COL774
Assignment 2

Rishi Jain
2020CS10373

October 6, 2022

Contents

Libraries Used

- numpy
- math
- matplotlib
- nltk
- sklearn
- sys
- random
- os
- collections
- re
- wordcloud
- cvxopt
- shutill
- time

1 Text Classification

1.1 Naive Bayes

Implemented Naive Bayes as bag of words model.

$$\phi_k = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = k\right\}}{m} = \frac{\text{number of documents of class } k}{m}$$

$$\theta_{j=l|k} = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = k\right\} \sum_{i=1}^{m} 1\left\{x_j^{(i)} = l\right\}}{\sum_{i=1}^{m} 1\left\{y^{(i)} = k\right\} n^{(i)}} = \frac{\text{total number of occurences of word } j \text{ in all documents of class } k}{\text{total number of words in all documents of class } k}$$

Accuracy is as follows

- Training Accuracy : 94.492%
- Test Accuracy : 80.75333333333333%
- F1-Score for Test Data is : 0.8426788730859355

Data Visualization

The word clouds for the training data are:-

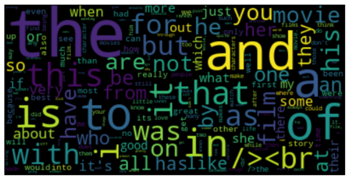


Figure 1: Word Cloud of Positive Reviews

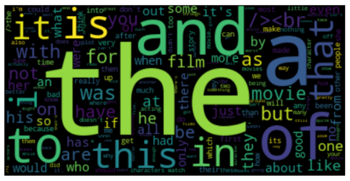


Figure 2: Word Cloud of Negative Reviews

1.2 Random Guessing and Constant Prediction

Random Guessing

Since the classification is binary, the probability of guessing the correct class is 1/2. Hence, the expected accuracy is 50%. The observations were

- Test Accuracy : 50.27333333333333%

Constant Prediction

If the prediction is set to constant, the test accuracy totally depends on the test dataset. The observations in our case were

- Test Accuracy : 66.66666666666667%

The rise in accuracy with respect to the random model is because the 10000 out of 15000 reviews are positive reviews. The Naive Bayes outperforms the above in accuracy by 14.12%.

1.3 Confusion Matrices

Naive Bayes

$$\begin{pmatrix} 7732 & 619 \\ 2268 & 4381 \end{pmatrix}$$

Random Guessing

$$\begin{pmatrix} 5065 & 2524 \\ 4935 & 2476 \end{pmatrix}$$

Constant Prediction

$$\begin{pmatrix} 10000 & 5000 \\ 0 & 0 \end{pmatrix}$$

Observations

- In Naive Bayes model almost equal percentage of positive reviews and negative reviews are miss-classified showing naive bayes is not biased towards classification .
- In case of Random Guessing, both classes are equally classified/misclassified bringing the accuracy close to 50%.
- The constant prediction model predicts positive for all test samples and hence misclassifies all samples with negative ground truth.
- All confusion matrices have high values in the first column because data contains mostly poitive reviews.

1.4 Stop-word removal and Stemming

Stop-words from training and test data were removed using nltk stopwords collection and stemming was performed using *PorterStemmer* from *nltk*
The resulting word clouds for positive and negative classes are

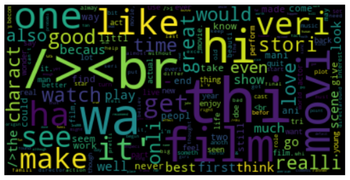


Figure 3: Word Cloud of Positive Reviews

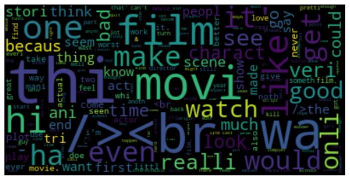


Figure 4: Word Cloud of Negative Reviews

Learning the Naive Bayes model on the transformed data, the observations were

- Training Accuracy : 95.096
- Test Accuracy : 81.98666666666666
- F1-Score : 0.853661178509532

Accuracy is increased as stemmed words have same meaning. Since nltk portstemmer does heavy stemming, words having different meaning may also become same leading to decrease in accuracy. Net effect is slight increase in accuracy here.

