August - December 2018

Odd Semester

CS669: Pattern Recognition

Programming Assignment 4 : Part - 2

Preethi Srinivasan S18001

1. **Fisher Discriminant Analysis (FDA):**

1.1 Linearly Separable Data

Number of GMM components: 1

FDA is used to project the linearly separable data points in 2 dimensional space onto a 1 dimensional line. On this projected points we built GMM. FDA helps in proper representation of data. Since the projected points are well separated, we get 100% classification accuracy as indicated in Table 1.1.2. For this data set, we got 100% accuracy in assignment 1 also. GMM is built for each of these 3 cases separately. Maximum voting rule is used for final assignment of class label.

Table 1.1 The Confusion Matrix for Linearly Separable data, K = 1

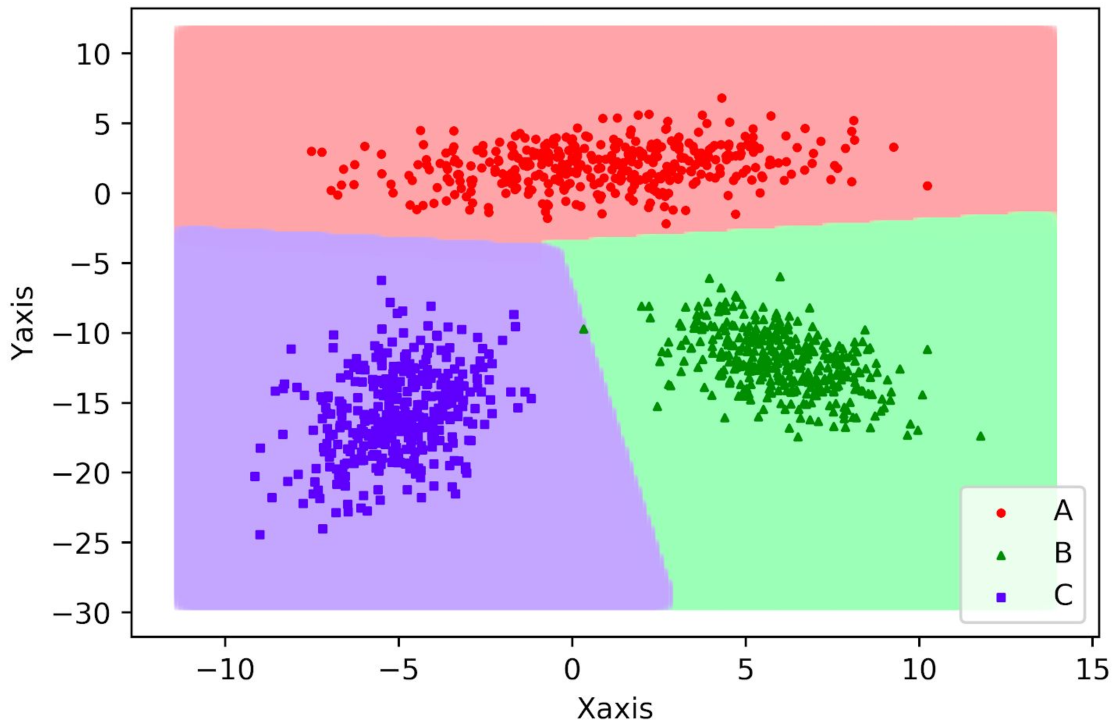
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*125*	*0*	*0*
	Class 2	*0*	*125*	*0*
	Class 3	*0*	*0*	*125*

Table 1.2 The Performance Matrix for Linearly Separable data, K = 1

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*100.0*	*100.0*	*100.0*
Class 2	*100.0*	*100.0*	*100.0*
Class 3	*100.0*	*100.0*	*100.0*
Mean Value	*100.0*	*100.0*	*100.0*

