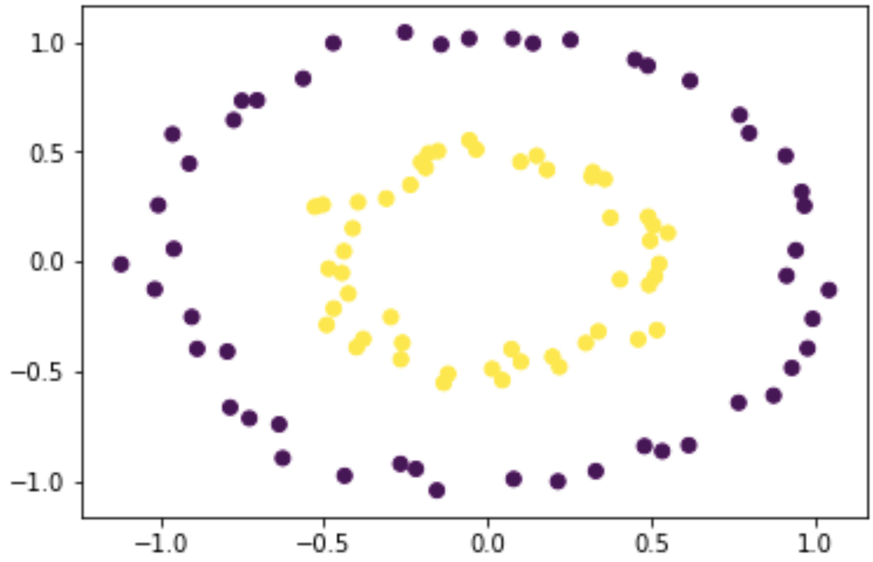
CSE-343 MACHINE LEARNING

Assignment 2 (Report & Theory)

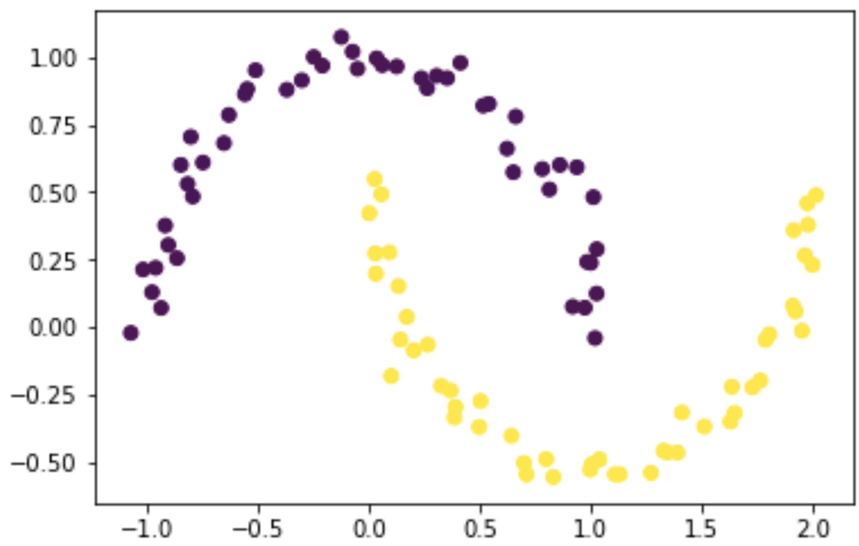
Nikita Mehrotra-PHD18013

Q1: PLOTS:

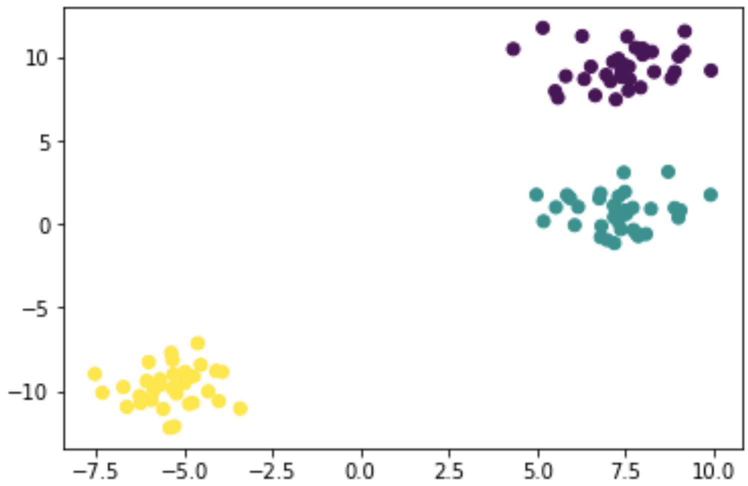
Dataset1 :



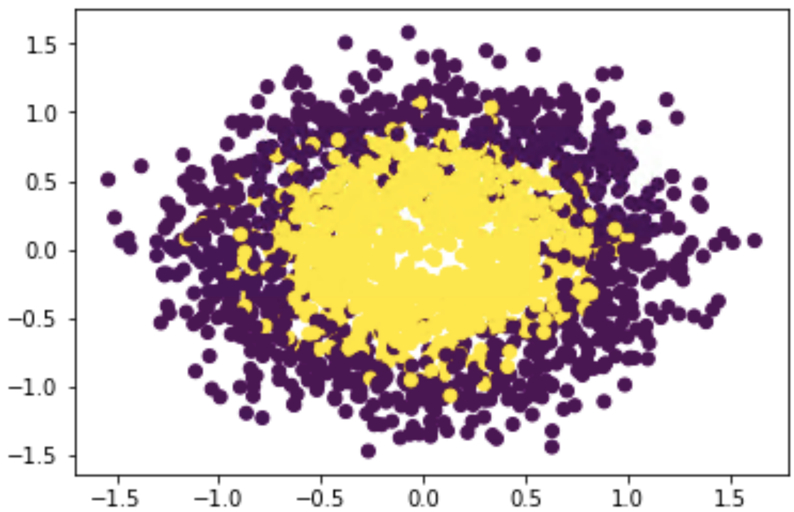
Dataset 2:



Dataset 3:



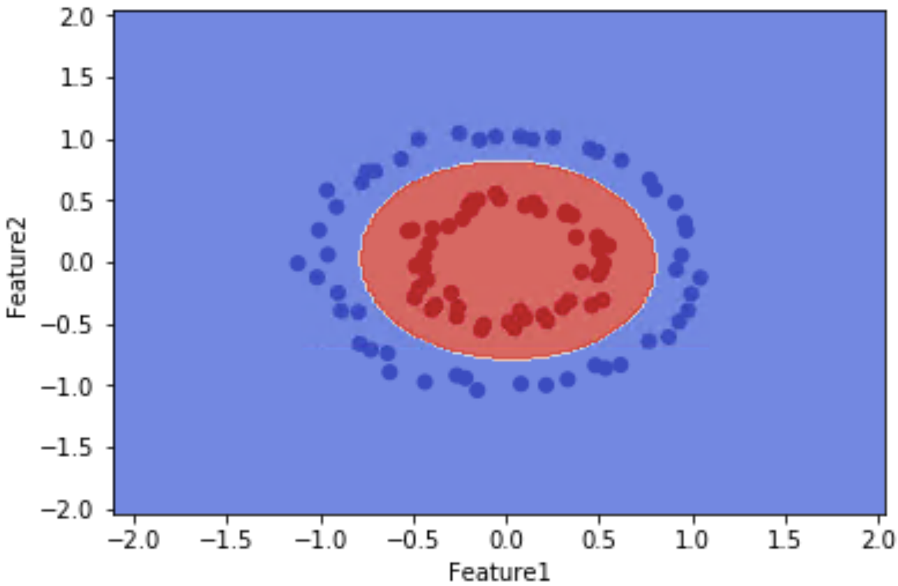
Dataset 4 :