1.5 Feature Engineering

Bigrams

Inclusion of Bigrams to the vocabulary as features without stemming results is

- Training Accuracy : 99.668%
- Test Accuracy : 85.0%

Inclusion of Bigrams to the vocabulary as features with stemming and training the Naive Bayes model, results in the following

- Training Accuracy : 99.924%
- Test Accuracy : 85.1%
- F1-Score : 0.8809206670573819

Trigrams

As a new feature, Trigrams were added to the vocabulary with single words as features and training the Naive Bayes model, results in the following

- Training Accuracy : 99.996%
- Test Accuracy : 87.54666666666667%

As a new feature, Trigrams were added to the vocabulary with single words as features and training the Naive Bayes model with stemming results in-

- Training Accuracy : 100.0%
- Test Accuracy : 85.86%
- F1-Score : 0.887569573283859

Observations

- The addition of Trigram features have resulted in better performance on the test data. This is because of the information gained on the relative position of the words in the text review.
- Judging by the performance on the training and test data, it can be clearly seen that the inclusion of Trigrams with stemming results in overfitting and hence the drop in test-accuracy but without stemming trigram model has the best test accuracy.

1.6 Precision, Recall, F1-score

The best model out of the above was Naive Bayes on the data with trigram feature engineering without stemmed words and exclusion of stop-words. The test data performance metrics were:

- Precision = 0.8643
- Recall = 0.9441774087830457
- F1-Score = 0.9024746789182416

Accuracy can be used when the class distribution is similar while F1-score is a better metric when there are imbalanced classes as in the above case. Since d, F1-score tends to a better measure of performance and hence, selecting a model by F1-score should be appropriate.

2 Binary Image Classification

Binary classification was performed for the classes 3 and 4 from the gives dataset.

2.1 Linear Kernel SVM

To utilize the *CVXOPT* package, the dual problem to be reduced to the following problem

$$min\,(\frac{1}{2}\alpha^T P\alpha + q^T\alpha)$$

$$G\alpha \preceq H$$

$$A\alpha = b$$

The dual problem of the linear kernel SVM model is

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

$$0 \leq \alpha_i \leq C \;\; \forall i \in [1, m] \quad and \quad \sum_{i=1}^{m}\alpha_i y^i = 0$$

Defining the matrix Z such that $Z_{ij} = x^i_j y^i$, we can say

$$\alpha^T P\alpha = \sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j P_{ij}$$

Comparing $\frac{1}{2}\alpha^T P\alpha$ with the first term of the objective function,

$$P_{ij} = (x^i_j y^i)^T (x^i_j y^i)$$

$$P = ZZ^T$$

Hence, $q^T\alpha$ represents the second term of the objective function, evaluating q to a m-size vector with $q_i = -1$ for all i
The inequality constraints can be represented as $2 * m$ inequalities

$$-\alpha_i \leq 0 \;\; \forall i \in [1, m]$$

$$\alpha_i \leq C \;\; \forall i \in [1, m]$$

To represent the above, G and H matrices will evaluate as $2m \times m$ and $2m \times 1$ matrices respectively.

$$Z_{ij} = x_j^{(i)} y^{(i)}$$

$$P = ZZ^\top \quad q = \begin{bmatrix} -1 \\ -1 \\ \vdots \\ -1 \end{bmatrix}$$

$$G = \begin{bmatrix} -1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

$$h = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ C \\ C \\ \vdots \\ C \end{bmatrix}$$

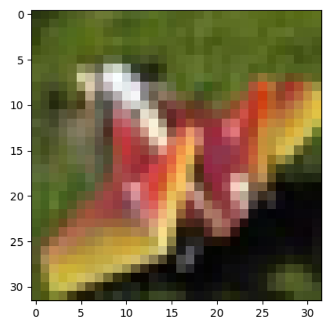
$$b = 0$$

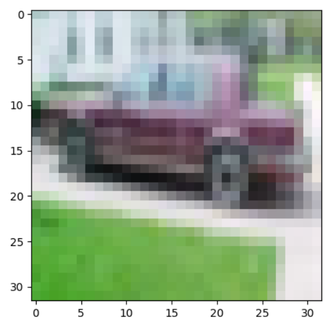
Feeding the matrices to the convex optimization algorithm for $C = 1.0$, the results were obtained as follows