Classification Accuracy: *100.0 %*

Figure 1.1.3 Decision Boundary Plot for K = 1



<u>Number of GMM Components = 2</u>

Table 1.1.4 The Confusion Matrix for Linearly Separable data, K = 2

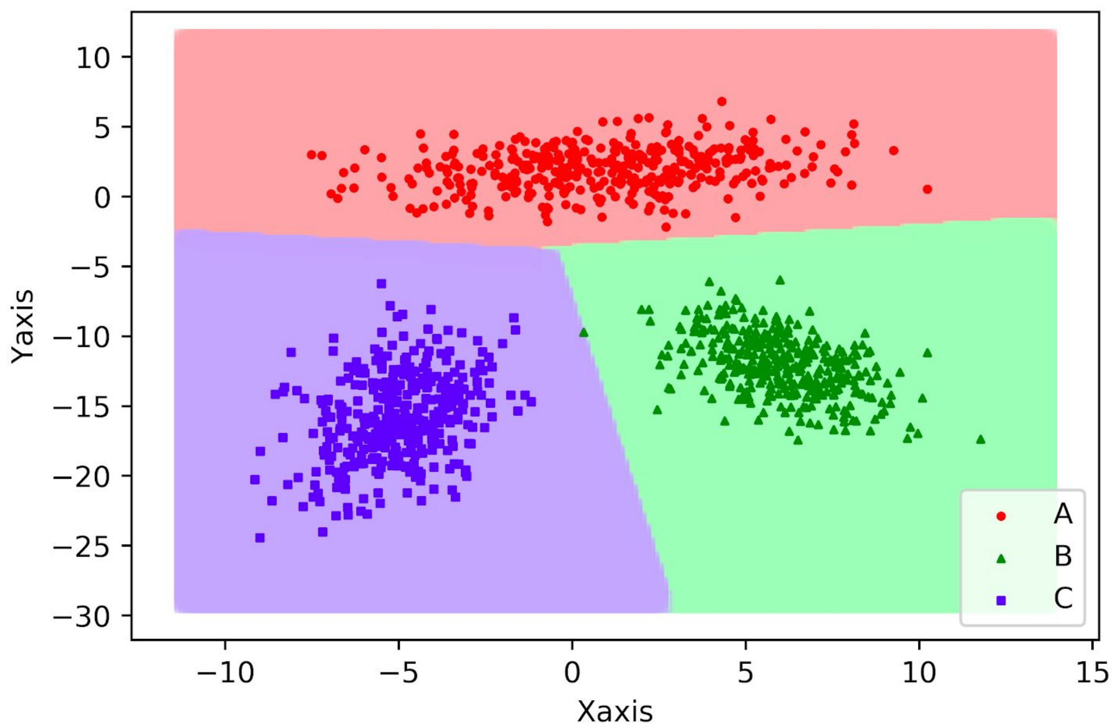
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*125*	*0*	*0*
	Class 2	*0*	*125*	*0*
	Class 3	*0*	*0*	*125*

Table 1.1.5 The Performance Matrix for Linearly Separable data, K = 2

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*100.0*	*100.0*	*100.0*
Class 2	*100.0*	*100.0*	*100.0*
Class 3	*100.0*	*100.0*	*100.0*
Mean Value	*100.0*	*100.0*	*100.0*

Classification Accuracy: *100.0 %*

Figure 1.1.6 Decision Boundary for K=2



<u>Number of GMM Components = 4</u>

Table 1.1.7 The Confusion Matrix for Linearly Separable data, K = 4

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*125*	*0*	*0*
	Class 2	*0*	*125*	*0*
	Class 3	*0*	*0*	*125*

Table 1.1.8 The Performance Matrix for Linearly Separable data, K = 4

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*100.0*	*100.0*	*100.0*
Class 2	*100.0*	*100.0*	*100.0*
Class 3	*100.0*	*100.0*	*100.0*
Mean Value	*100.0*	*100.0*	*100.0*

