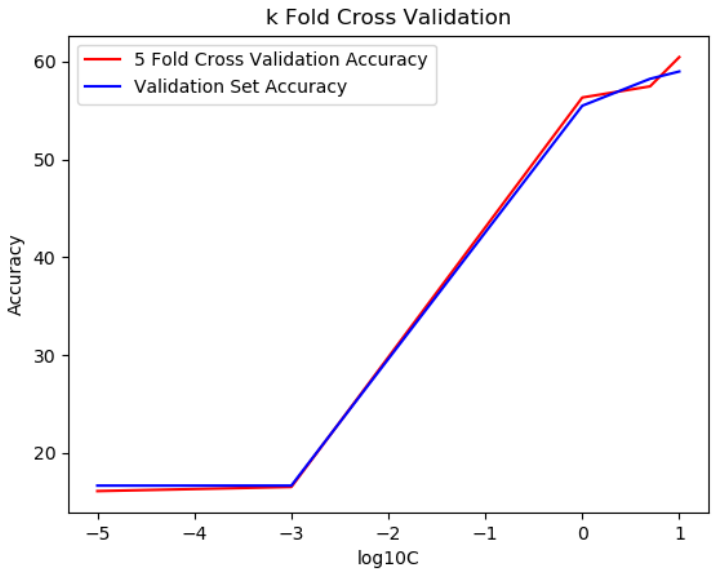


Figure 23: Plot for K Fold Cross Validation

C = 10 gives best value for both 5 fold Validation Accuracy as well as Validation Set Accuracy. As C increases both type of accuracies increases in the given range of C.