**Name:** Mohit Mohit

**BU ID:** U33745431

CS 542

**Programming Report**

# Question 1.

Develop code for training and testing an SVM classifier with nonlinear kernel. You are welcome to use either formulation described in the textbook (chapter 7). You cannot use an SVM library to complete this assignment. You can use quadratic programming library if you like. Using your implementation of the SVM classifier, compare multi-class classification performance of two different voting schemes:

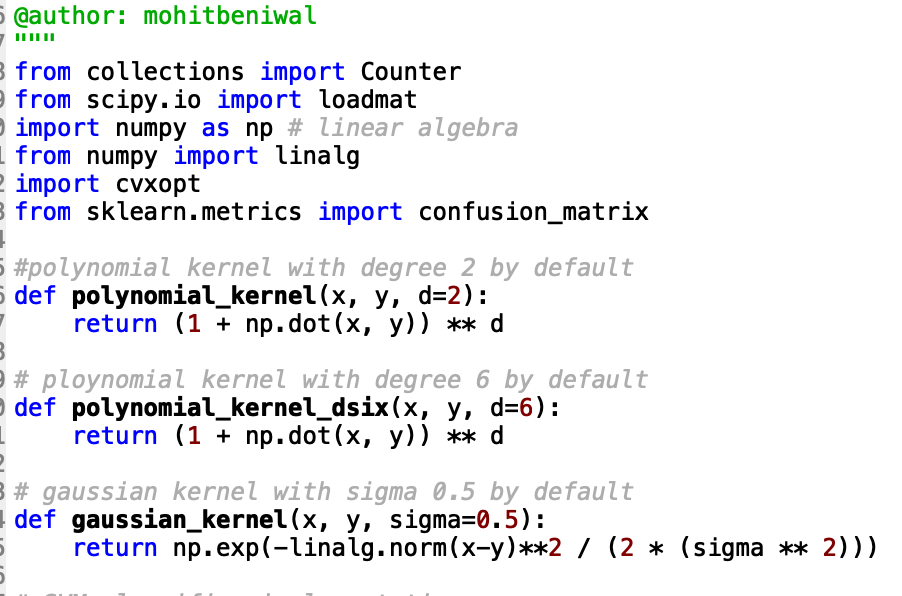
1. “one versus the rest”
2. “one versus one”

Be sure to specify your voting scheme using a method described in the book. To analyze accuracy, you will find it helpful to produce and analyze the multiclass confusion matrix in addition to examining the overall error rate.

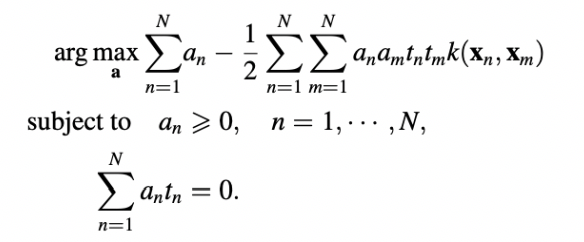
**Solution**

I have implemented the code for SVM in python. It will take care of finding support vectors and margins of the supplied points. I have used polynomial kernel with degree 2 and 6. Also, I have tried using Gaussian kernel with sigma 0.5.

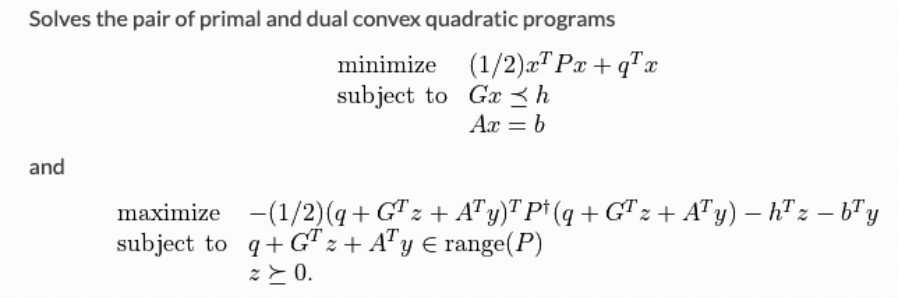
Code Kernel Functions:-



In Fit function I have solved the Dual Optimization problem.



To use cvxopt library we need to convert the above equation in a form as below.