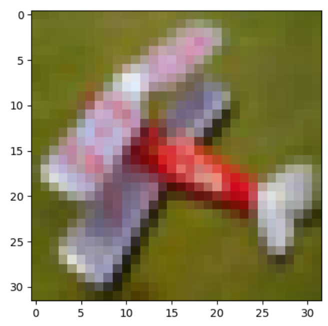
- No. of Support Vectors = 2159, which constitutes 53.9755% of training sample
- w = [-0.51130583 -0.36870322 -0.05281205 ... 0.44036109 0.1363592 -1.16794184]
- b = -0.3880946114307968
- Test Accuracy : 67.4%

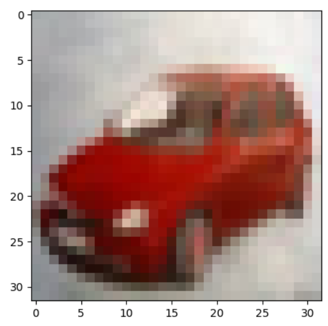
Visualizing the Support Vectors

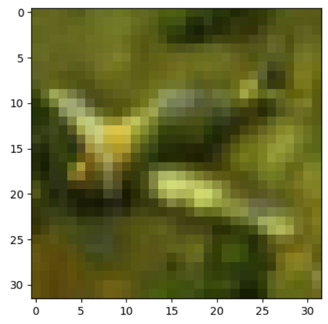
The top-5 support vectors(by value of α) were reshaped and plotted as an RGB image.


0
5
10
15
20
25
30
0
5
10
15
20
25
30


0
5
10
15
20
25
30
0
5
10
15
20
25
30


0
5
10
15
20
25
30
0
5
10
15
20
25
30


0
5
10
15
20
25
30
0
5
10
15
20
25
30



2.2 Gaussian Kernel SVM

The new dual problem with Gaussian Kernels will be

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

Where,

$$\phi(x^i)^T\phi(x^j) = e^{-\gamma||x^i - x^j||^2}$$

With the constraint being the same, change is only observed in the P matrix.

$$P_{ij} = y^i y^j e^{-\gamma||x^i - x^j||^2)}$$

$$P_{ij} = y^i y^j e^{-\gamma(||x^i||^2 + ||x^j||^2 - 2x^i x^{jT})}$$

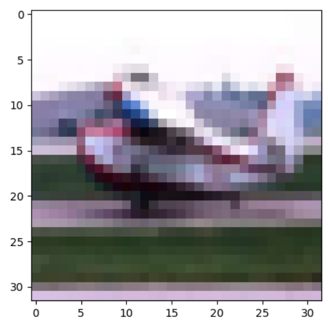
The above expression can hence be vectorized and calculated efficiently. Since we can't explicitly store w since it is an infinite-dimensional vector, we use the entire set of support vectors to make predictions on the test dataset.
Feeding the matrices to the convex optimization algorithm with $C = 1.0$ and $\gamma = 0.001$, the results were obtained as follows

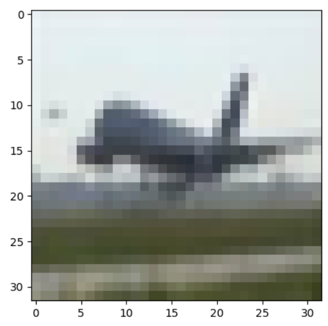
- No. of Support Vectors = 2643, which constitutes 66.075% of training samples
- Test Accuracy : 77.85%

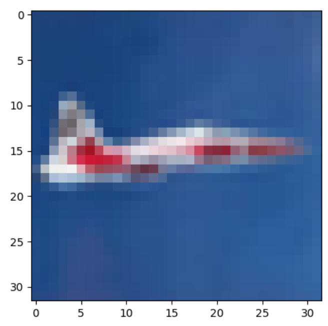
The Gaussian Kernel obviously performs better than a linear kernel considering the it's capability to learn a lot more features than a linear kernel(which is limited to the number of features in a training sample).

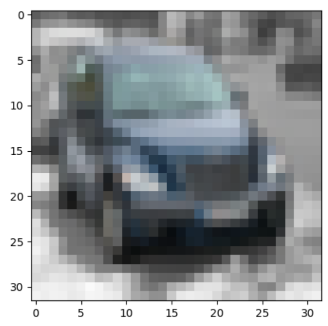
Visualizing the Support Vectors

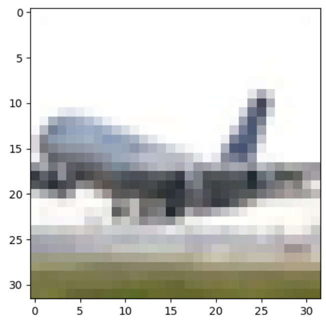
The top-5 support vectors(by value of α) were reshaped and plotted as an RGB image.











