COL774:A1

Sidharth Agarwal 2019CS50661

October 5, 2022

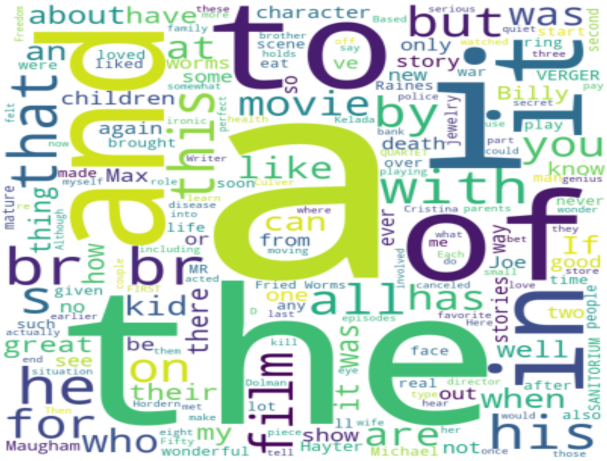
Libraries Used:

matplotlib, numpy, pandas, scipy, wordcloud, nltk, sklearn, cvxopt, standard python libraries.

1: Text Classification

1.a

I first tried with multivariate naive bayes but the accuracy was not good. So in the final submission I implemented multinomial naive bayes. I got an accuracy of 91.74% on the training data and an accuracy of 79.74% on the test data.
The wordlcoud obtained without removing stopwords, for both the labels was same and was as follows.



1.b

The accuracies obtained were as follows.

- i) 50%
- ii) 66.66%
- iii) our algorithm gives 79.74% which is large improvement as compared to random and positive baselines.

1.c

The confusion matrices obtained for the test data are as follows.

$$\text{For naive bayes in part a} \begin{pmatrix} 7583 & 622 \\ 2417 & 4378 \end{pmatrix}$$

$$\text{For random model in part b} \begin{pmatrix} 5000 & 5000 \\ 2500 & 2500 \end{pmatrix}$$

$$\text{For positive model in part b} \begin{pmatrix} 10000 & 5000 \\ 0 & 0 \end{pmatrix}$$

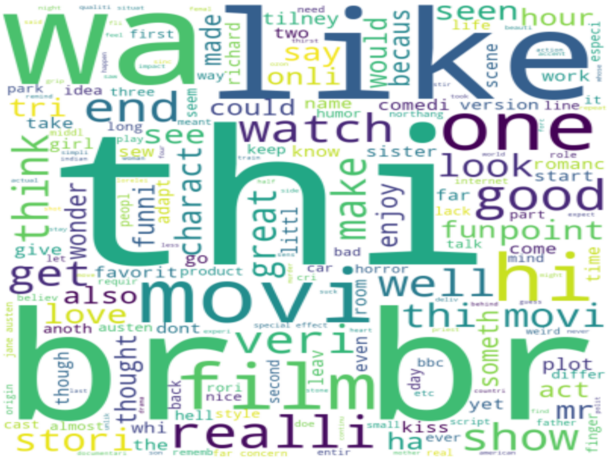
Now note that the sum of diagonal being 11961 is highest in naive bayes. This means that the number of data points classified correctly are highest in the naive bayes model.

The pattern observed is as the accuracy increases the sum of number of entries in diagonals increases and the sum of entries in non-diagonals decreases.

1.d

With this new model the acccuracy over the training set was 91.74% and the accuracy over test data was 81.02%. Hence the accuracy slightly increases for test set because of stemming and stopword removal.

The wordlcoud obtained for this model is as follows and is same for both the classes.



1.e

I constructed new features by taking all the bigrams formed by taking consecutive terms. This gave a good boost in performance by giving the model accuracy on training data to be 99.2% and on testing data to be 84.1%.

Now I also tested with adding more features like trigrams. Where I took all the three consecutive appearing words as additional features with previously initialized bigrams. This increased the accuracy to 84.4%, which was not to much as compared to additional computation time.

If we compare our models used here with part a and part d then additional set of features have definitely increased performance. This is because now we are seeing the pattern and structure of word occurances as compared to previous one where we were just focused with the frequency of word occurances.

1.f

Model tested	Precision	Recall	F1 Score
Model a	0.924	0.758	0.833
Model d	0.927	0.775	0.844
Model e	0.944	0.809	0.871

For this kind of data I believe that F1 score is a better metric since the data is not equally distributed among different classes as we have 10k documents labelled as positive and 5k documents labelled as negative.

2: Binary Image Classification:

Since my entry number is 2019CS50661, so the last digit is 1 and hence the classes of my concern were class 1 and class 2. the input format of cvxopt is as follows.

$$\begin{aligned} min_{\alpha}(\alpha^T P\alpha + q^T\alpha + d) \\ G\alpha \leq H \\ A\alpha = b \end{aligned} \tag{1}$$

The dual in summation format is given as:

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

So after looking closely we can clearly see that..

