ASSIGNMENT 3 REPORT

MACHINE LEARNING

Waquar Shamsi (MT20073)
Reshan Faraz (PhD19006)

10th November, 2020

Question 1. (60 Points)

Use the CIFAR-10 dataset for all the experiments. Choose hyper-parameters in a systematical manner.

ABOUT THE DATASET:

The CIFAR10 dataset consists of color images of 10 classes. We found that the correct representation of image data is 32*32*3 where the 3 is for Red Blue and Green colors.

There are 50000 images in total. We split that data in a ratio of 80:20 to train and test. A sample image of truck from the dataset is attached below:

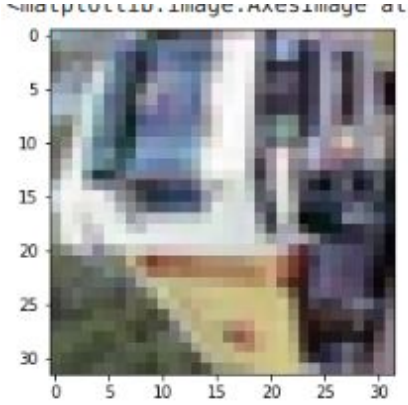


Fig: Image of a Truck in given dataset

What is a feature descriptor:

It is a simplified representation of the image that contains only the most important information about the image. [ref]

1.a: Perform PCA using sklearn on the dataset such that 90% of the total variance is retained - feature descriptor(1)

Approach :

PCA is affected by scale so scaled the features before applying PCA. (It's required as some features might have less variance due to its scale whereas some other features might have high variance due to different scales).

Here I used Standardization as the scaling technique. In standardization, values are centered around the mean with a unit standard deviation. Resulting in 0 mean and unit standard deviation.

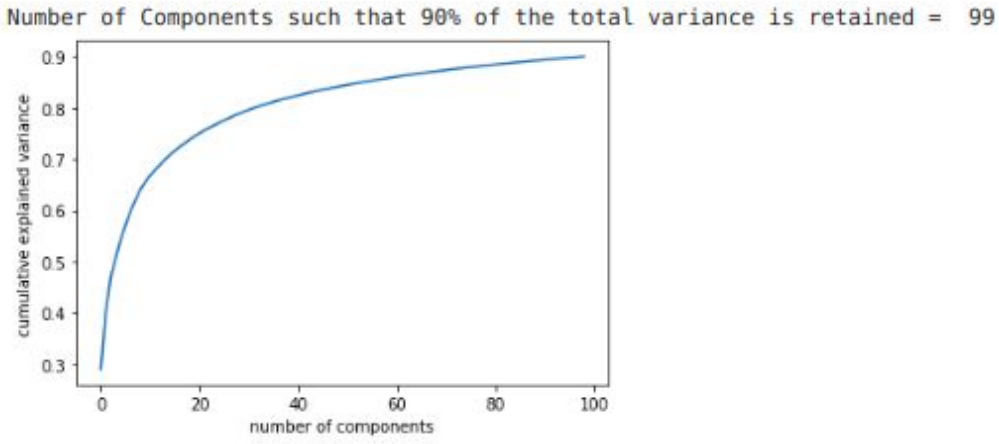
The formula for standardization of features:

$$X' = \frac{X - \mu}{\sigma}$$ [ref]

To find the number of features such that 90% of the variance is retained I used sklearn.decomposition.PCA.

It gave me the number of features as 99.

The plot for the same is attached following:



As it can be seen that number of total variance retained is proportional to the number of features. So the total variance retained increases with the number of features.

1.b: Combine HOG and color histogram (must be implemented from scratch) on a whole ie., (hog + color hist) - feature descriptor(2)

Approach :

HOG:

HOG stands for histogram of oriented gradients. It is an image feature extraction algorithm. It is used for edge detection. The steps to obtain the HOG features are described below:

1. Converted image to grayscale using cv2. So the image dimensions become (32*32)
2. Obtained a single patch of size 8*8 from the entire image of 32*32.
   16 such patches will be obtained for an image of size 32*32.
3. Calculated the X gradients.
   By subtracting the left cell value of the cell in question from the right cell value of the cell in the right of the cell in focus.
4. Calculate the Y gradients.
   Similar to the previous step just replacing right with top and left with bottom.
5. Calculate magnitude for each cell using the following formula:
   $$\text{Gradient Magnitude} = \sqrt{[(G_x)^2+(G_y)^2]}$$
6. Calculate the direction of each cell using the following formula:
   $$\tan(\Phi) = G_y / G_x$$
7. atan function returns angle in radians, convert it to degrees.
8. Create histogram using numpy.histogram using 9 bins of equal width.

9. Save the vector of histogram.
10. Now 4 vectors representing the block of size 16*16 thus combining 4 patches of 8*8 size. And normalize using the following formula and store as feature of image:

$k = \sqrt{(a1)2+ (a2)2+ (a3)2+ .... (a36)2}$ calculate k like this and then,

$$\text{Normalised Vector} = \left(\frac{a_1}{k}, \frac{a_2}{k}, \frac{a_3}{k}, ... \frac{a_{36}}{k} \right)$$

The resultant normalized vector will have size of 36*1. This is called as one block.

11. Now slide the window. Like this, (4-1)*(4-1) = 3*3 = 9 such blocks will be created. These all contribute to the features of hog image so the total number of features for single image becomes:

    9*36 = 324.