2.3 Scikit-learn SVM model

The SVM model from Scikit-learn was used to perform the binary classification task as above with Linear and Gaussian Kernels with the same hyperparameters.

Linear Kernel

- No. of Support Vectors = 2140, which constitutes 53.5% of training samples
- Test Accuracy : 67.4%
- w = [[-0.51065518 -0.36886153 -0.05346012 ... 0.44094125 0.13648895 -1.16855984]]
- b = -0.3883493

Observations

- The number of support vectors dropped when compare to the model learnt using CVXOPT from 2159 to 2140 with 2140 support vectors common between the two models (Hence, the support vectors here are a subset of the ones in CVXOPT)
- Value of b and w are also very similar in both cases. Also the accuracy is also same in both cases. The change in number of support vectors is only due to the the change in precision in the way these two libraries calculate.

Gaussian Kernel

- No. of Support Vectors = 2631, which constitutes 65.775% of training samples
- Test Accuracy : 77.8%

The number of support vectors dropped when compare to the model learnt using CVXOPT from 2643 to 2631 with 2631 support vectors common between the two models (Hence, the support vectors here are a subset of the ones in CVXOPT)

Computational Cost (Training)

SVM Learning Algorithm	**Linear Kernel**	**Gaussian Kernel**
CVXOPT quadratic solver	27.975309133529663s	60.03721332550049s
Scikit-learn	25.835566520690918s	21.277379751205444s

Judging by the above data, it is clear that the optimization algorithm used by Scikit-learn outperforms the *CVXOPT*'s quadratic solver. Also, for the give training data, CVXOPT seems to take more time in optimizing with a gaussian kernel than the linear kernel, which is the opposite with Scikit-learn.

3 Multi-Class Image Classification

The data(A subset of CIFAR-10) consisted of 5 classes, Airplane(0), Automobile(1), Bird(2), Cat(3) and Deer(4).

3.1 One-vs-One SVM model with CVXOPT

For the given training data with 5 classes, ${}^5C_2 = 10$ binary classifiers were trained. For testing, majority voting was considered among the 10 classifiers.

- Test Accuracy : 59.24%
- Training time : 960.5925s

3.2 Scikit-learn One-vs-One SVM model

Using the one-vs-one SVM model by Scikit-learn, the results were

- Test Accuracy : 59.3%
- Training time : 182.73473191261292s

Observations

- The test accuracy has improved slightly with Scikit-learn and hence outperforms the CVXOPT's quadratic solver optimization algorithm.
- The computational cost in training on the given training data has reduced drastically in case of Scikit-learn's SVM model.

3.3 Confusion matrices and Misclassifications

One-vs-One SVM model with CVXOPT

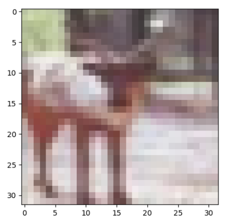
The confusion matrix on the test data was observed

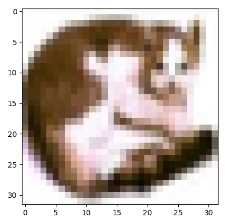
$$\begin{pmatrix} 723 & 96 & 141 & 79 & 92 \\ 84 & 728 & 59 & 95 & 44 \\ 78 & 46 & 407 & 122 & 217 \\ 65 & 93 & 138 & 577 & 120 \\ 50 & 37 & 255 & 127 & 527 \end{pmatrix}$$

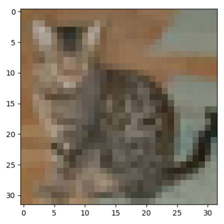
The misclassifications by the model were randomly picked and reshaped for visualization.

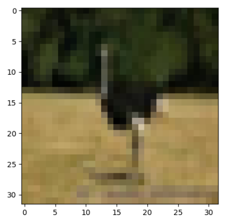
Observations

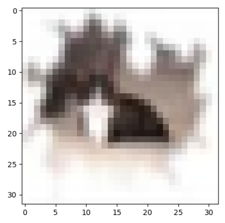
- Correct classifications can be read along the left diagonal of the confusion matrix, with the highest value being for the Airplane(0) and Automobile(1) classes.
- The highest misclassifications are observed in Bird(2) class. It can also be seen that the model misclassifies too many Bird(2) as Deer(4),Cat(3) or Airplane(1). The Misclassifications in the figure below are hence justified by the confusion matrix.