$P = \sum_{i,j} y^i y^j < x^i, x^j >$ For linear

$P = \sum_{i,j} y^i y^j K(x, z)$ For Gaussian where K is gaussinan kernel

$b = 0$

$q = -1[m]$

$A = y^T$

G is I of dimension m*m stacked with -1*I.

H is column vector C[1] of dimension stacked with column vector 0[1] of dimension m

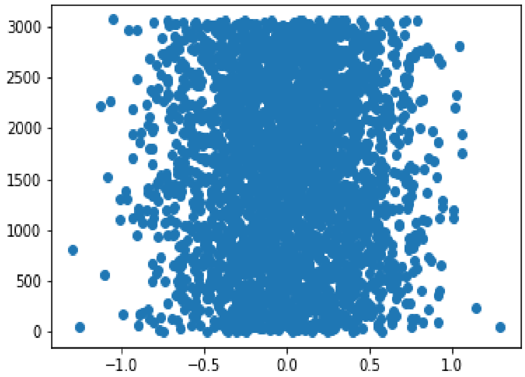
where m is the number of training examples.

The value of C taken was 1.0 and gamma as 0.001

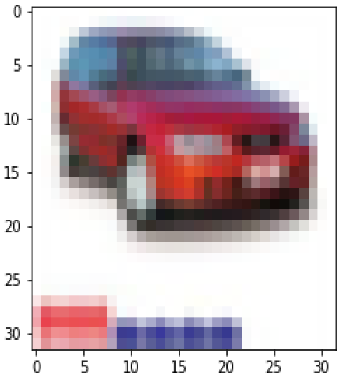
2.a

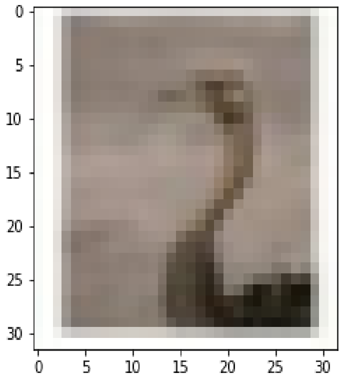
After taking the minimum value of alpha to be 1e-4 for it to be classified as support vector. I found the total number nSV to be 1543. About 38.5% of the training data constitute support vectors.

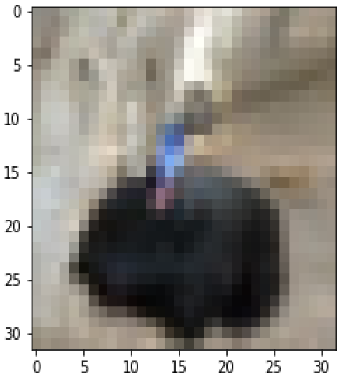
The training accuracy is 94.25% and the testing accuracy is 77.25%, the value of b was found to be -0.415. The representation of weight vector in decimal plot is as follows.

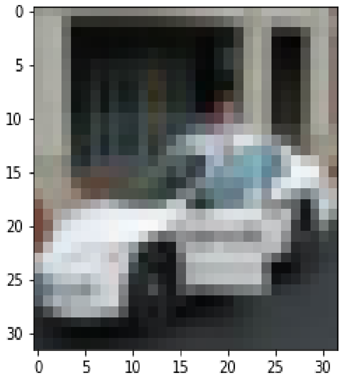


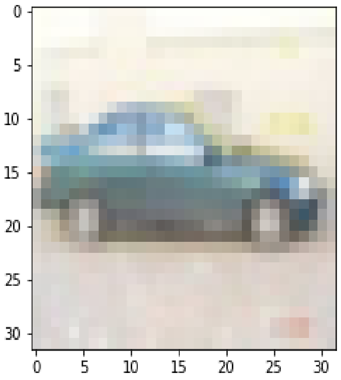
Now plotting the top 5 coefficient support vectors as images.



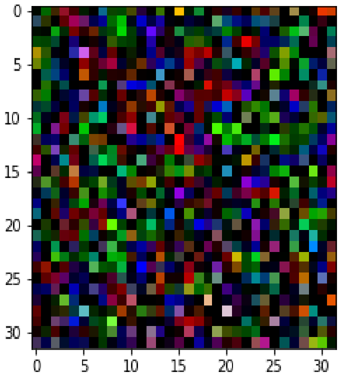






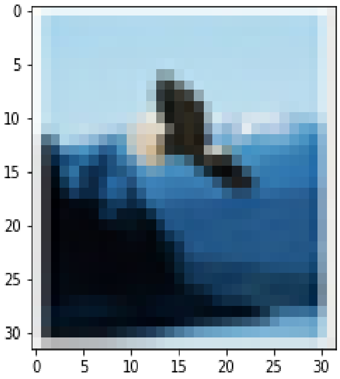


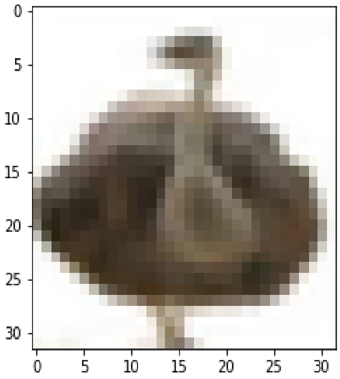
The representation of weight vector as RBG image is as follows.