12. Repeat these steps for all 50000 images.

COLOR HISTOGRAM:

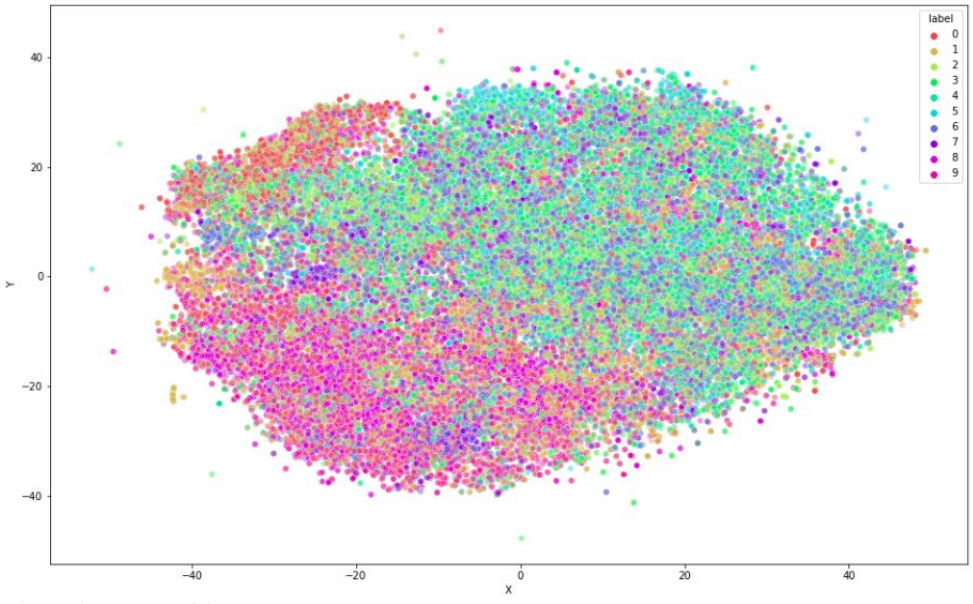
1. Convert a single image to HSV using cv2 library.
2. Pick only the hue part because it best represents the image.
3. Apply numpy.historgram with 9 bins to get the histogram.
4. Each of these bins becomes a feature thus there are 9 features in total.
5. Repeat the steps for all images.

COMBINE HOG + COLOR HISTOGRAM:

1. It is observed that values of color_histogram are much greater, so normalize the color_histogram features before appending to hog_features.
2. Append the features of color_histogram to features of hog using numpy.concatenate.
3. So the total number of features becomes 324+9 = 333
4. Save the feature descriptor.

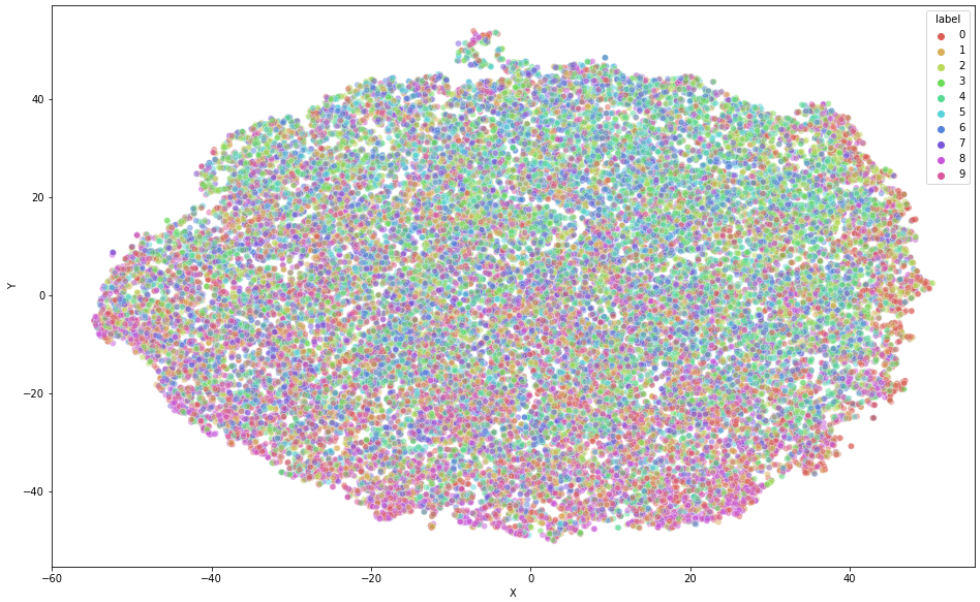
2.a: Visualize the 2D t-SNE plot. State your observations.

TSNE PLOT FOR PCA:



It can be observed that reducing the number of features from 99 to 2 makes it difficult to linearly separate the classes. The classes seem to heavily overlap. No ML algorithm might classify the samples correctly in 2 dimensions, but it's easier to visualize the data in 2 dimensions.

TSNE PLOT FOR HOG + COLOR_HISTOGRAM:



It can be observed that its even more overlapping in comparison to PCA-TSNE, It may be due to the fact that there were more features in hog+col_hist and reducing from more features to two features will definitely make the plot more incomprehensible to understand. Though the shape of the plot seems very similar to the plot in PCA.

It fails to preserve the global geometry of the data

############################## ADDED 10/11/2020 ################
###

Answer 1 -3

FOR PCA :

Following are the parameter obtained when applied grid search cv:

```
{'C': 10, 'gamma': 8.98775673026395e-08, 'kernel': 'rbf'}
SVC(C=10, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=8.98775673026395e-08,
    kernel='rbf', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

Following are the parameter used in the grid search cv based on the data visualization from the PCA (Answer 1-b):