Here,

G: diagonal matrix of of -1s of size m\*m

A: label vector of size m\*1

q: vector of size m\*1

h: vector of zeroes of size m\*1

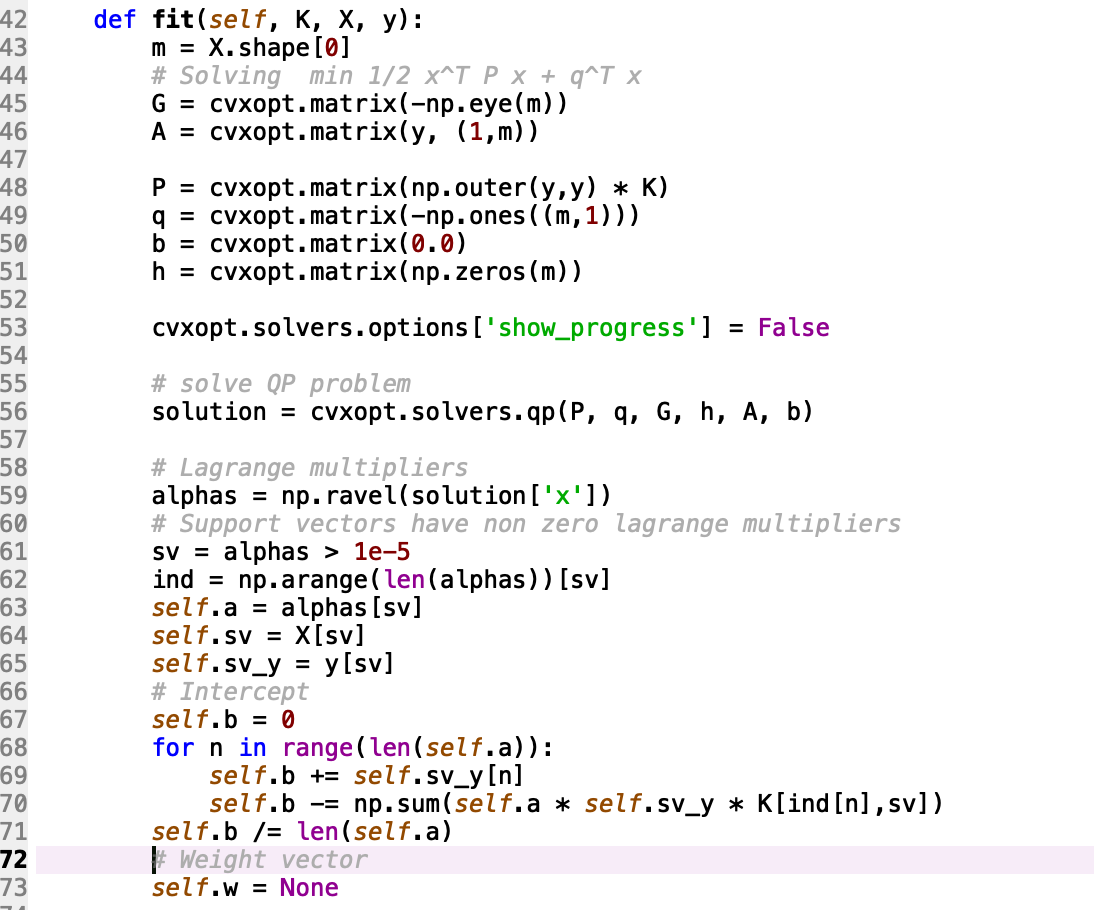
b: scalar with value as 0.

Passing these parameters to the *quadratic programming* interface of cvxopt, we get the solution for x

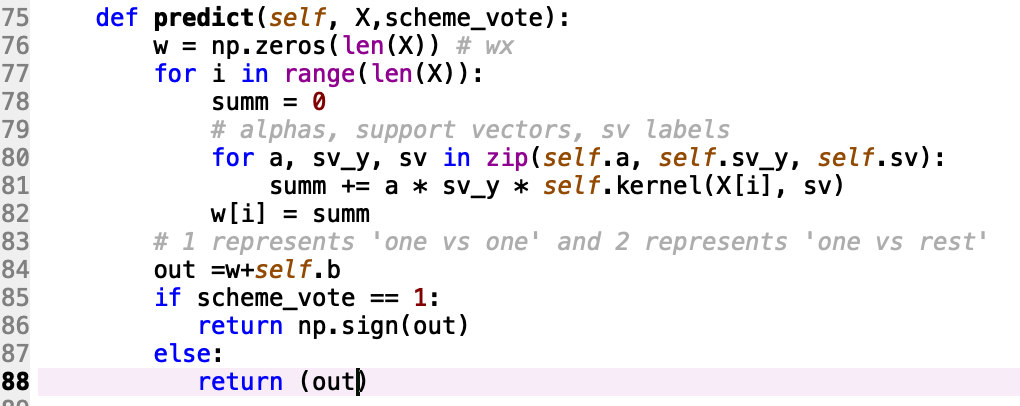
We then determine the set of support vectors *S* by finding the indices such that αm>0 and intercept *b* is calculated by:

b = Ts − ∑αmTmK(Xm, Xs)

Code. For fit function:



In predict function, I classified X\_test by using alphas, support vectors and intercept with the values of the fit function. This function takes an argument ‘way’ which defines the voting scheme that can be one\_vs\_one or one\_vs\_rest to predict.