Classification Accuracy: *100.0 %*

Figure 1.1.9 Decision Boundary for K=4

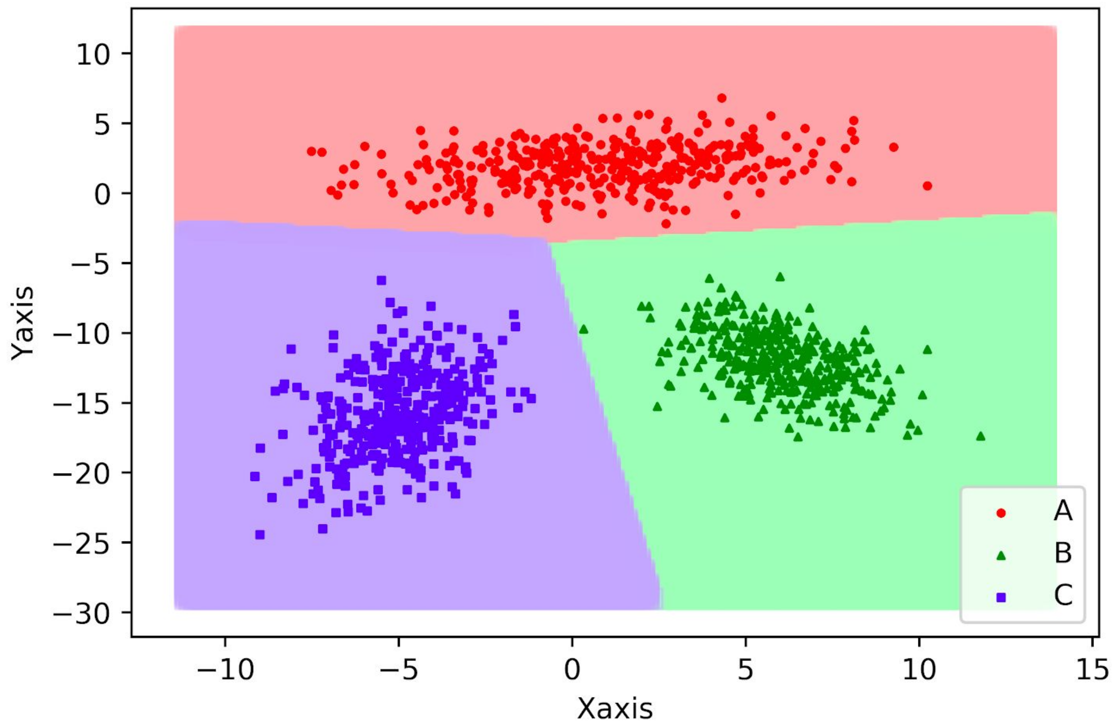


Figure 1.1.10 Linear Separable data of Class 1 and Class 2 is projected on W using FDA.

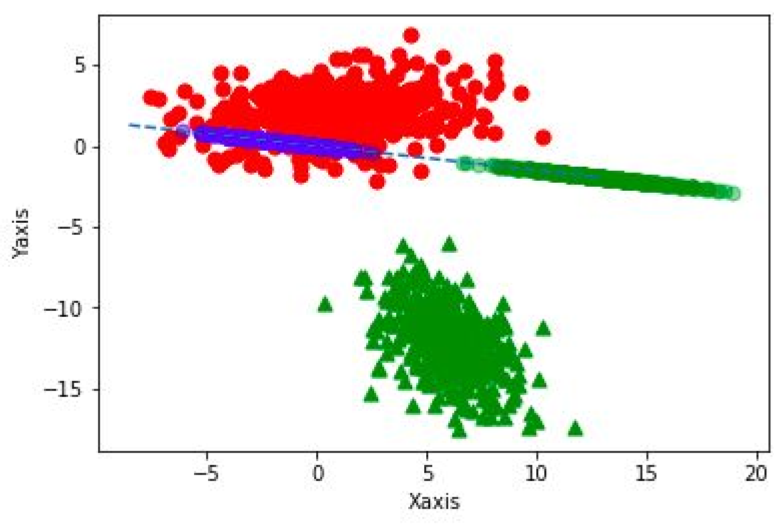
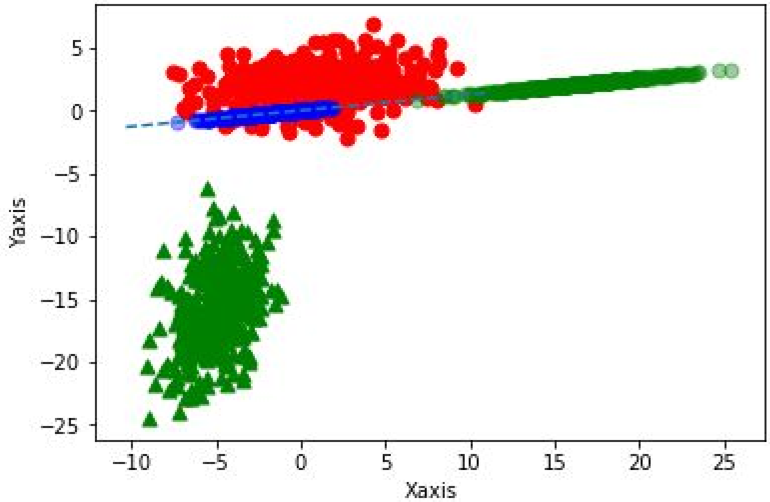