```
param_grid = [
 {'C': [0.01, 1, 10,100],
          'gamma': [0.1, 0.01, xx,1],
          'kernel': ['rbf']}  ]
```

Accuracy over Testing dataset = **0.5521**

Accuracy over Training dataset = **0.928325**

Over all time to find the parameter using grid search cv (cv= 5) took **494.0 minute**

And for the best parameter it took 5.2 minute. From the testing and training accuracy we conclude that the model will perform better over training data rather than on testing data sets.

For HoG :

Accuracy over Testing dataset = `0.3002`

Accuracy over Training dataset =`0.5502`

Answer 1 -4:

For PCA

From comparing with solution 1-3 we come to know that testing accuracy is drop by about 10% while training accuracy will increase and it overfit the model with the accuracy of 100%.

Following are the results after removing the support vectors.

We found 35350 number of support vectors and hence our training dataset will be reduced so the model will fit for a small dataset and perform poorly over testing dataset.

Accuracy for Testing dataset : **0.4658** (which approx 10% less than the result obtained in Q1-3)

Accuracy for Training dataset : **1.0** (which approx 8% more than the result obtained in Q1-3)

For HoG :

Following are the results after removing the support vectors.

We found 38586 number of support vectors and hence our training dataset will be reduced so the model will fit for a small dataset and perform poorly over testing dataset.

Accuracy for Testing dataset : **0.2474**

Accuracy for Training dataset : **0.4405**

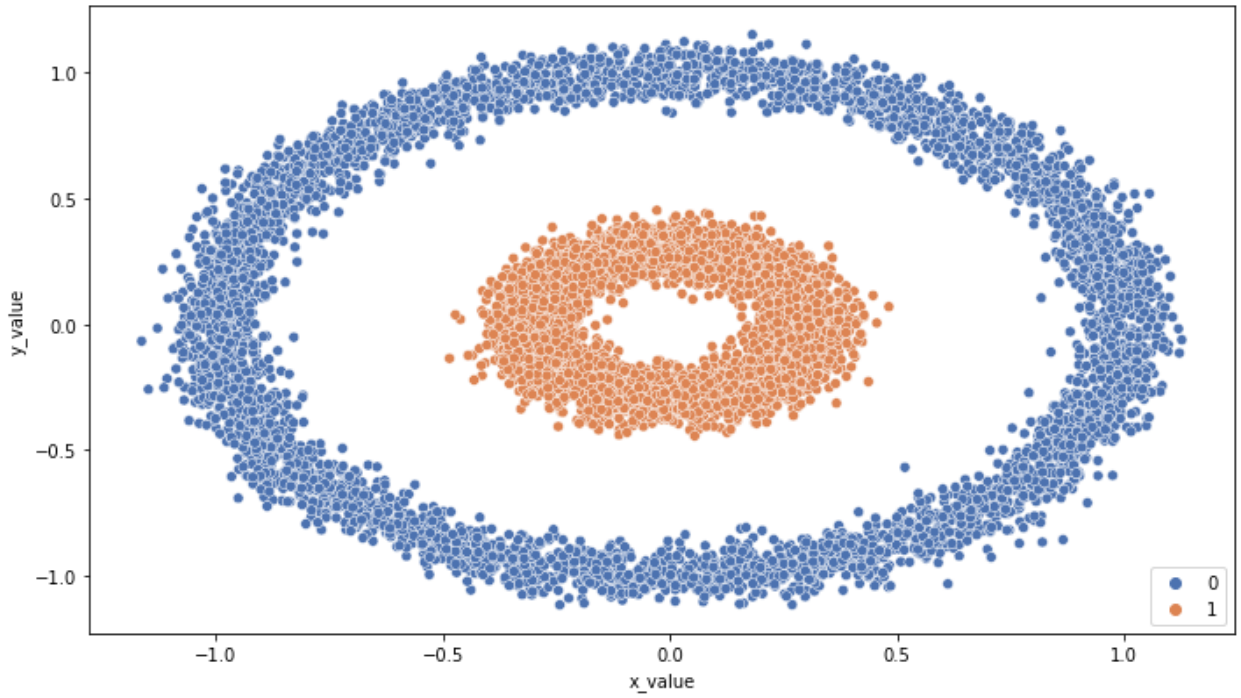
Answer to Question 2

Preprocessing of Data :

After loading the data we divided the data into testing and training set. 20% of data is used for testing and the rest is used by training. I wrote my own function to divide the data.

Answer 2-1):

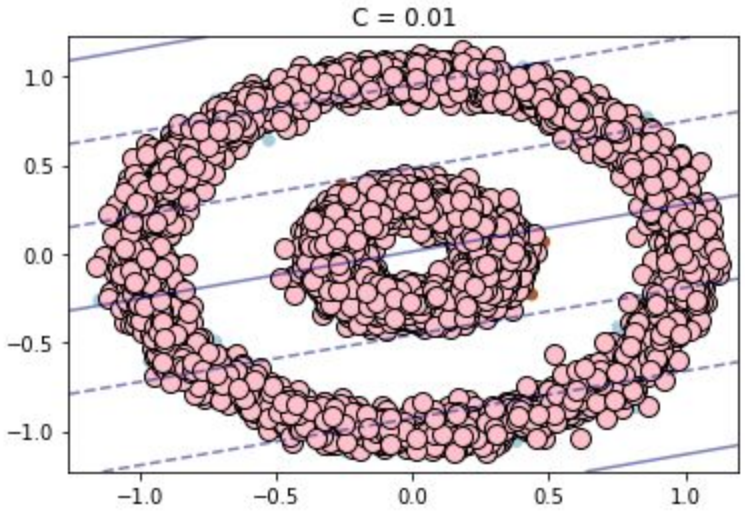
Following is the visualization of data. I used matplotlib to plot the data along with their labels.

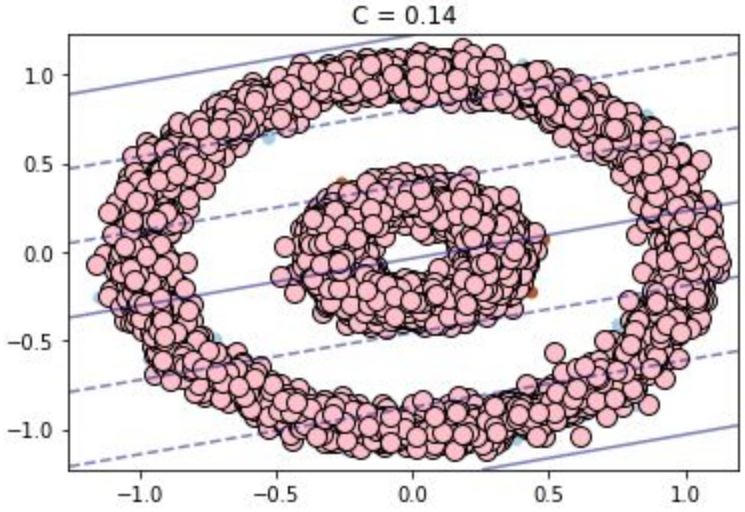


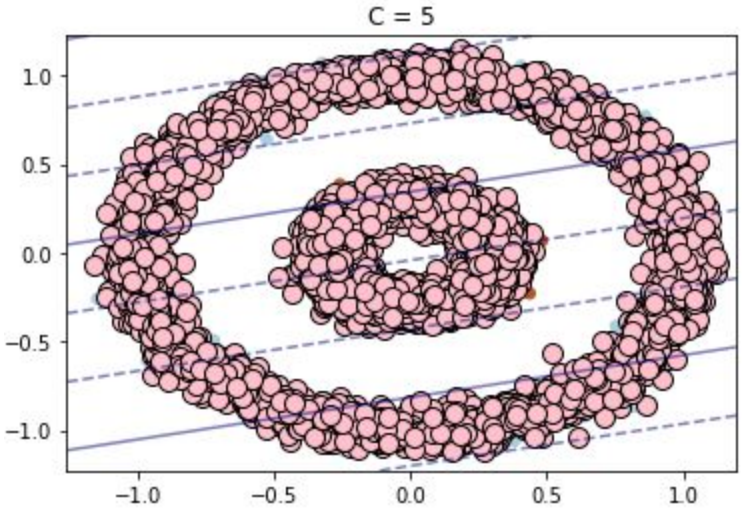
From the above visualization, it is clear that there are two class labels. The labels are concentrated along the circle with center (0,0) . the labels are not linearly classified . (i.e) there is no such line which separates the labels.

Answer 2-2):

For this Question I implemented my own userSVM class which take the *fit()* from the sklearn SVM and train the model after that it find the equation of the plane with the help of coefficient and intercept of plane since for this we uses linear SVM classifier so predict will classify the data based on the equation of the line.


C = 0.01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0


C = 0.14
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

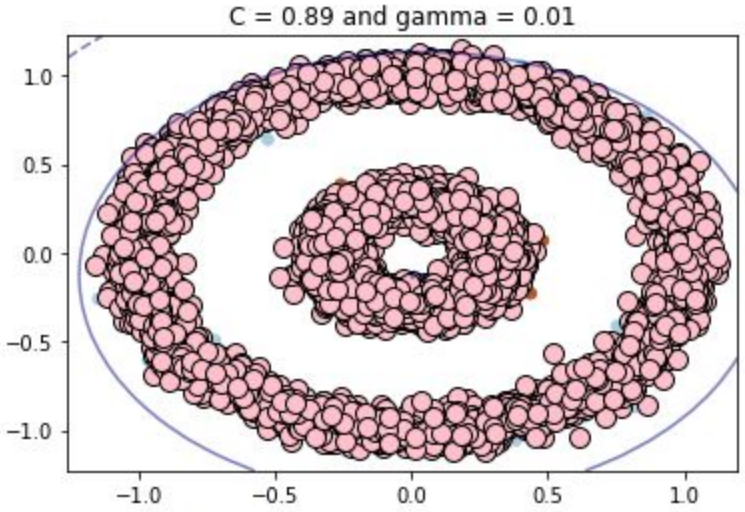

C = 5
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