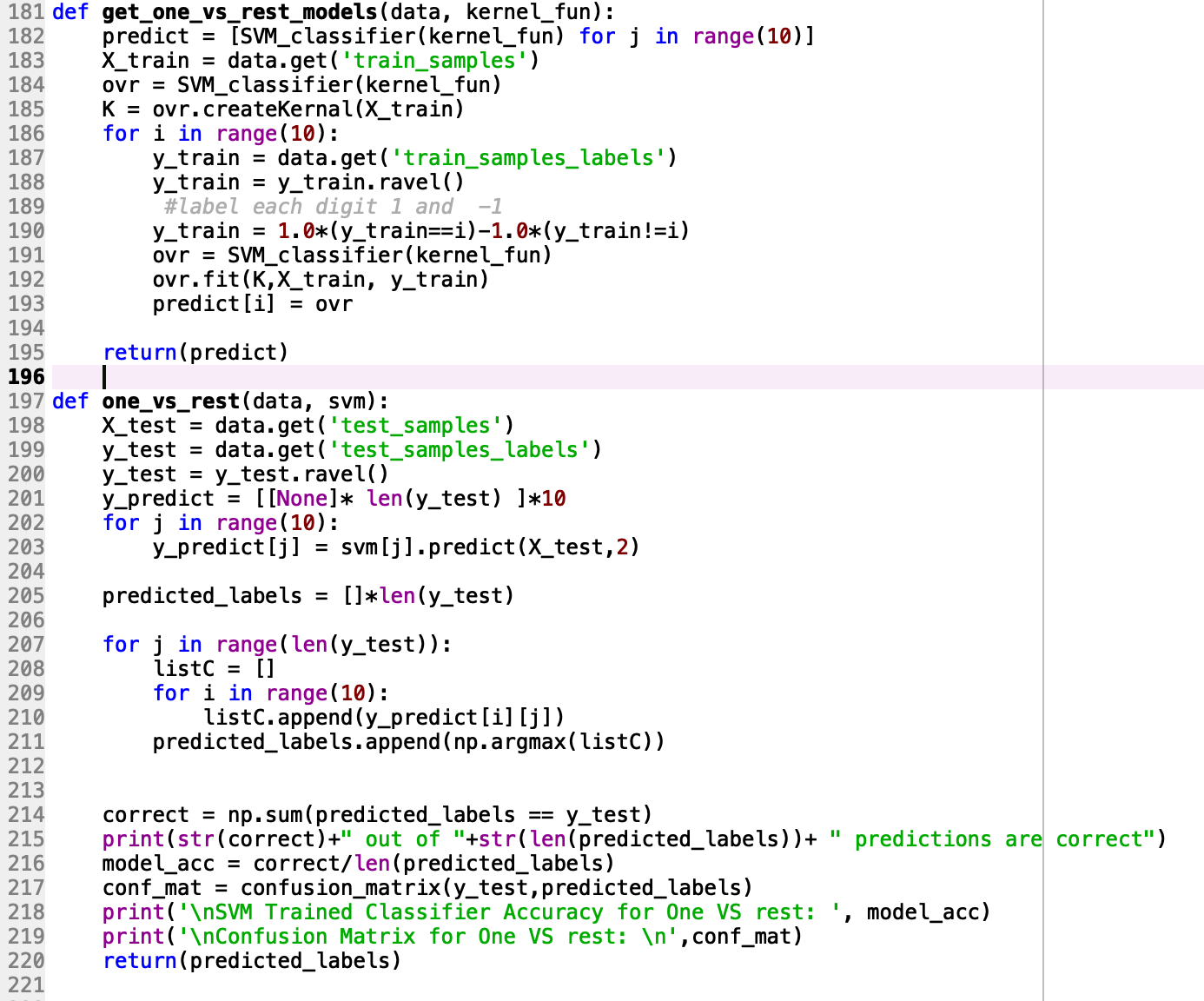
Above functions are present in the SVM\_classifier class.

## One vs Rest:

To perform One vs Rest we need to build and train out model keeping one out of (0,1…9) class as one and rest which is left as one another class. While predicting one class is defined as +1 and rest classes are defined with label -1

we will take the actual confidence score for each classifier to decide the label by choosing the maximum confidence score.

I have divided the task in 2 parts by making a function to create models and other to predict and plot the accuracy and confusion matrix.