Figure 1.1.11 Linear Separable data of Class 1 and Class 3 is projected on W using FDA.



1.2 Non_linearly Separable Data:

Similar to the first case, FDA to project the points in 2 dimensional space onto 1 dimensional line. Figure 1. shows the projection of class 1 and 2 points on W line (W12 is in the direction with maximum discrimination). We see that the points projected points are overlapping. So, GMM built on this, is not able to give 100% accuracy and we can see that too.

While in Assignment 2, with increase in number of mixtures, accuracy increased, we can see that phenomenon here. This can be observed from the below tables.

Number of GMM Components = 1

Table 1.2.1 The Confusion Matrix for Non-Linearly Separable data, K = 1

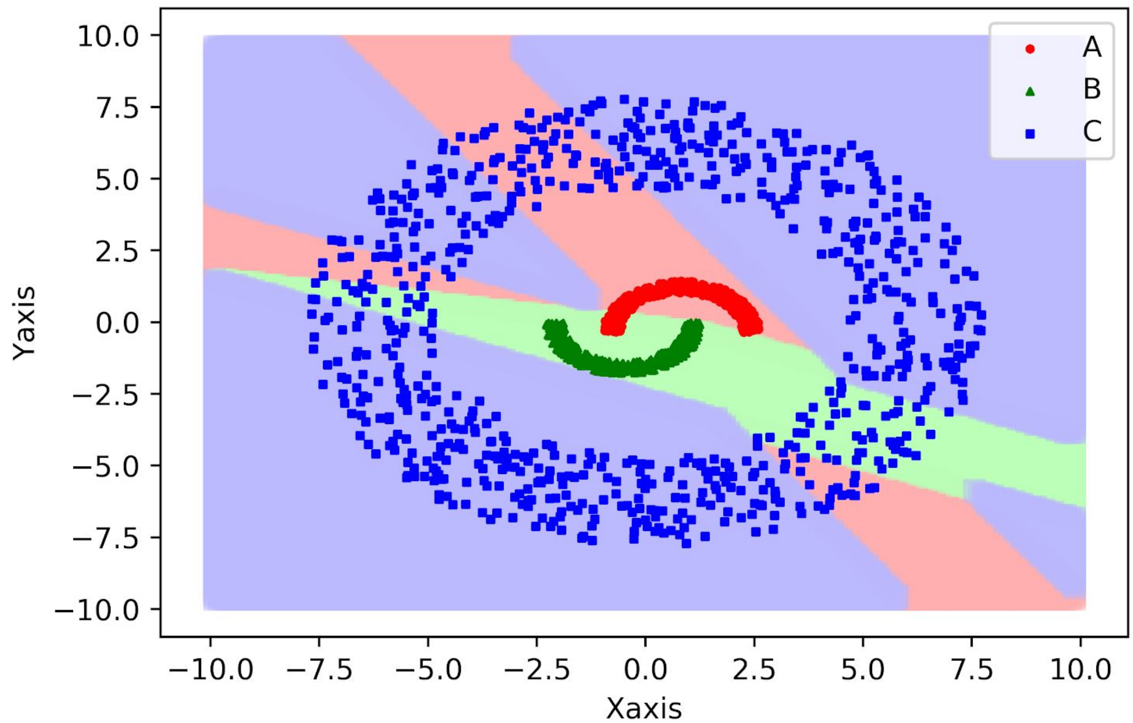
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*120*	*5*	*0*
	Class 2	*4*	*121*	*0*
	Class 3	*48*	*21*	*181*