```
Maximum accuracy Optimal C =  0.6865
Minimum accracy for poor C =  0.4995
Acuracy for random C =  0.6805
C value for Maximum Accuracy =  0.14
C value for Minimum Accuracy =  0.01
Any Random C value =  5
```

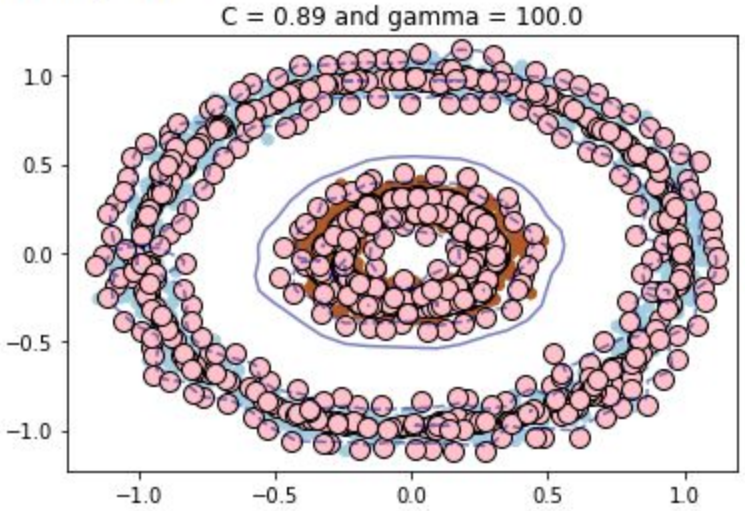
Answer 2-3):

```
Maximum accuracy = 1.0
Minimum accuracy = 0.4995
C = 1000.0 and Gamma = 100.0 value for Maximum accuracy
C = 0.89 and Gamma = 0.01 value for Minimum accuracy
Accuracy = 1.0 for random C = 5 and Gamma = 5
```

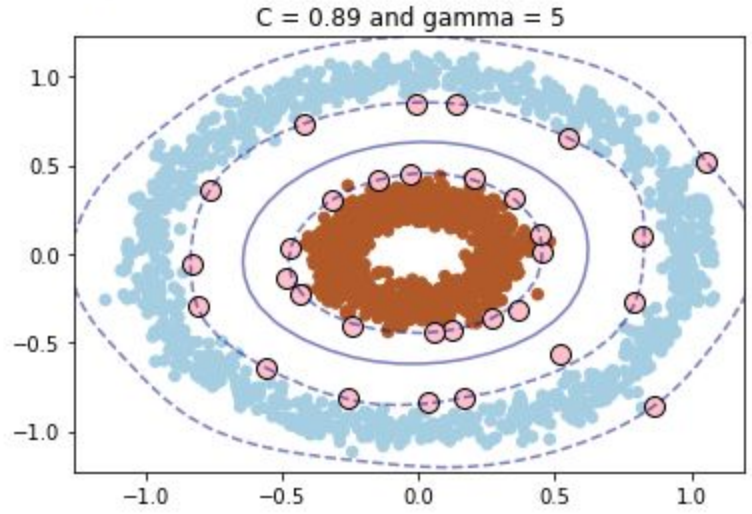
```
Accuracy = 0.4995
```



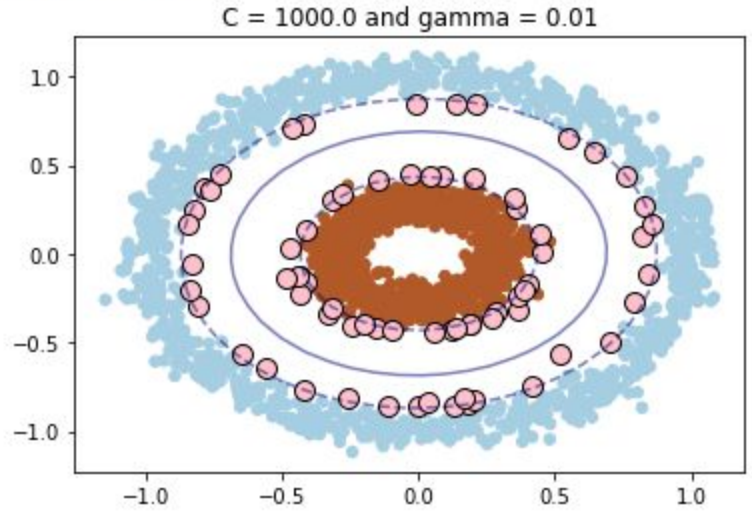