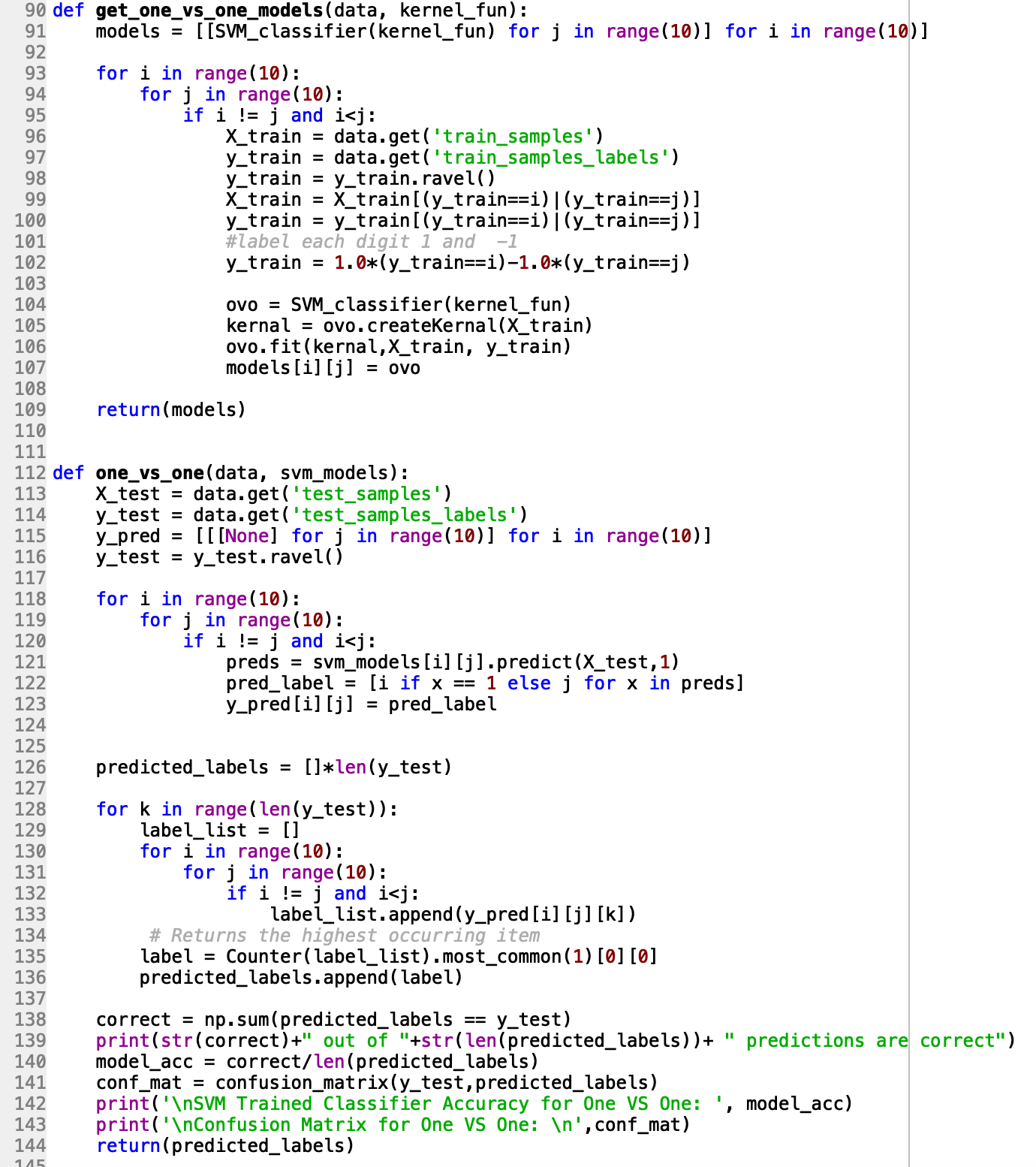


## One vs One:

For this strategy, we will pick classes in pair where both the classes are not same. By doing so we will end up creating 45 models for distinct pair of classes.

We will assign +1 to one class and -1 to the other class, later we will take the count of the which class got the highest count as +1 or -1 and accordingly we will finalize the prediction for the data point.

Same as One vs Rest I have divided the task into 2 parts.