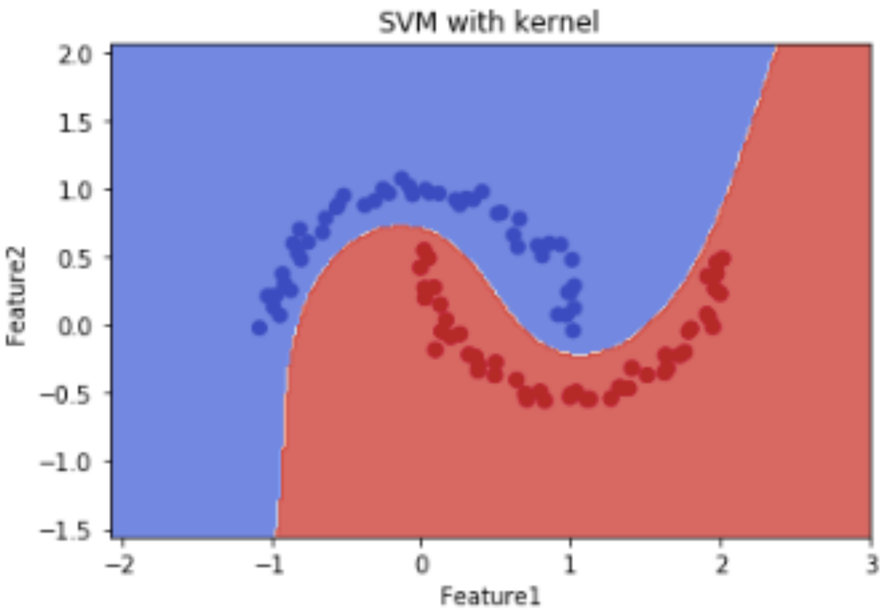
From the above 4 data sets, data set is linearly separable.

Q2:

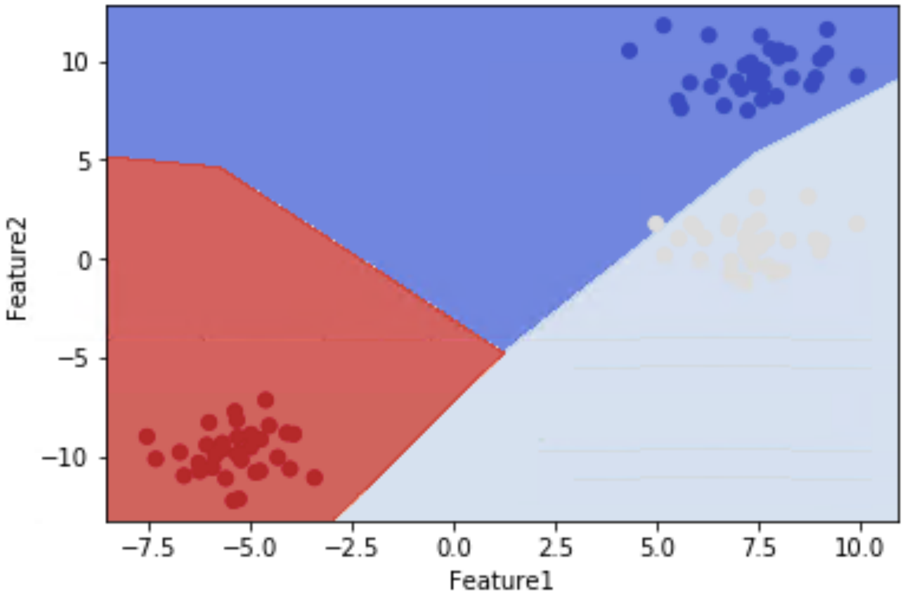
Dataset 1: Since the dataset resembles to concentric circles, We have used polynomial kernel with degree 2 to make data linearly separable



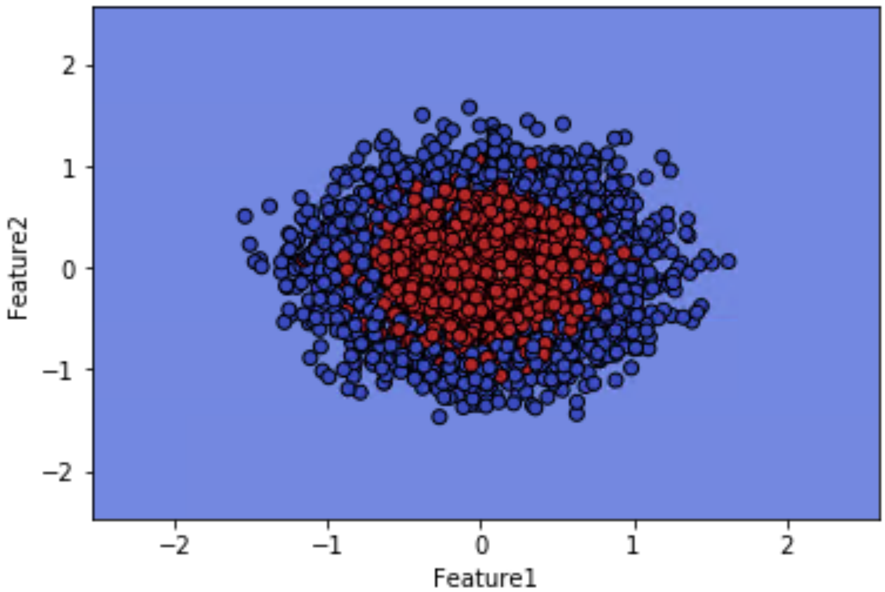
Dataset 2: We have used polynomial kernel with degree 3 to make data linearly separable



Dataset 3 : We have used linear kernel to make data separable



Dataset 4: We have used polynomial kernel with degree 2 to make data separable

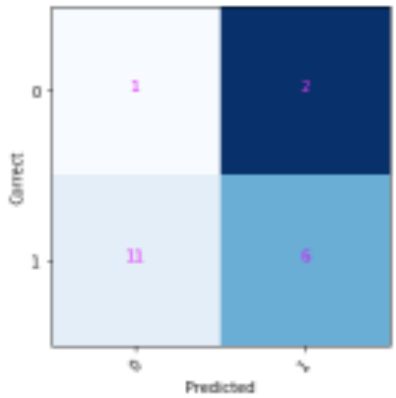


Q3. **One Vs Rest Classifier (Linear Kernel)**

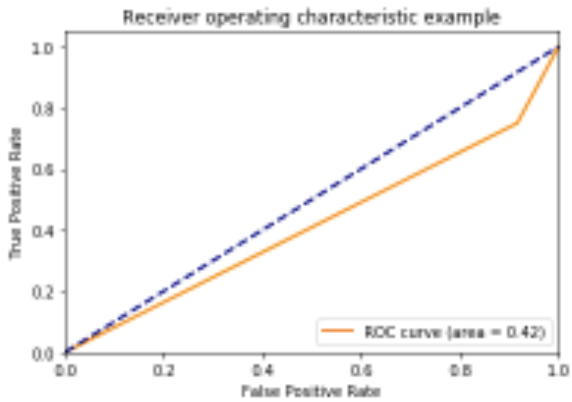
Confusion Matrix, ROC Curve, Support Vector and Margin separating the hyperplane, Accuracy score and F1-Score for each of the dataset is shown below:

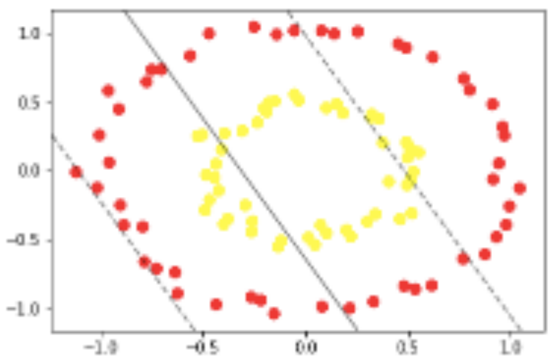
Dataset1:

```
For DATASET 1
{'C': 0.01, 'gamma': 1}
Predicted  0   1
Correct
0          1  11
1          2   6
```




```
<Figure size 432x288 with 0 Axes>
```