Table 1.2.2 The Performance Matrix for Non-Linearly Separable data, K = 1

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*69.77*	*96.0*	*80.81*
Class 2	*82.31*	*96.8*	*88.97*
Class 3	*100.0*	*72.4*	*83.99*
Mean Value	*84.03*	*88.4*	*84.59*

class accuracy: 84.399

Figure 1.2.3 Decision Boundary on non linearly separable data for k = 1



<u>Number of GMM Components = 2</u>

Table 1.2.4 The Confusion Matrix for Non-Linearly Separable data, K = 2

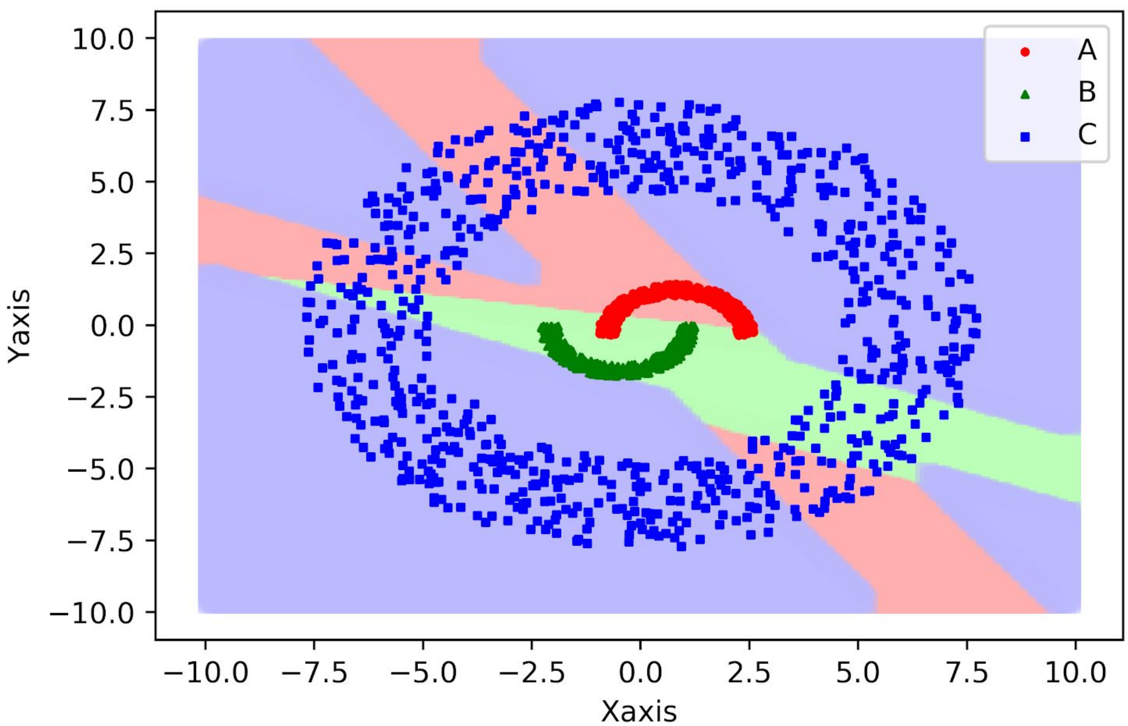
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*121*	*4*	*0*
	Class 2	*6*	*119*	*0*
	Class 3	*54*	*23*	*173*