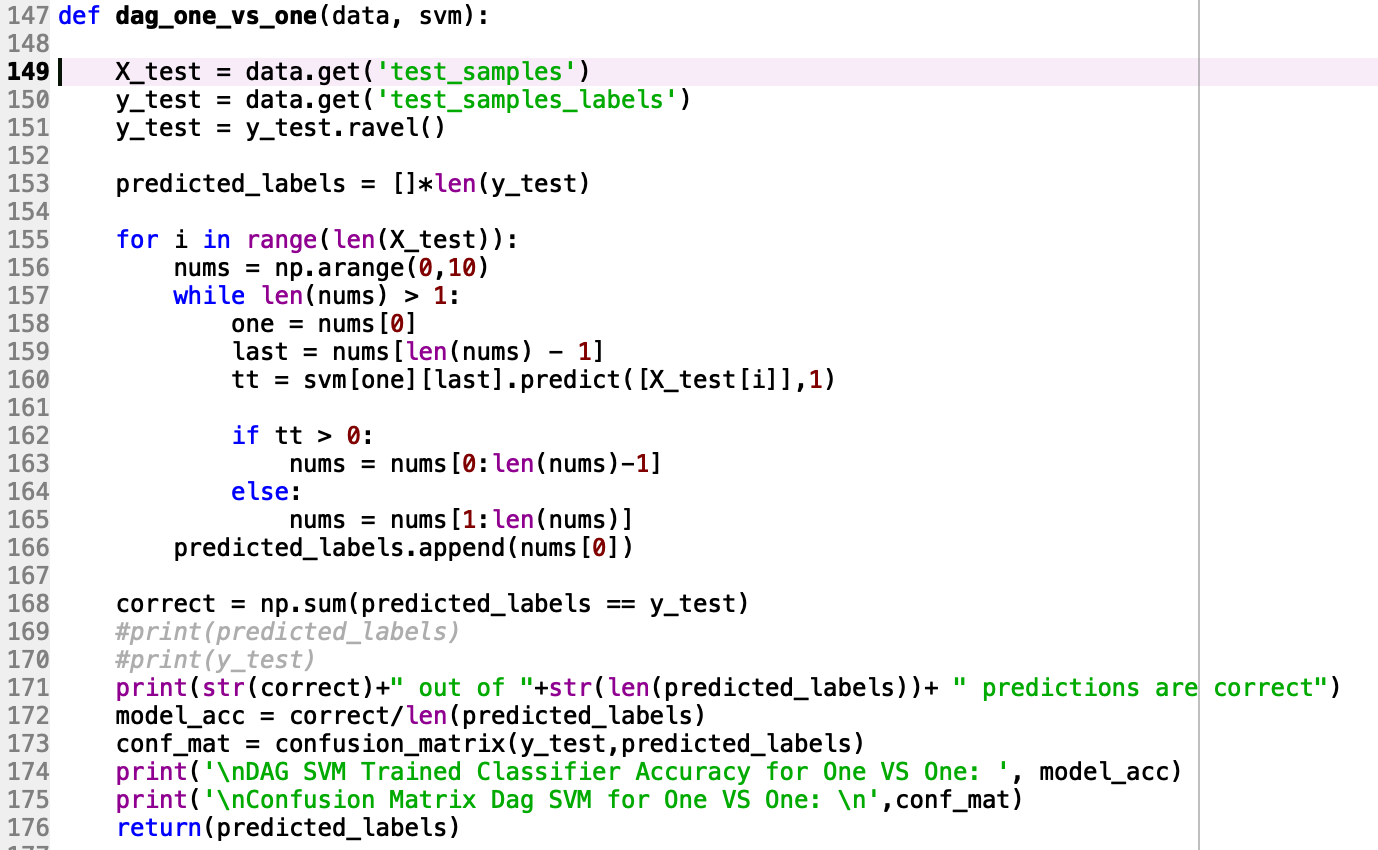


# Question 2.

## DAGSVM (One vs One):

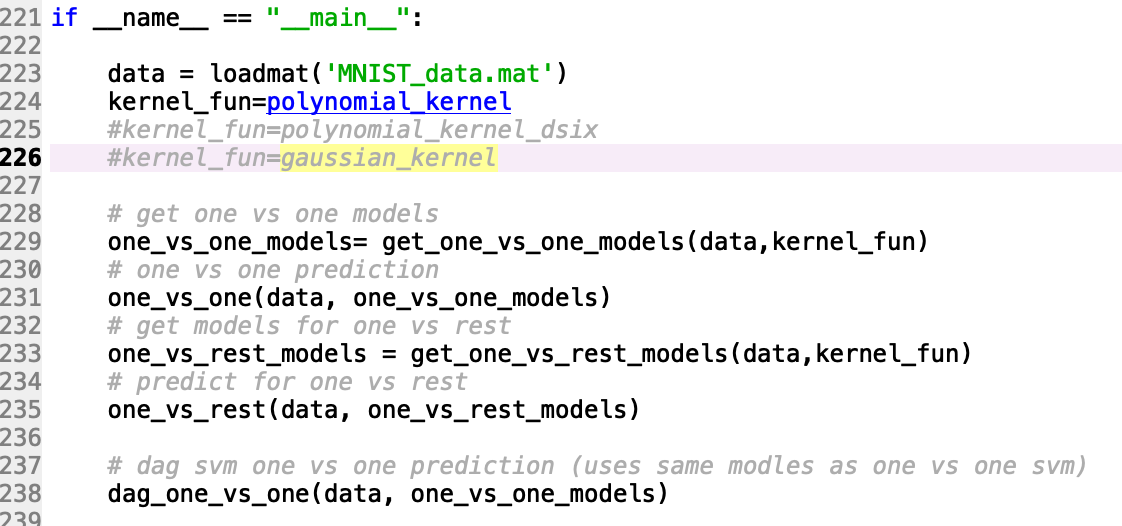
In DAGSVM with one vs one strategy is almost same as the one vs one task with one difference that we will take the 1st and last class in the beginning e.g (0,9)