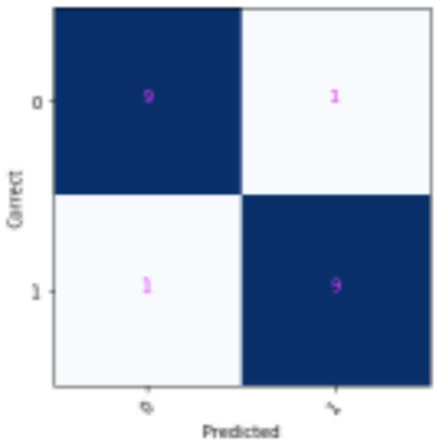




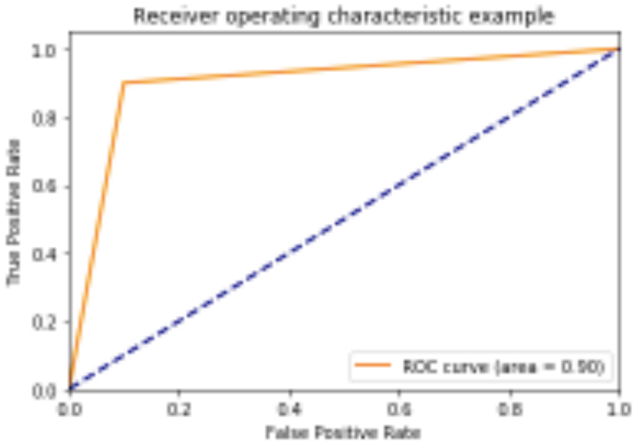
```
ACCURACY 0.35
F1-Score 0.30666666666666664
```

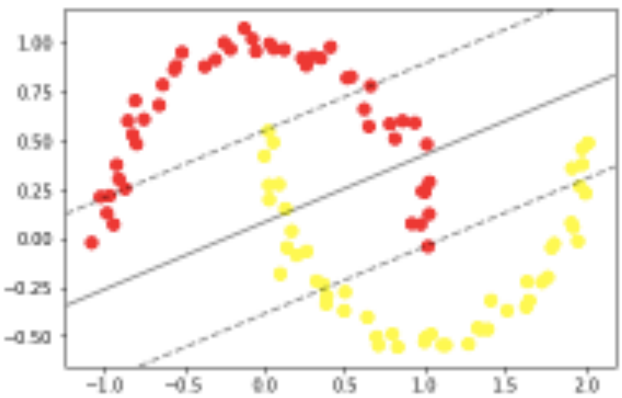
Dataset2:

```
For DATASET 2
{'C': 1, 'gamma': 1}
Predicted  0  1
Correct
0          9  1
1          1  9
```




```
<Figure size 432x288 with 0 Axes>
```

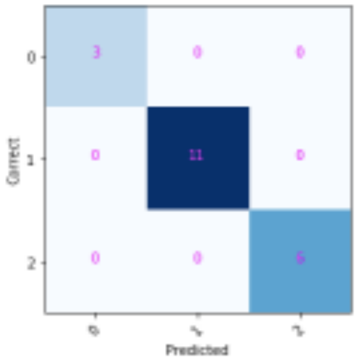




```
ACCURACY 0.9
F1-Score 0.9
```

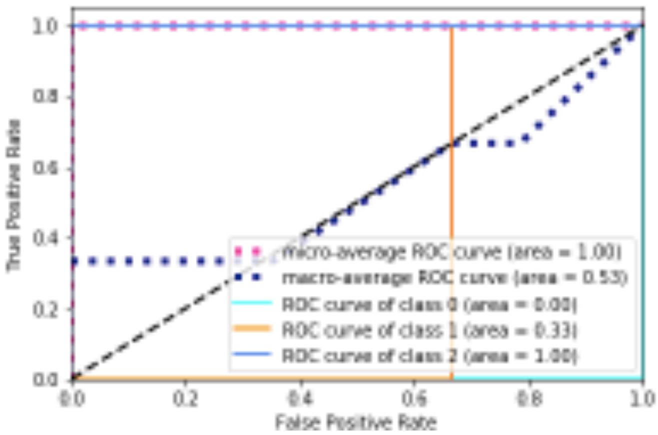
Dataset 3 :

```
{'C': 0.1, 'gamma': 0.01}
Predicted  0   1  2
Correct
0          3   0  0
1          0  11  0
2          0   0  6
```

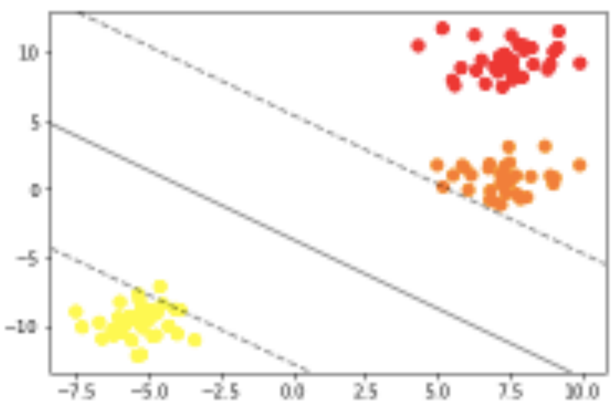


<Figure size 432x288 with 0 Axes>

```
[0.         0.35294118 1.         1.         ]
[0.         0.66666667 0.66666667 1.         ]
[0.         0.         0.78571429 1.         ]
```



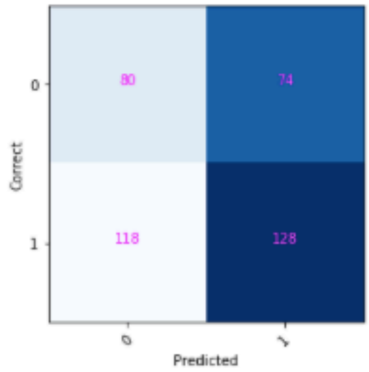
<Figure size 432x288 with 0 Axes>



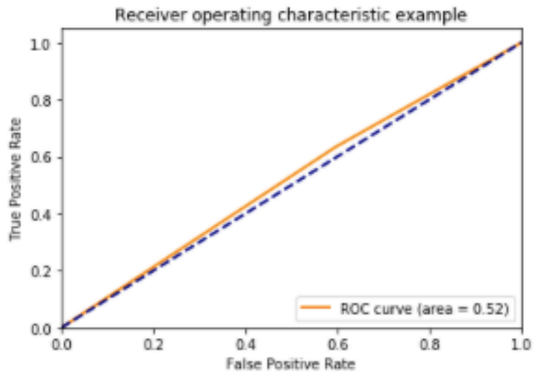
```
ACCURACY 1.0
F1-Score 1.0
```

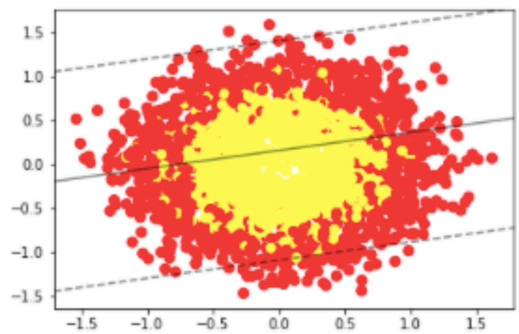
Dataset4:

```
For DATASET 4
{'C': 100, 'gamma': 1}
Predicted    0     1
Correct
0           80   118
1           74   128
```




```
<Figure size 432x288 with 0 Axes>
```