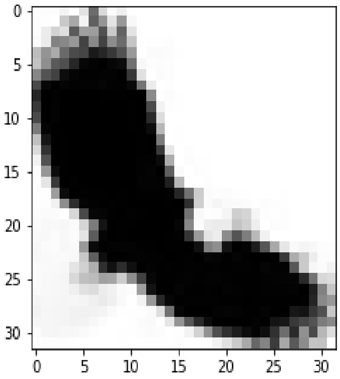


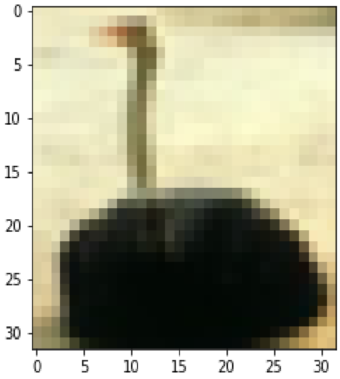
2.b

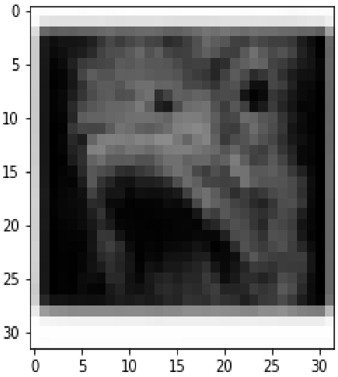
The nSV was found to be 1872 for gaussian kernel. With the training accuracy to be 84.2% and the testing accuracy to be 82.1%. The number of matching support vectors are 1161.











2.c

Model tested	Train Accuracy	Testing Accuracy	nSV	Training Time(in s)
Linear Sklearn SVM	94.8	78.05	1541	38.04
Gaussian Sklearn SVM	89.32	87.74	1865	24.10
Linear CVXOPT	94.25	77.25	1543	68.30
Gaussian CVXOPT	84.2	82.1	1872	91.7s

As we can clearly see that the number of support vectors generated in CVXOPT method and Sklearn SVM are almost same and as well the performance is almost similar, with the sklearn counterpart performing slightly better. After investigating for Linear SVM 1541 support vectors are common among Sklearn and CVXOPT method and for Gaussian SVM 1865 support vectors are common among sklearn and CVXOPT.

The mean square error between linear cvxopt W and linear sklearn opt is 0.525 and the absolute difference between there b was 0.4270

3: Multi Class classification

For all the experiments here we used C = 1.00 and gamma = 0.001.

3.a

The accuracy on the test data comes out to be 56.25% and the time taken for training was around 1600 seconds.

3.b

The accuracy on training data comes out to be 62.28% and on testing data comes out to be 59.3%. The training time being 213 seconds.

3.c

The confusion matrix obtained for 2(a) is as follows.

$$\begin{pmatrix} 739 & 47 & 105 & 62 & 47 \\ 139 & 567 & 90 & 175 & 29 \\ 145 & 24 & 459 & 207 & 165 \\ 97 & 39 & 134 & 661 & 69 \\ 83 & 20 & 288 & 192 & 417 \end{pmatrix}$$

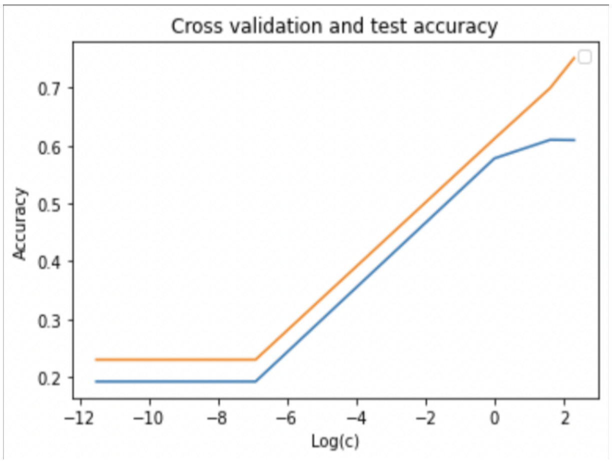
The confusion matrix obtained for 2(b) is as follows.

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

As we observe classes 0 and 1 are classified correctly maximum number of times. And class 2 is misclassified maximum number of times and mostly into class 4. Both class 3 and 4 are missclassified average number of times as compared to other ones.

3.d

So the plot obtained for gamma = 0.001 and values of C being from the list [1e-5,1e-3,1,5,10].



SO as we can see the validation accuracy increases as we increase the value of C, and so does the value of test accuracy, except when we go from C = 5 to C = 10 where it slightly decreases. So as we increase C we are going from soft margin to hard margin, since we will try to make epsilon as minimum as possible. Now since their are at least some samples always lying on the wrong side of boundary for this data, optimal C is not infinity as the value of test accuracy starts to converge.