Now, if the predicted values comes out in favor of 0 in that case we will remove the class ‘9’ from the class set for this data point as this data point can never be classified same this class.



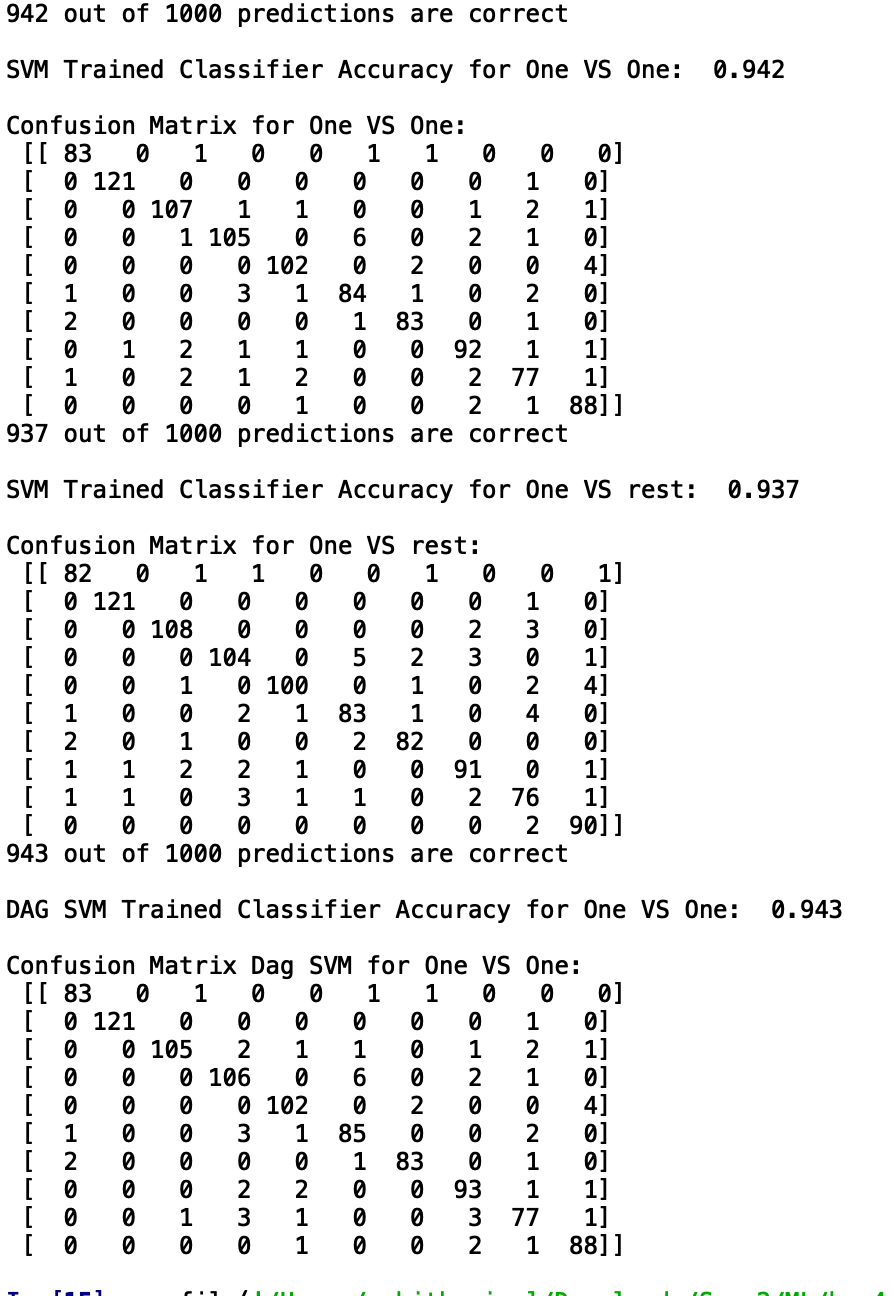
Main method:

Please comment and uncomment the kernel\_fun while testing the code.