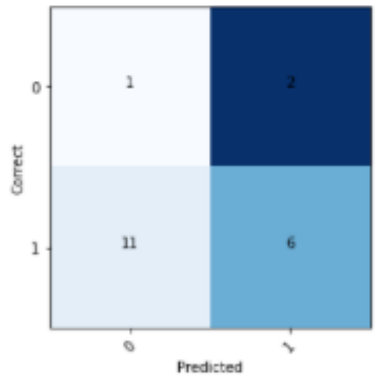




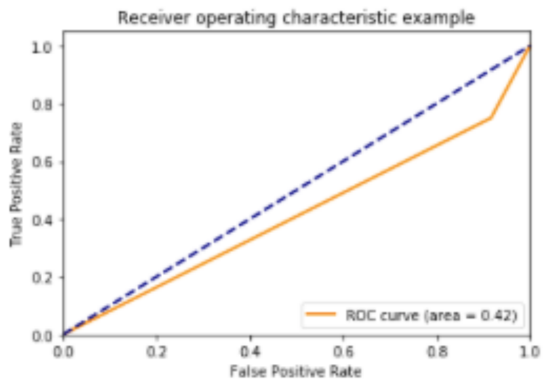
```
ACCURACY 0.52
F1-Score 0.5129870129870131
```

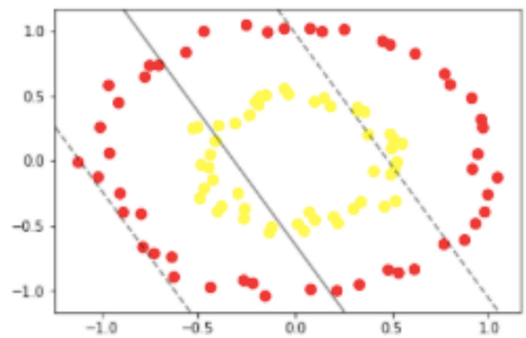
One Vs One Classifier (Linear Kernel):
Dataset 1:

```
for DataSet 1
{'C': 0.01, 'gamma': 1}
Predicted  0   1
Correct
0          1  11
1          2   6
```




```
<Figure size 432x288 with 0 Axes>
```

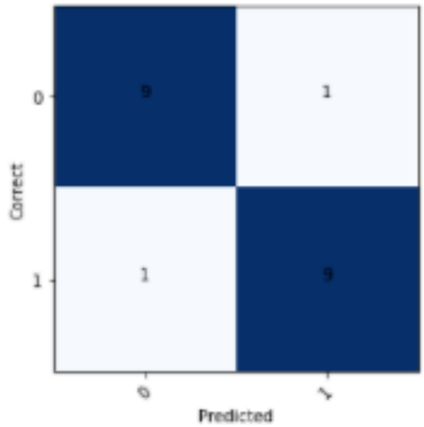




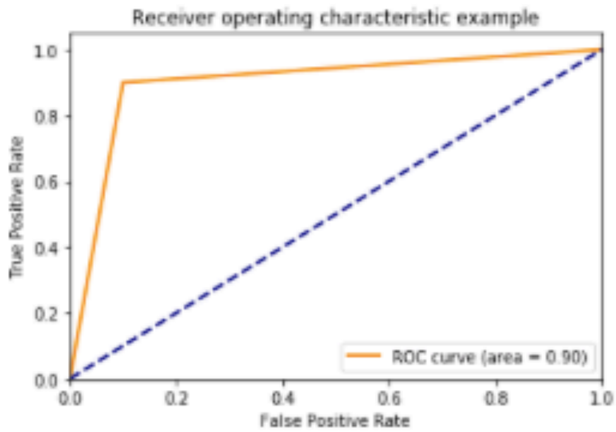
```
Accuracy 0.35
F1-Score 0.30666666666666664
```

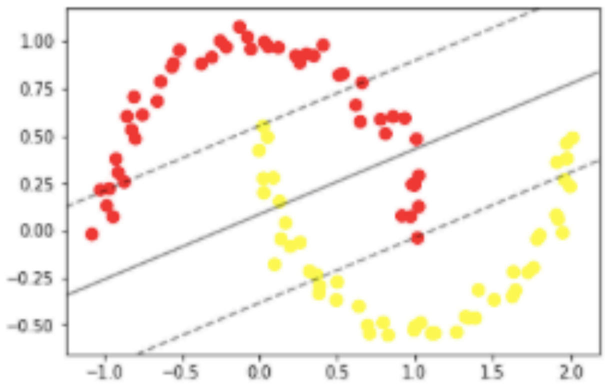
Dataset 2:

```
for DataSet 2
{'C': 1, 'gamma': 1}
Predicted  0  1
Correct
0          9  1
1          1  9
```




```
<Figure size 432x288 with 0 Axes>
```