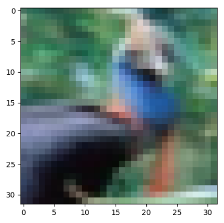

(a) Predicted : Cat(3)

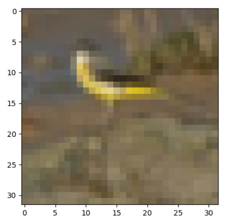

(b) Predicted : Automobile(1)

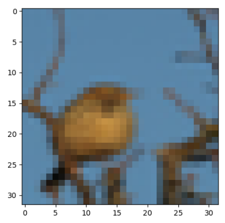

(c) Predicted : Bird(2)

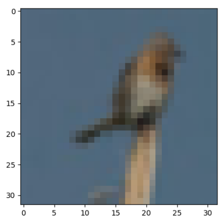

(d) Predicted : Deer(4)


(e) Predicted : Airplane(0)


(f) Predicted : Automobile(1)


(g) Predicted : Deer(4)


(h) Predicted : Airplane(0)


(i) Predicted : Cat(3)

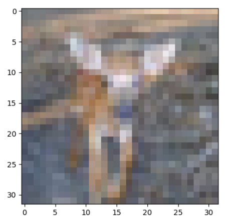

(j) Predicted : Cat(3)

Figure 5: Misclassifications by the model

One-vs-One SVM model with Scikit-Learn

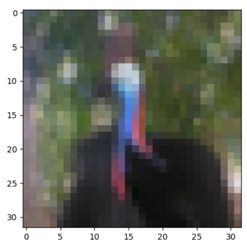
The confusion matrix on the test data was observed

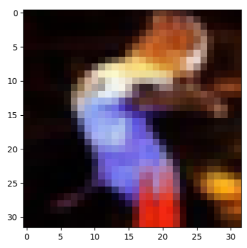
$$\begin{pmatrix} 729 & 100 & 145 & 82 & 98 \\ 83 & 731 & 61 & 97 & 48 \\ 78 & 42 & 409 & 123 & 212 \\ 61 & 92 & 133 & 572 & 118 \\ 49 & 35 & 252 & 126 & 524 \end{pmatrix}$$

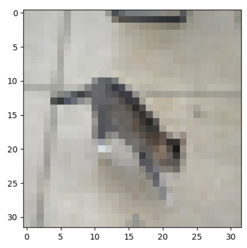
The misclassifications by the model were randomly picked and reshaped for visualization.

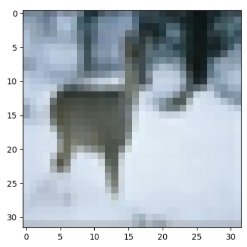
Observations

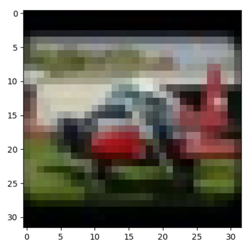
- Correct classifications can be read along the left diagonal of the confusion matrix, with the highest value being for the Airplane(0) and Automobile(1) classes.
- The highest misclassifications are observed in Bird(2) class. It can also be seen that the model misclassifies too many Bird(2) as Deer(4),Cat(3) or Airplane(1). The Misclassifications in the figure below are hence justified by the confusion matrix.

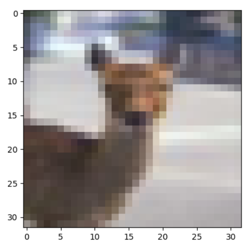

(a) Predicted : Automobile(1)

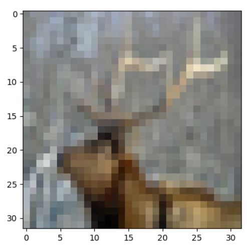

(b) Predicted : Cat(3)

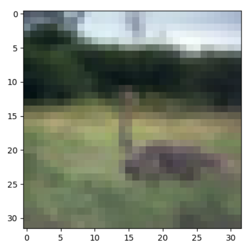

(c) Predicted : Bird(2)