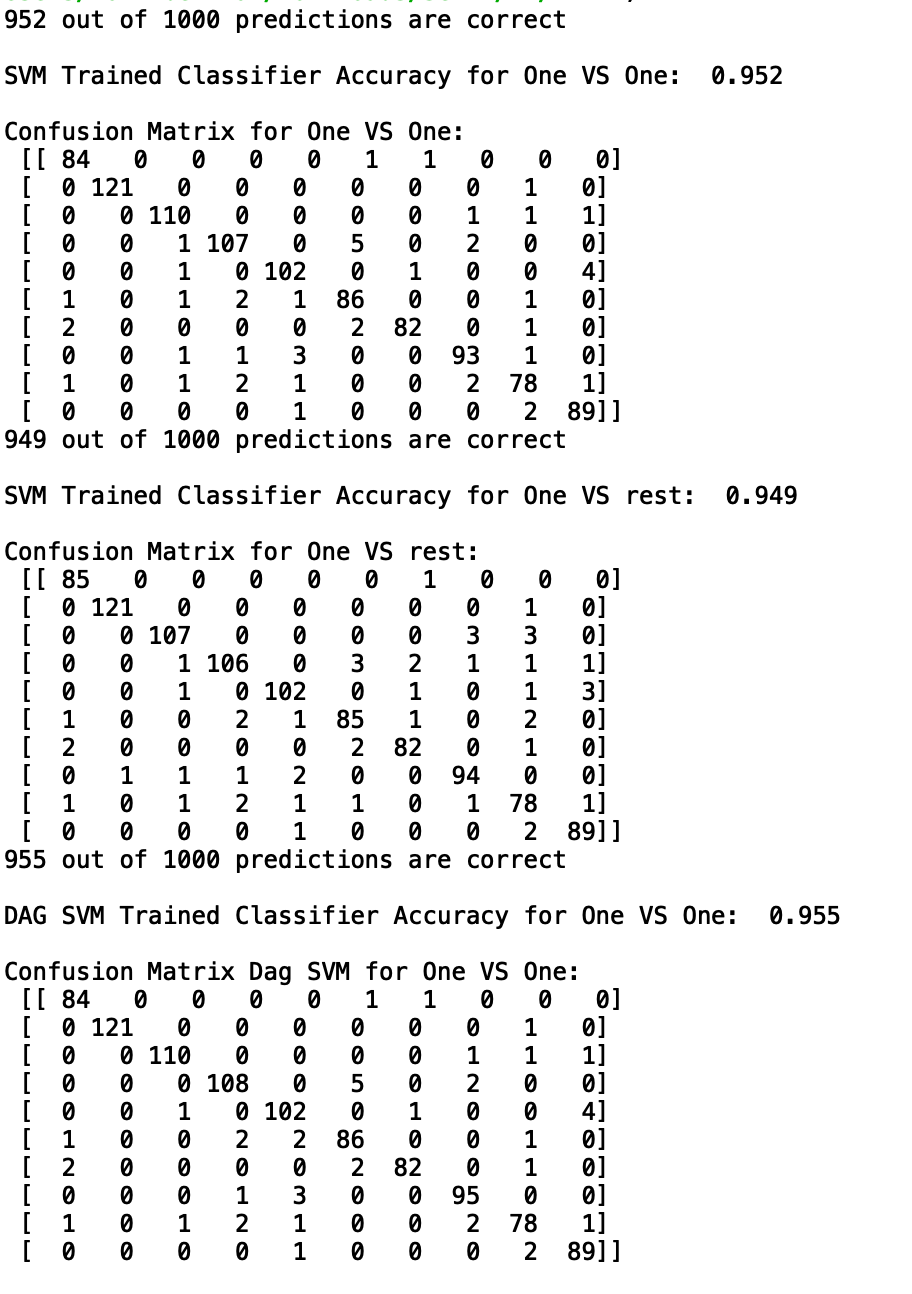


Results:

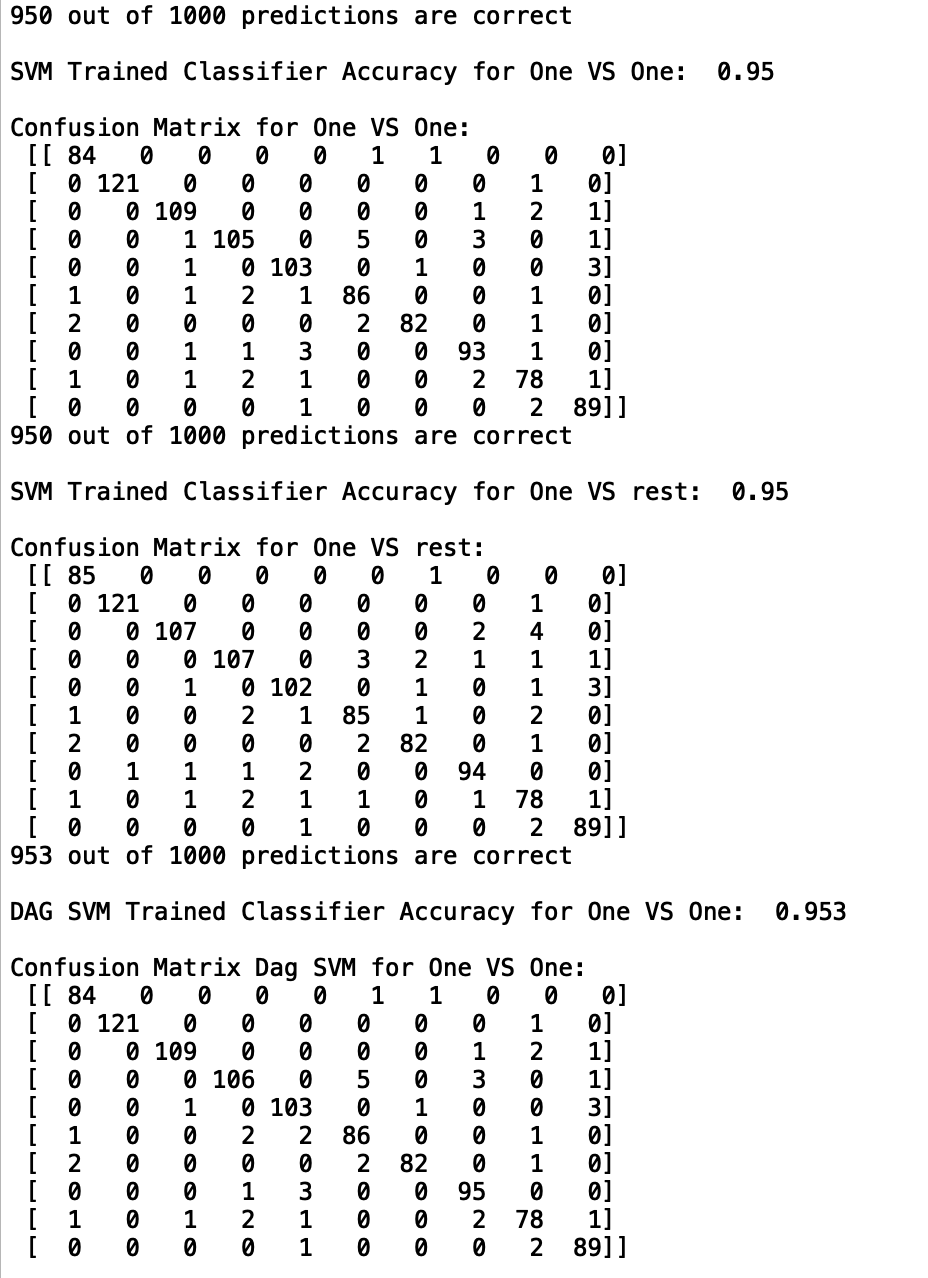
Results for One vs One , One vs Rest and DAGSVM (One vs One ) using ﻿polynomial\_kernel with degree 2.



Results for One vs One , One vs Rest and DAGSVM (One vs One ) using ﻿polynomial\_kernel with degree 6.



Results for One vs One , One vs Rest and DAGSVM (One vs One ) using ﻿gaussian\_kernel with sigma 0.5.



Performance Comparison:-

I have tried using Gaussian function to improve the

|  |  |  |  |
| --- | --- | --- | --- |
|  | One vs Rest | One vs One | DAGSVM(one vs one ) |
| Polynomial (d = 2) | 93.7 | 94.2 | 94.3 |
| Polynomial (d = 6) | 94.9 | 95.2 | 95.5 |
| Gaussian | 95 | 95 | 95.3 |

# Question 3

I have used the polynomial kernel with degree = 6 with DAGSVM one vs one scheme. The result is better (95.5% accuracy) than the baseline implementation which is 94.3. Whereas, the implementation with Gaussian kernel and sigma = 0.5, I was getting 95.3 accuracy, which is also better than the baseline implementation.

**REFERENCES**

1. Bishop, Pattern Recognition And Machine Learning - Springer  2006, Chapter 7,

Sparse Kernel Machines.

1. <https://cvxopt.org/userguide/coneprog.html#quadratic-programming>
2. <https://courses.csail.mit.edu/6.867/wiki/images/a/a7/Qp-cvxopt.pdf>
3. <https://courses.media.mit.edu/2006fall/mas622j/Projects/aisen-project/>
4. <http://www.mit.edu/~9.520/spring09/Classes/multiclass.pdf>