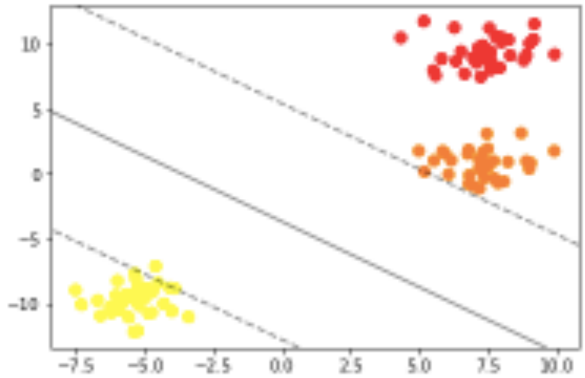




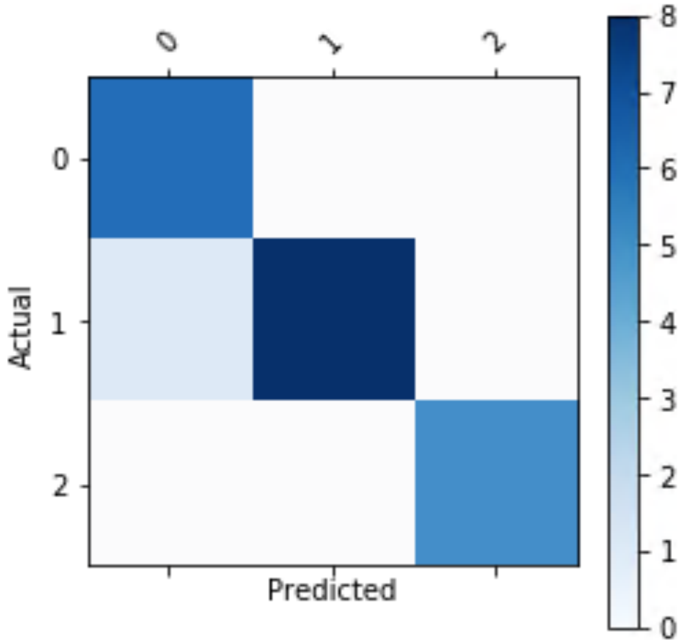

```
Accuracy 0.9
F1-Score 0.9
```

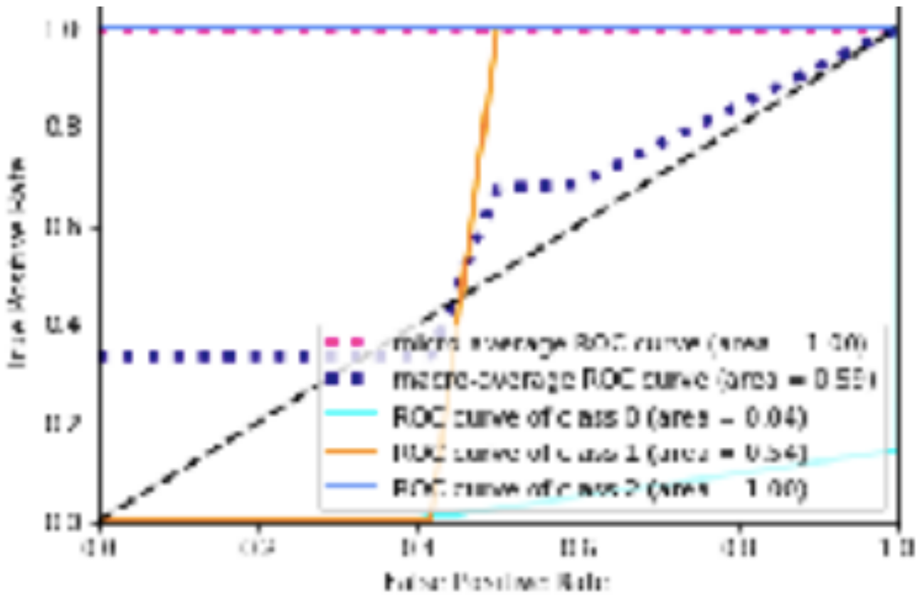
Dataset 3:

```
for DataSet 3
```



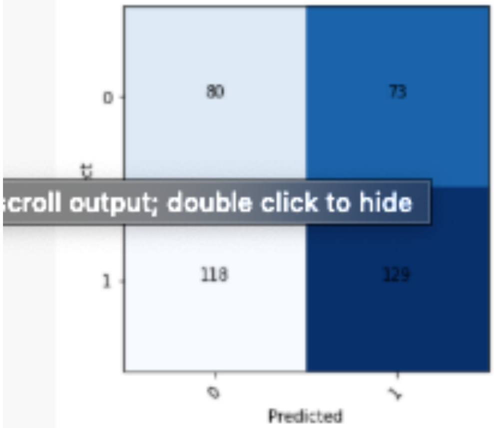
```
Accuracy 0.55
F1-Score 0.29333333333333333
```



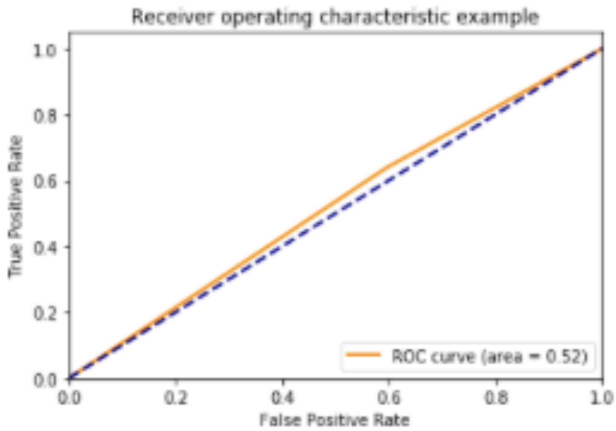


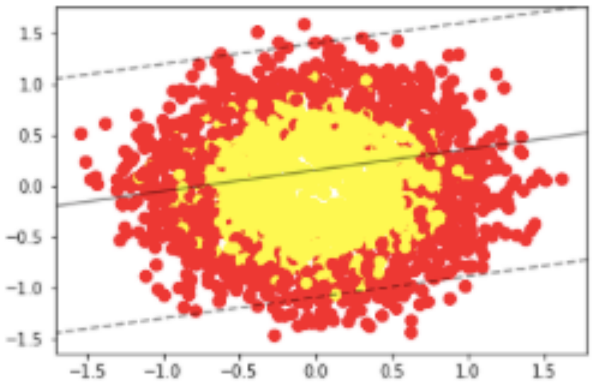
Dataset 4:

```
{'C': 100, 'gamma': 1}
Predicted    0    1
Correct
0           80  118
1           73  129
```




```
<Figure size 432x288 with 0 Axes>
```

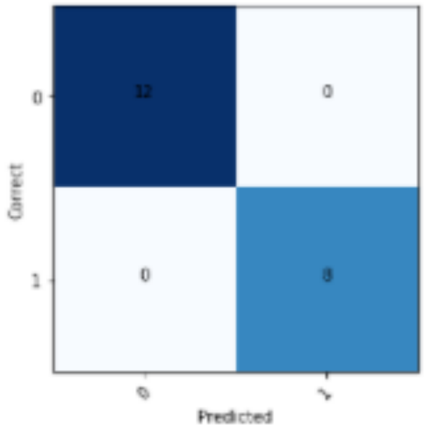




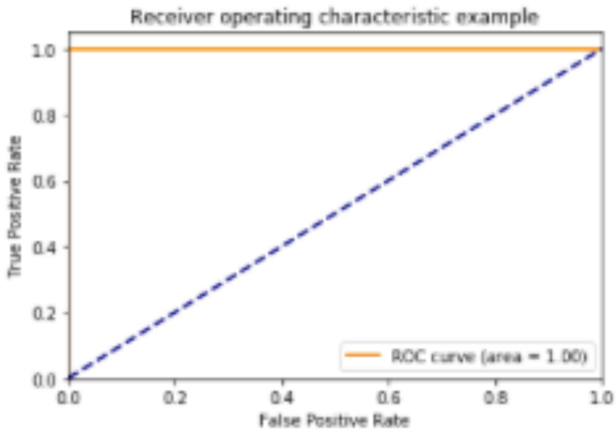
```
Accuracy 0.5225
F1-Score 0.51522535041466
```

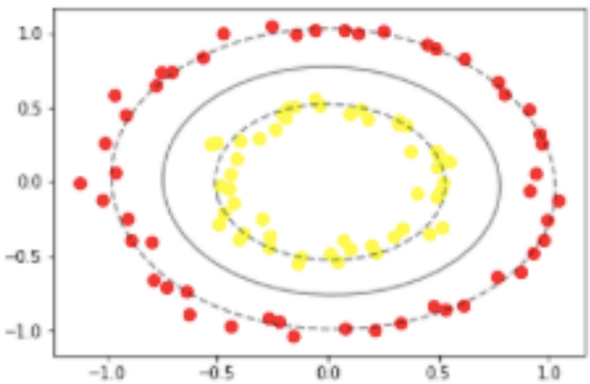
One Vs Rest Classifier (RBF Kernel):
Dataset 1:

```
For DATASET 1
{'C': 0.01, 'gamma': 1}
Predicted    0  1
Correct
0           12  0
1            0  8
```




```
<Figure size 432x288 with 0 Axes>
```

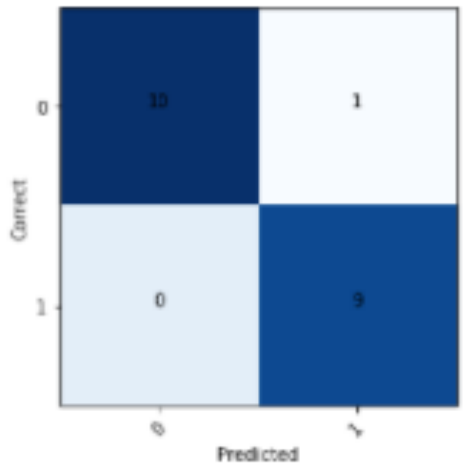




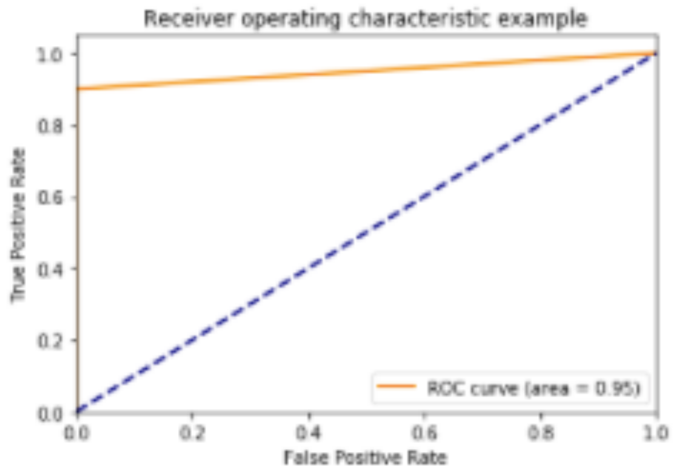

```
Accuracy 1.0
F1-Score 1.0
```

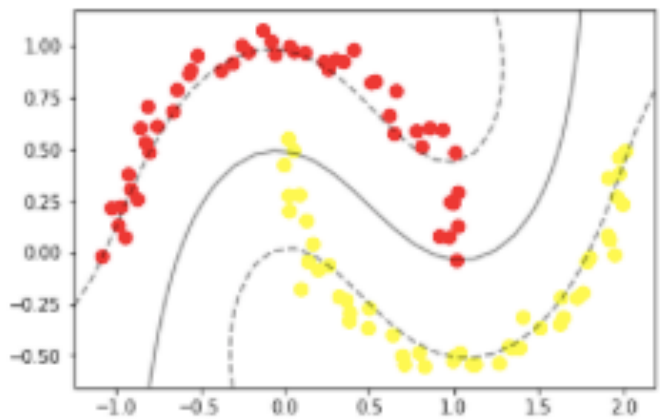
Dataset 2:

```
For DATASET 2
{'C': 1, 'gamma': 1}
Predicted    0  1
Correct
0           10  0
1            1  9
```




```
<Figure size 432x288 with 0 Axes>
```