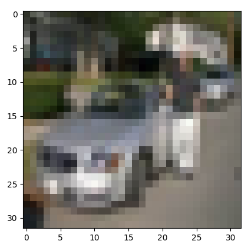

(d) Predicted : Airplane(0)

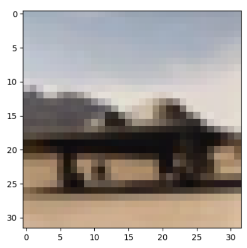

(e) Predicted : Automobile(1)


(f) Predicted : Cat(3)


(g) Predicted : Bird(2)


(h) Predicted : AutoMobile(1)

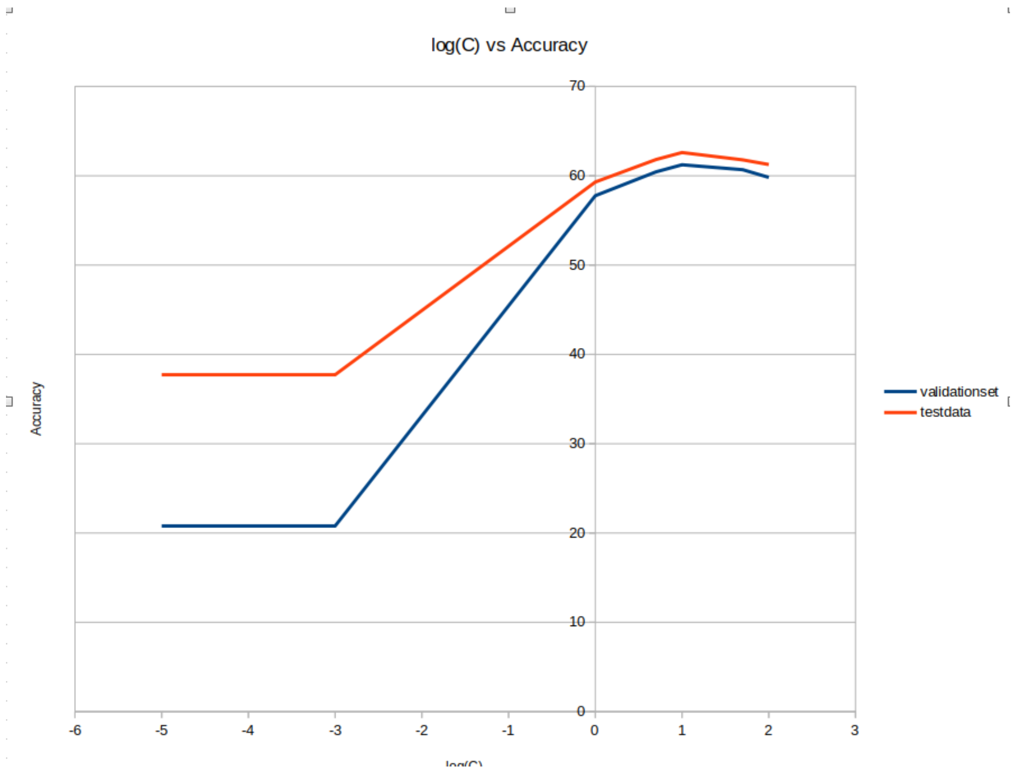

(i) Predicted : Deer(4)


(j) Predicted : Automobile(1)

3.4 Cross-Validation and Regularization

5-Fold Cross Validation was implemented with Sckit-learn's SVM model and Validation and Test Accuracies were noted for different values of the regularization parameter C.

C	Validation Accuracy	Test Accuracy
10^{-5}	20.79%	37.72
10^{-3}	20.79%	37.72%
1	57.77%	59.30%
5	60.44%	61.82%
10	61.23%	62.6%
50	60.68%	61.78%
100	59.82%	61.26%



- The validation accuracy peaks at $C = 10$, which also gives the highest accuracy in the test data. We can conclude that cross validation is a good method to determine model which will perform best on training data set.

- From the primal problem of the SVM, we can deduce the for lower values of C, the model tends to have a high relaxation on ξ_i and hence under-fits on the training data, resulting in low validation and test accuracies. On the other side, for high values of C, the model has a high penalty on the values of ξ_i and results in a model that over-fits on the training data, and again dropping in accuracy. The model peaks in performance between under-fitting and over-fitting regions. This is all in compliance with the above experimental observations.