```
Accuracy = 1.0
```



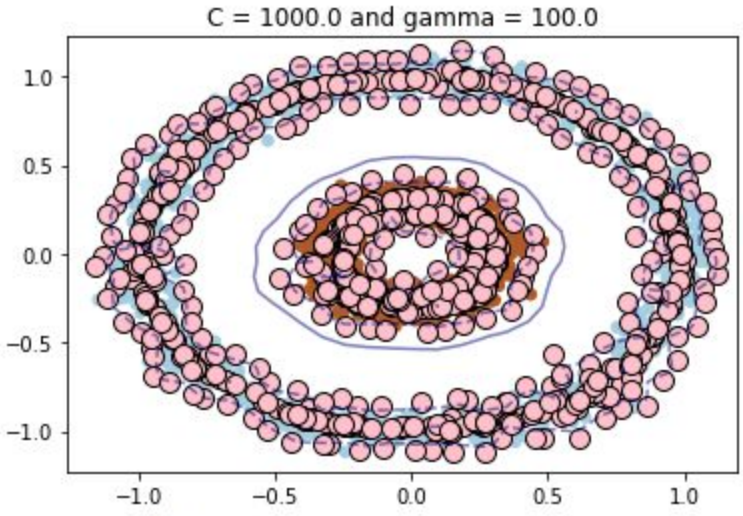
```
Accuracy = 1.0
```



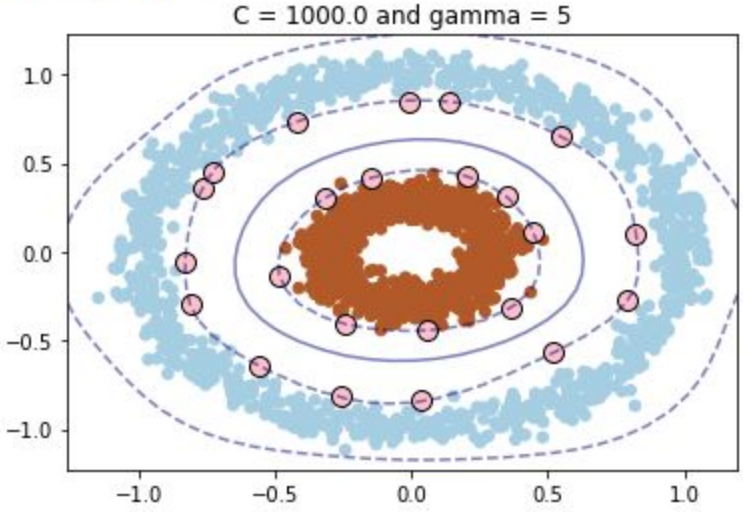

```
Accuracy = 1.0
```



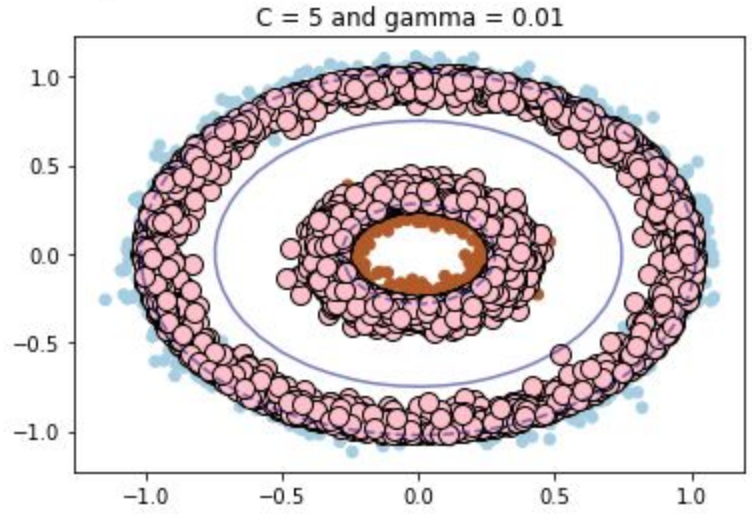
```
Accuracy = 1.0
```



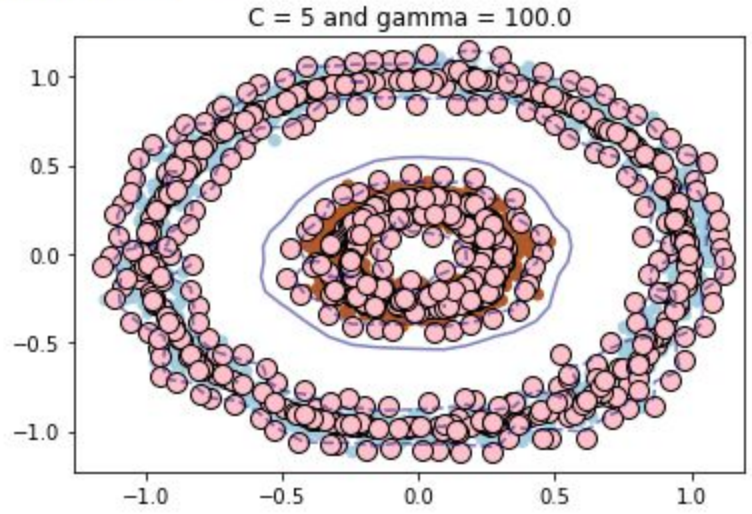