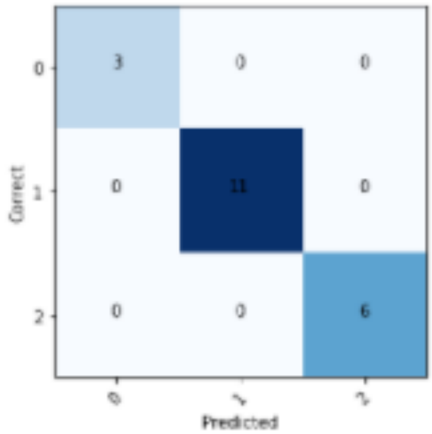




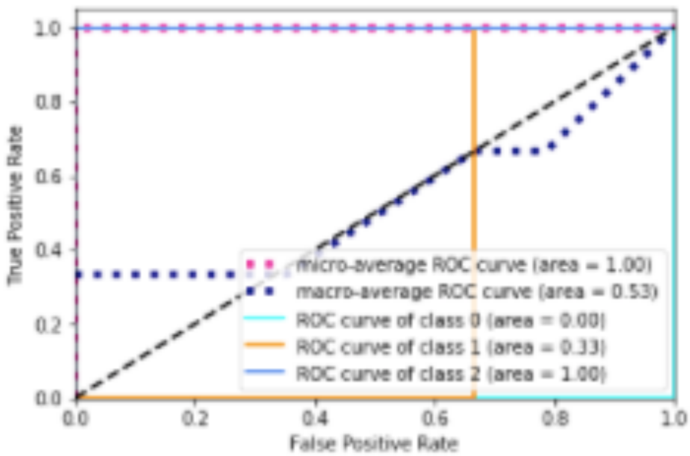
```
Accuracy 0.95
F1-Score 0.949874686716792
```

Dataset 3:

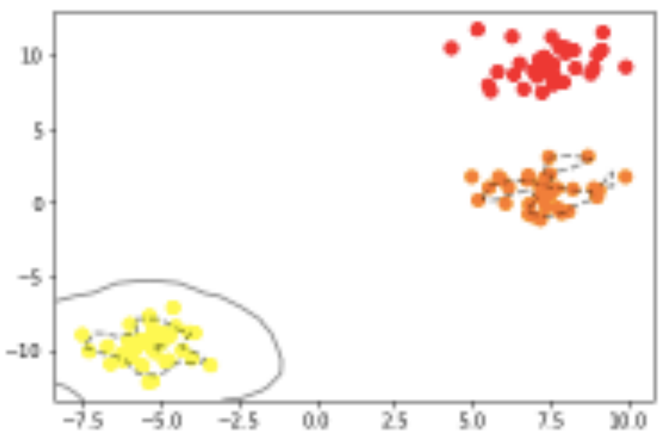
```
{'C': 0.1, 'gamma': 0.01}
Predicted  0   1  2
Correct
0          3   0  0
1          0  11  0
2          0   0  6
```




```
<Figure size 432x288 with 0 Axes>

[0.         0.35294118 1.         1.         ]
[0.         0.66666667 0.66666667 1.         ]
[0.         0.         0.78571429 1.         ]
```

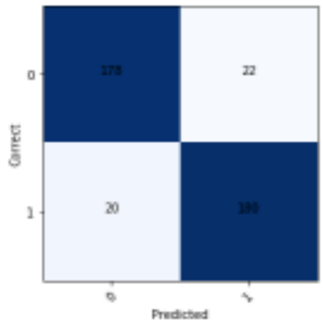



```
Accuracy 1.0
F1-Score 1.0

<Figure size 432x288 with 0 Axes>
```

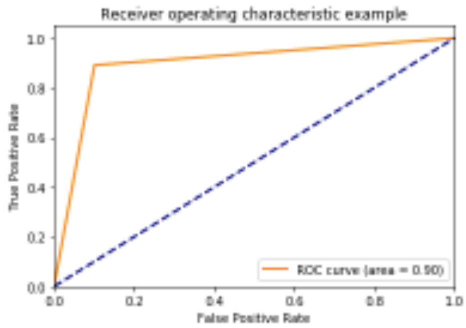


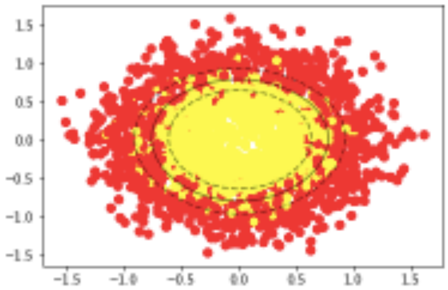
Dataset 4:

```
{'C': 100, 'gamma': 1}
for DataSet 4
Predicted     0    1
Correct
0           178   20
1            22  180
```




```
<Figure size 432x288 with 0 Axes>
```



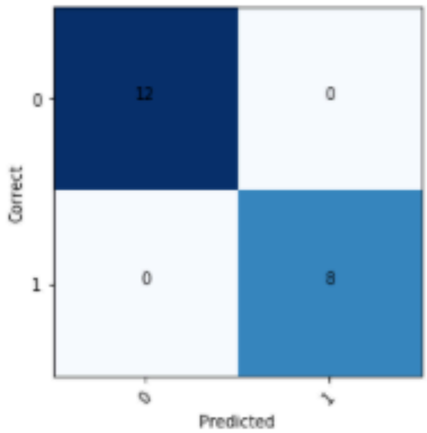


```
Accuracy 0.895
F1-Score 0.8949973749343734
```

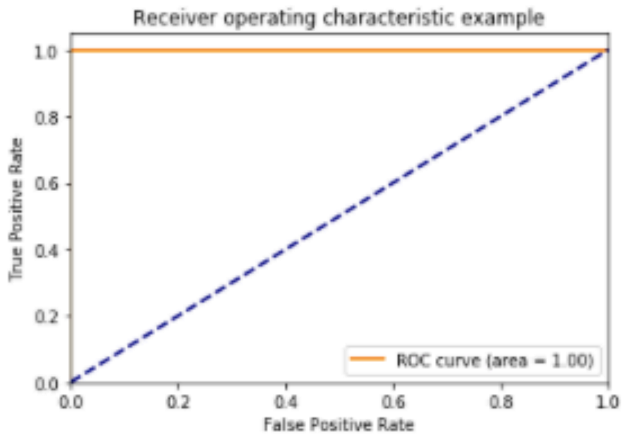
4. One Vs One Classifier (RBFKernel):

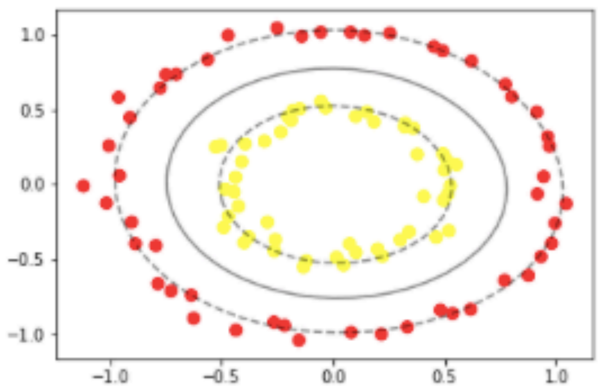
Dataset 1:

```
{'C': 0.01, 'gamma': 1}
For DATASET 1
Predicted    0  1
Correct
0           12  0
1            0  8
```




```
<Figure size 432x288 with 0 Axes>
```

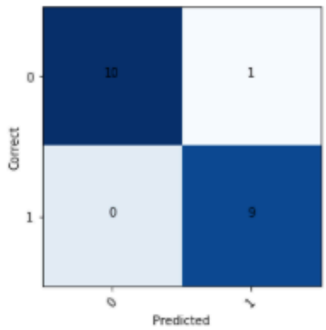




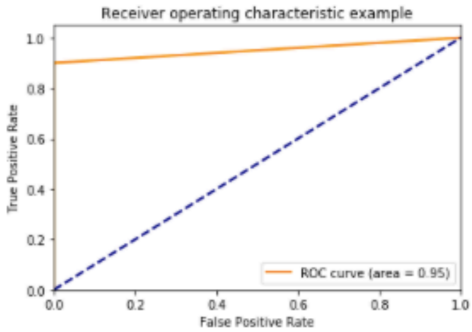

```
Accuracy 1.0
F1-Score 1.0
```

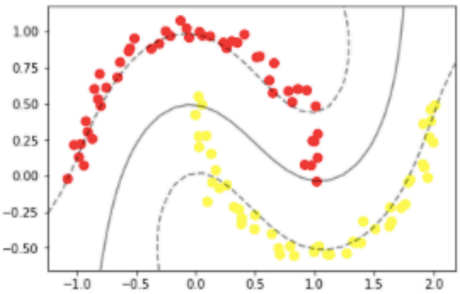
Dataset 2:

```
{'C': 1, 'gamma': 1}

For DATASET 2
Predicted    0  1
Correct
0           10  0
1            1  9
```




```
<Figure size 432x288 with 0 Axes>
```

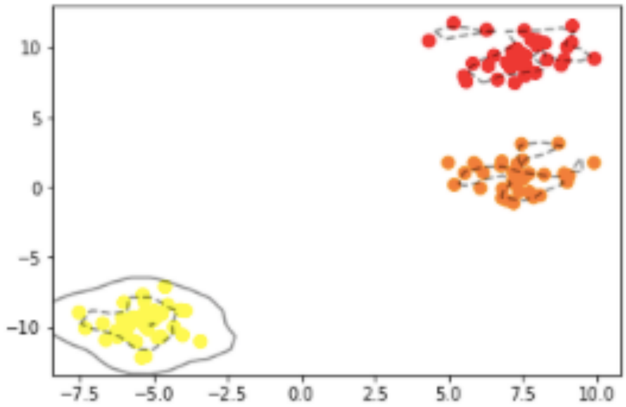




```
Accuracy 0.95
F1-Score 0.949874686716792
```

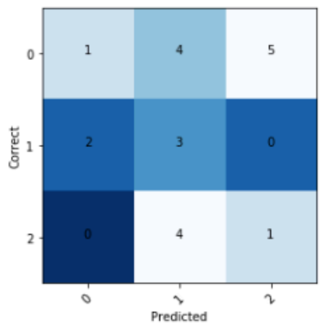
Dataset 3:

```
{'C': 0.1, 'gamma': 0.01}
<Figure size 432x288 with 0 Axes>
```

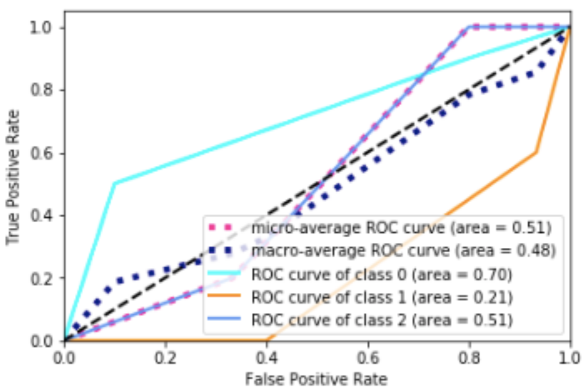



```
Accuracy 1.0
F1-Score 1.0
```

```
Predicted  0  1  2
Correct
0          1  2  0
1          4  3  4
2          5  0  1
```



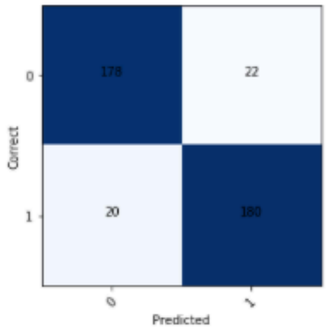

```
<Figure size 432x288 with 0 Axes>

[0.  0.1 0.8 1. ]
[0.         0.4        0.93333333 1.        ]
[0.         0.33333333 0.8        1.        ]
```



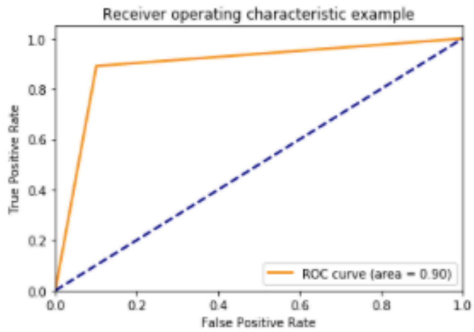

```
Accuracy 0.25
F1-Score 0.23688811188811185
```

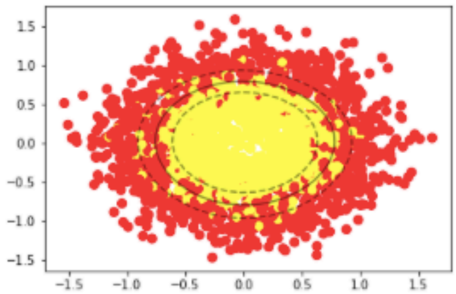
Dataset 4:

```
For DATASET 4
Predicted      0     1
Correct
0            178    20
1             22   180
```




```
<Figure size 432x288 with 0 Axes>
```



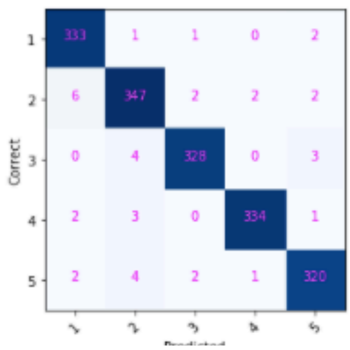



```
Accuracy 0.895
F1-Score 0.8949973749343734
```

5. **For hand-written digits datasets:**

```
Accuracy on Validation Set 0.9776470588235294
F1-Score on Validation Set 0.9777704749753635
Predicted      1     2     3     4     5
Correct
1            333     1     1     0     2
2              6   347     2     2     2
3              0     4   328     0     3
4              2     3     0   334     1
5              2     4     2     1   320
```

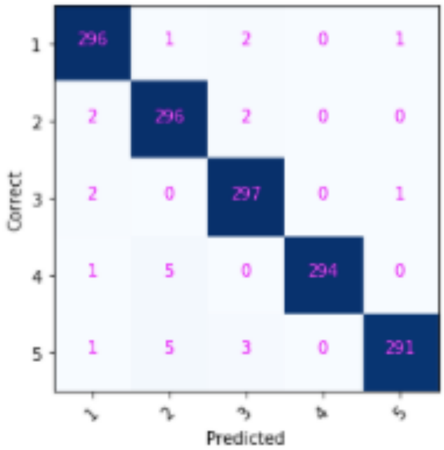
Out[48]: `<Figure size 432x288 with 0 Axes>`