```
Accuracy = 1.0
```



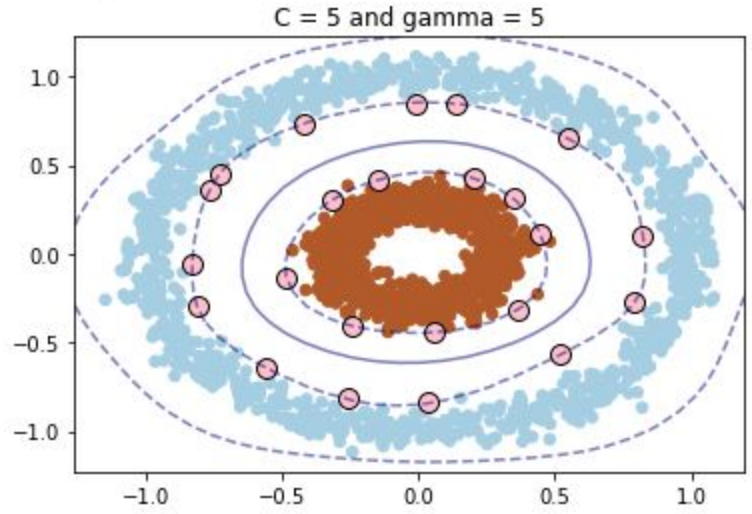
```
Accuracy = 1.0
```




```
Accuracy = 1.0
```



```
Accuracy = 1.0
```



Answer 2-4):

```
Accuracy for Linear SVM with C = 0.14 with sklearn 0.6865
Accuracy for Linear SVM with C = 0.14 with user Defined SVM 0.6865
Accuracy for RBF kernel SVM with C = 1000 and gamma =100 with sklearn 1.0
Accuracy for RBF kernel SVM with C = 1000 and gamma = 100 with user Defined SVM 1.0
```

Q3. Introduction

The given dataset had 2 features and 1 label.

The label had 3 distinct classes, so multi-class classification was to be applied.

The number of samples in dataset = 10000

The dataset already seemed to be normalized so no feature scaling required.

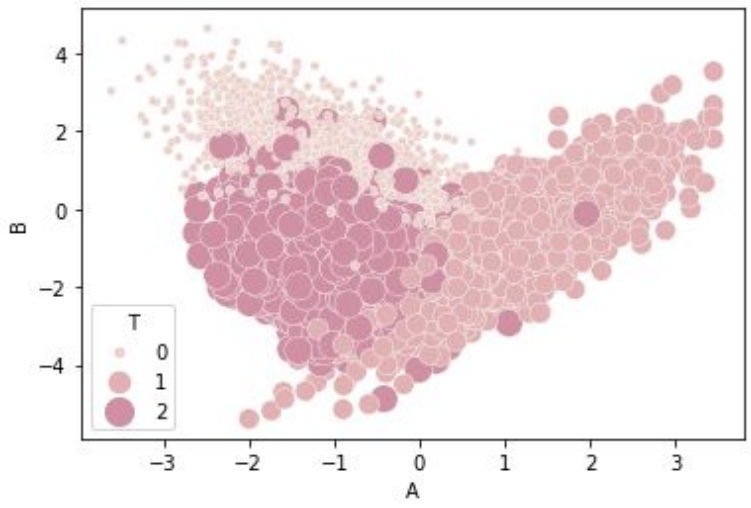
For this I used the fit() method of the svm library and implemented the predict method from scratch.

Approach for the MultiClassification:

Created a common class for multi-class classification which takes the parameter of kernel, gamma and C. It uses another self defined class which is used for binary classification. The specific approach for ovr and ovo are defined later in this report.

Q3. a. Visualize the (whole dataset) and state your observations

Ans:



It can be observed that there are three classes, with almost clear decision boundaries. Though the decision boundaries are not separated and overlap. As it involves class so classification algorithms must be applied, if we apply SVM we must use multi-class SVM. Other algorithms like KNN can also be applied easily here. If the value of A feature is high there is a high probability that the class would be class 1. For high values of B there is high probability that the sample will belong to class 0. And for low values of A and B features, there is a high probability that the sample will belong to class 2.

Approach: As the data set already had 2 dimensions of features, there was nothing much to be done and simply the scatter plot was made using seaborn.

Q3. b. Use a SVM with RBF kernel to classify this dataset through one vs rest approach. Perform the grid search for parameters C and γ to obtain their optimal values. Report the accuracy over the five folds along with the mean accuarcy. Also, report the mean class accuracy.

Ans:

Approach:

Defined a function to split the dataset into k-folds. Here I passed the value of k as 5 to the function and the function returned indices for the k folds which are to be taken for the dataset.

Defined a accuracy metric function which calculates the accuracy.

I created a function to perform grid search over the parameters of C and Gamma. The grid search function takes either 'ovo' or 'ovr' as a model and then applies the algorithm on the parameters.

The C parameter trades off correct classification of training examples against maximization of the decision function's margin. 'C' behaves as a regularization parameter in the SVM.

The behavior of the model is very sensitive to the gamma parameter. If gamma is too large, the radius of the area of influence of the support vectors only includes the support vector itself and no amount of regularization with C will be able to prevent overfitting.

The algorithm uses the binary SVM classifier defined by me as follows:

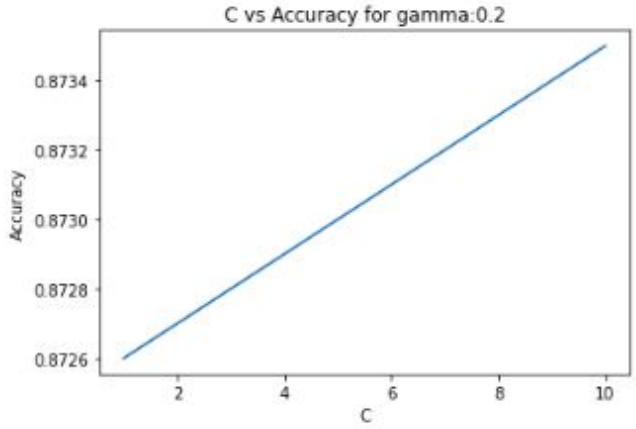
a. For training:
   - The number of models to be trained = Number of unique classes
   - For each model take one of the classes as 1 and make all other classes as 0.
   - It is to be noted that here it will not create class imbalance as there are only three distinct classes, if there were many classes, it would have created the situation of class imbalance.
   - Then use the binary classifier to train the model and store the model

b. For predicting:
   - While predicting, predict using all the models.
   - Only one of the models will be giving 1 as output and all others will classify it as 0. The model which classifies it as 1 will be the class of that sample, as each model represents a class. Assign the class which is represented by the model as the predicted class for the sample.

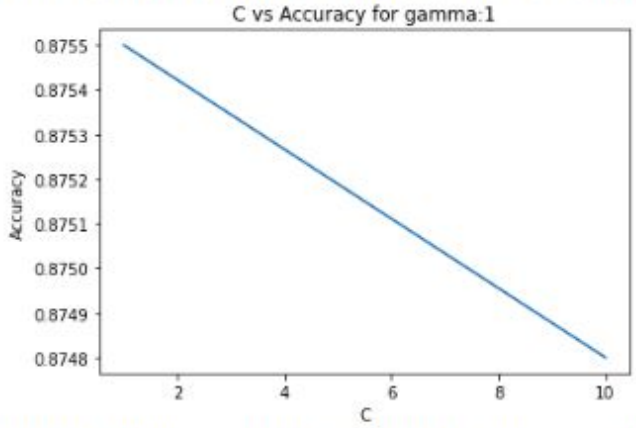
BEST ACCURACY: 0.874 FOR C: 1 AND GAMMA: 1

On Performing the Grid Search Got Results as Follows for One vs Rest:

```
Accuracy for Gamma: 0.2  and C: 1  and Fold:  1  is: 0.875
Accuracy for Gamma: 0.2  and C: 1  and Fold:  2  is: 0.8815
Accuracy for Gamma: 0.2  and C: 1  and Fold:  3  is: 0.8815
Accuracy for Gamma: 0.2  and C: 1  and Fold:  4  is: 0.8605
Accuracy for Gamma: 0.2  and C: 1  and Fold:  5  is: 0.8645
MEAN ACCURACY FOR GAMMA: 0.2  and C: 1  is  0.8725999999999999
Accuracy for Gamma: 0.2  and C: 10  and Fold:  1  is: 0.8755
Accuracy for Gamma: 0.2  and C: 10  and Fold:  2  is: 0.884
Accuracy for Gamma: 0.2  and C: 10  and Fold:  3  is: 0.8825
Accuracy for Gamma: 0.2  and C: 10  and Fold:  4  is: 0.8605
Accuracy for Gamma: 0.2  and C: 10  and Fold:  5  is: 0.865
MEAN ACCURACY FOR GAMMA: 0.2  and C: 10  is  0.8734999999999999
```




```
Accuracy for Gamma: 1  and C: 1  and Fold:  1  is: 0.876
Accuracy for Gamma: 1  and C: 1  and Fold:  2  is: 0.8835
Accuracy for Gamma: 1  and C: 1  and Fold:  3  is: 0.882
Accuracy for Gamma: 1  and C: 1  and Fold:  4  is: 0.866
Accuracy for Gamma: 1  and C: 1  and Fold:  5  is: 0.87
MEAN ACCURACY FOR GAMMA: 1  and C: 1  is  0.8755000000000001
Accuracy for Gamma: 1  and C: 10  and Fold:  1  is: 0.8745
Accuracy for Gamma: 1  and C: 10  and Fold:  2  is: 0.882
Accuracy for Gamma: 1  and C: 10  and Fold:  3  is: 0.8825
Accuracy for Gamma: 1  and C: 10  and Fold:  4  is: 0.867
Accuracy for Gamma: 1  and C: 10  and Fold:  5  is: 0.868
MEAN ACCURACY FOR GAMMA: 1  and C: 10  is  0.8747999999999999
```