Accuracy on Validation Set

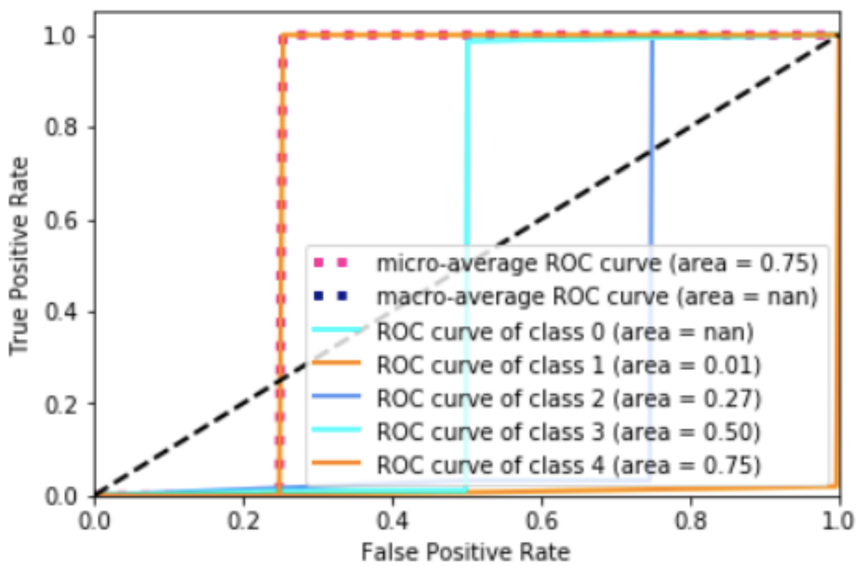
```
Accuracy on Test Set 0.9826666666666667
F1-Score on Test Set 0.9826939917816885
Predicted     1    2    3    4    5
Correct
1           296    1    2    0    1
2             2  296    2    0    0
3             2    0  297    0    1
4             1    5    0  294    0
5             1    5    3    0  291
```

Out[49]: <Figure size 432x288 with 0 Axes>



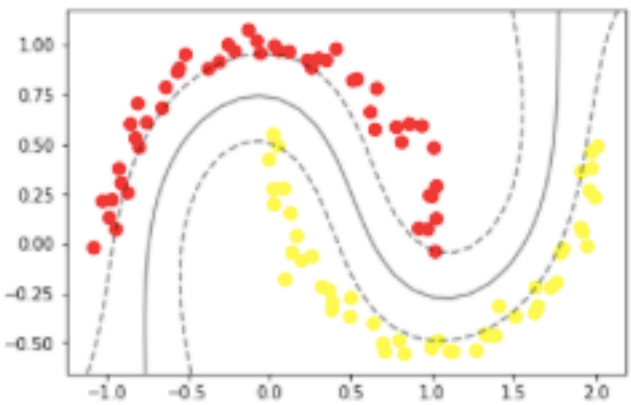
Accuracy on Test Set

ROC curve for the DataSet

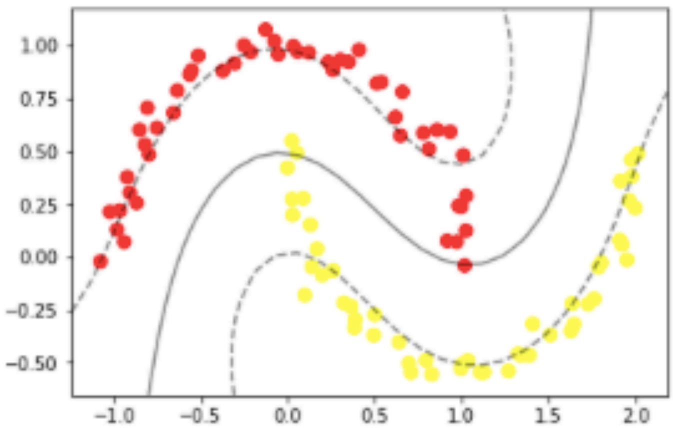


6.
(i) For most of the cases hyper-parameter tuning is not required as accuracy is 100%. For rest I have used GridSearchCV to find best hyper parameter for the model.
(ii) support vectors and margins are plotted above.

(iii) If we take a large value of C then our model overfits. For larger of value of C SVM tries to classify more and more labels correctly, thereby reducing margin width. For reference an example is attached:



SVM with C=100



SVM with C=1

C is a trade-off between training error and the flatness of the solution. The larger C is the less the final training error will be. But if you increase C too much you risk losing the generalization properties of the classifier, because it will try to fit as best as possible all the training points (including the possible errors of your dataset).

(iv) In terms of accuracy, RBF kernel outperforms linear kernel, as it best predicts our dataset. Basically linear kernel is a degenerate version of RBF, hence the linear kernel is never more accurate than a properly tuned RBF kernel. But in terms of speed linear kernel is much faster and solving optimisation problem with linear kernel is much faster and easier.

(v) Confusion matrices for each dataset is plotted above.

(vi) ROC curves for each dataset is plotted above.

(vii) plotted above.

Theory Questions

Homework - 2
Theory Questions

2: The SVM classifier is written as:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x', x) + b$$

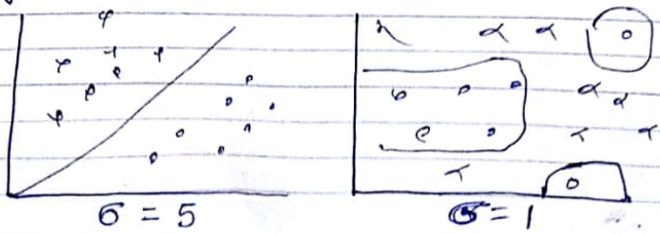
where $K(x', x) \Rightarrow$ kernel function
In case of RBF Kernel

$$K(x', x) = \exp\left(-\frac{||x - x'||^2}{2\sigma^2}\right)$$

Where x' = support vector.

σ = standard deviation which determines the width of guassian distribution

As we can see for the equation of kernel function sigma plays the role of amplifier of the distance between x and x'. if the distance b/w x & x' is much larger than σ then kernel functn tends to zero & if sigma is very small, only the x within the certain distance can effect the predicting point.
Thus a larger sigma tends to make local classifier, larger sigma tends to make a more general classifier.



Page No.
Date : / /

Thus setting a value of kernel hyperparams -ter somewhere between, can help in ~~making~~ avoiding overfitting also in making a classifier that predicts most of the values correctly.

3 A function is called convex if:

$$f(kx_1 + (1-k)x_2) \leq kf(x_1) + (1-k)f(x_2)$$

where $x_1, x_2 \in X$; where X is a convex set

now let $(x, y) \in X_1 + X_2$ where X_1, X_2 are convex
let $x = x_1 + x_2$ $y = y_1 + y_2$ for some $x_1, y_1 \in X_1$ & $x_2, y_2 \in X_2$

Since X_1 & X_2 are convex, we have

$$kx_1 + (1-k)y_1 \in X_1 \ \& \ kx_2 + (1-k)y_2 \in X_2$$

hence $kx_1 + kx_2 + (1-k)y_1 + (1-k)y_2 \in X_1 + X_2$
$k(x_1 + x_2) + (1-k)(y_1 + y_2) \in X_1 + X_2$
Hence $X_1 + X_2$ is also convex

Loss function of L1 regularised linear regression:

$$J(w) = \sum_{i=1}^{m} (y_i - w^T x_i)^2 + \lambda \|w\|$$

Each linear function is parameterized by a weight vector $w \in R^d$. Hence we can define set H, of all parameters, namely $H = R^d$
The set of all examples $Z = X \times Y = R^d \times R = R^{d+1}$
& the loss function is $f(w, (x, y)) = (\langle w, x \rangle - y)^2$

Page No.
Date : / /

Since H is the conven set, & the loss function is conven as we can model $J(w) = (\langle w, x\rangle - y)^2$ as a composition of function $g(a) = a^2$ onto a linear function, hence J is conven.

$\lambda \|w\|$ is also a conven function & we have showed above, that addition of 2 conven function is also conven. Hence ~~of~~ Loss function of Lasso regression is conven.