```
BEST ACCURACY:  0.8755000000000001  FOR C: 1  AND GAMMA: 1
```

Q3. c. Use a SVM with RBF kernel to classify this dataset through one vs one approach. Perform the grid search for parameters C and γ to obtain their optimal values. Report the accuracy over the five folds along with the mean accuarcy. Also, report the mean class accuracy.

Ans:

Approach:

Similar to the function defined in Q3 b, I defined a function to split the dataset into k-folds.

Also similarly. defined a accuracy metric function which calculates the accuracy.

Similarly grid search was also performed but this time using 'ovo' as model.

The algorithm uses the binary SVM classifier defined by me as follows:

c. For training:
   - The number of models to be trained = (N * (N - 1)) / 2 i.e combination of two classes.

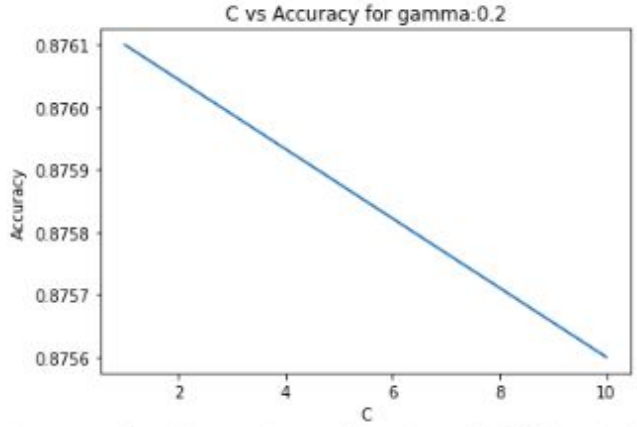
     Where N = Number of unique classes
   - For each model two classes are taken,
   - For each model take the subset of the dataset, take only those samples which have labels of either to the two classes taken at the time.
   - Train the model using the two classes picked and repeat it for every model.

d. For predicting:
   - While predicting, predict using all the models and store in a 2 dimensional list of dimensions (number of samples to be predicted * number of models)
   - Use the two dimensional list generated as a voting matrix, for each sample, take into considerations the class predicted by all the models, then take the majority class which is predicted as the final prediction for the sample. Repeat this process for all samples to get the predicted vector.

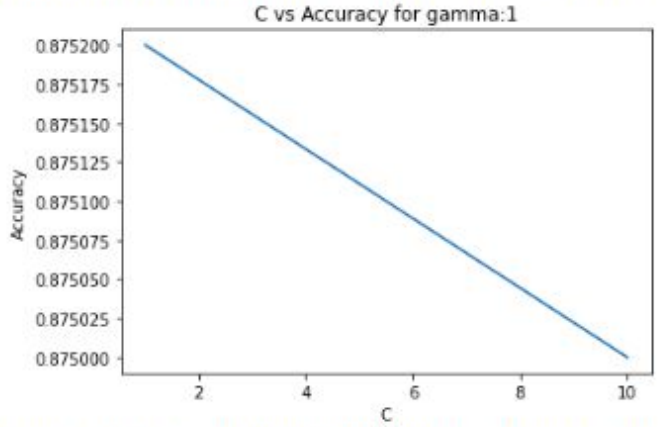
BEST ACCURACY: 0.8755 FOR C: 10 AND GAMMA: 1

On Performing the Grid Search Got Results as Follows for One vs One:

```
Accuracy for Gamma: 0.2  and C: 1  and Fold:  1  is: 0.874
Accuracy for Gamma: 0.2  and C: 1  and Fold:  2  is: 0.869
Accuracy for Gamma: 0.2  and C: 1  and Fold:  3  is: 0.88
Accuracy for Gamma: 0.2  and C: 1  and Fold:  4  is: 0.8775
Accuracy for Gamma: 0.2  and C: 1  and Fold:  5  is: 0.88
MEAN ACCURACY FOR GAMMA: 0.2  and C: 1  is  0.8760999999999999
Accuracy for Gamma: 0.2  and C: 10  and Fold:  1  is: 0.872
Accuracy for Gamma: 0.2  and C: 10  and Fold:  2  is: 0.8695
Accuracy for Gamma: 0.2  and C: 10  and Fold:  3  is: 0.8795
Accuracy for Gamma: 0.2  and C: 10  and Fold:  4  is: 0.878
Accuracy for Gamma: 0.2  and C: 10  and Fold:  5  is: 0.879
MEAN ACCURACY FOR GAMMA: 0.2  and C: 10  is  0.8756
```




```
Accuracy for Gamma: 1  and C: 1  and Fold:  1  is: 0.874
Accuracy for Gamma: 1  and C: 1  and Fold:  2  is: 0.869
Accuracy for Gamma: 1  and C: 1  and Fold:  3  is: 0.877
Accuracy for Gamma: 1  and C: 1  and Fold:  4  is: 0.8765
Accuracy for Gamma: 1  and C: 1  and Fold:  5  is: 0.8795
MEAN ACCURACY FOR GAMMA: 1  and C: 1  is  0.8752000000000001
Accuracy for Gamma: 1  and C: 10  and Fold:  1  is: 0.873
Accuracy for Gamma: 1  and C: 10  and Fold:  2  is: 0.8675
Accuracy for Gamma: 1  and C: 10  and Fold:  3  is: 0.877
Accuracy for Gamma: 1  and C: 10  and Fold:  4  is: 0.8775
Accuracy for Gamma: 1  and C: 10  and Fold:  5  is: 0.88
MEAN ACCURACY FOR GAMMA: 1  and C: 10  is  0.875
```




```
BEST ACCURACY:  0.8760999999999999  FOR C: 1  AND GAMMA: 0.2
```

Q3. d. Use a SVM with RBF kernel to classify this dataset through one vs one approach. Perform the grid search for parameters C and γ to obtain their optimal values. Report the accuracy over the five folds along with the mean accuarcy. Also, report the mean class accuracy.

Ans:

Approach:

Got the best parameters as C=1 and **γ**=1 using Q3b and Q3c.

For one vs one classification I used the SVC from the svm class with decision_function_shape='ovo'.

Accuracy obtained using 'ovo': 0.8795

For one vs rest classification I used OneVsRestClassifier from sklearn.multiclass.

Accuracy obtained using 'ovr': 0.8785

Conclusions from Q3:

- It is observed that 'ovr' and 'ovo' perform almost similar, this is also due the fact that as there are only three unique classes so ovr doesn't suffer from class imbalance here.
- The accuracies obtained by user defined multi-class SVM is the same as SVM using the library. The minor differences are due to the fact that different samples are taken at different times affecting the accuracies in very small magnitude.
- For very low values of C and gamma and for very high values of C and gamma, the accuracy falls. This is due to underfitting on increasing the C too much.
- For high values of gamma, the radius of the area of influence of the support vectors only includes the support vector itself and so the model overfits.

CONTRIBUTIONS

Reshan Faraz	Waquar shamsi
Q1-3 (Marks = 15)	Q1-1 (Marks: 5 + 15 = 20)
Q1-4(Marks = 15)	Q1-2 (Marks: 10)
Q2(Marks = 50)	Q3(Marks = 50)
Total Marks = 80	Total Marks = 80

References :

https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

https://stats.stackexchange.com/questions/313660/what-are-the-support-vectors-in-a-support-vector-machine

https://www.analyticsvidhya.com/blog/2016/03/pca-practical-guide-principal-component-analysis-python/

https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

https://www.cs.toronto.edu/~kriz/cifar.html

https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/