Table 1.2.5 The Performance Matrix for Non-Linearly Separable data, K = 2

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*66.85*	*96.8*	*79.08*
Class 2	*81.51*	*95.2*	*87.82*
Class 3	*100.0*	*69.2*	*81.8*
Mean Value	*82.79*	*87.07*	*82.9*

class accuracy: 82.6%

Figure 1.2.6 Decision Boundary on non linearly separable data for k = 2



Number of GMM Components = 4

Table 1.2.7 The Confusion Matrix for Non-Linearly Separable data, K = 4

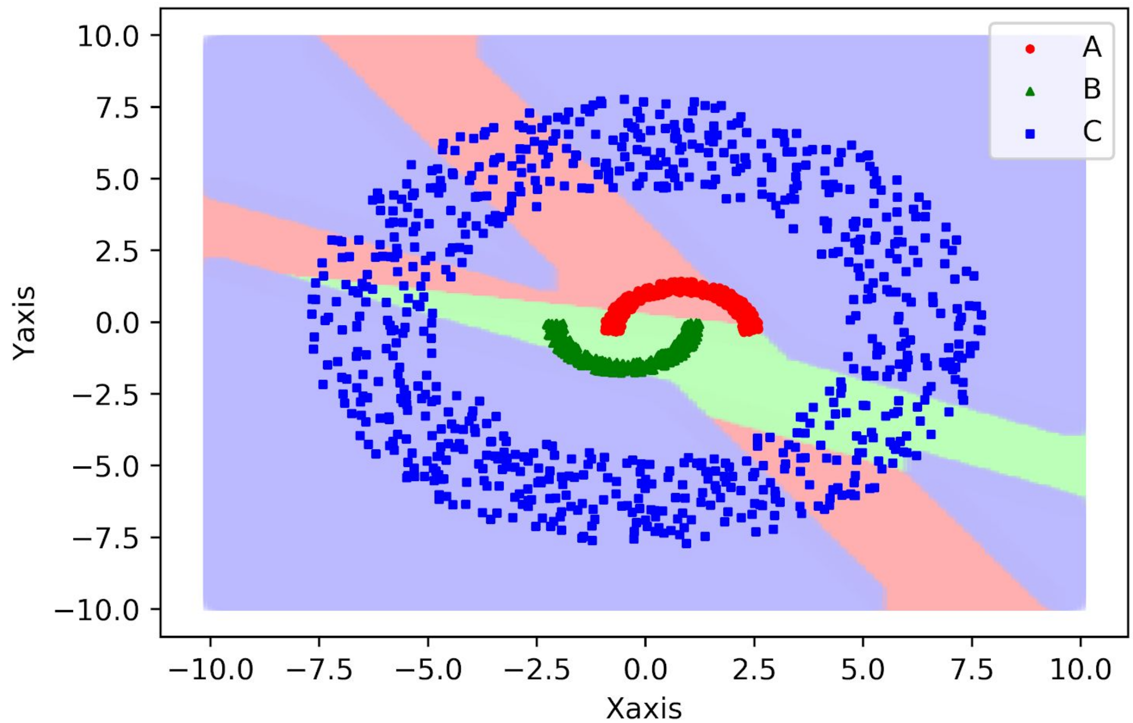
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*119*	*6*	*0*
	Class 2	*4*	*121*	*0*
	Class 3	*53*	*21*	*176*

Table 1.2.8 The Performance Matrix for Non-Linearly Separable data, K = 4

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*67.61*	*95.2*	*79.07*
Class 2	*81.76*	*96.8*	*88.64*
Class 3	*100.0*	*70.4*	*82.63*
Mean Value	*83.12*	*87.47*	*83.45*

class accuracy: 83.2%

Figure 1.2.9 Decision Boundary on non linearly separable data for k = 4



Number of GMM Components = 8

Table 1.2.10 The Confusion Matrix for Linearly Separable data, K = 8

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*117*	*8*	*0*
	Class 2	*3*	*122*	*0*
	Class 3	*53*	*20*	*177*

Table 1.2.11 The Performance Matrix for Linearly Separable data, K = 8

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*67.63*	*93.6*	*78.52*
Class 2	*81.33*	*97.6*	*88.73*
Class 3	*100.0*	*70.8*	*82.9*
Mean Value	*82.99*	*87.33*	*83.38*

class accuracy: 83.2%

Figure 1.1.12 Non Linear Separable data of Class 1 and Class 2 is projected on W using FDA.

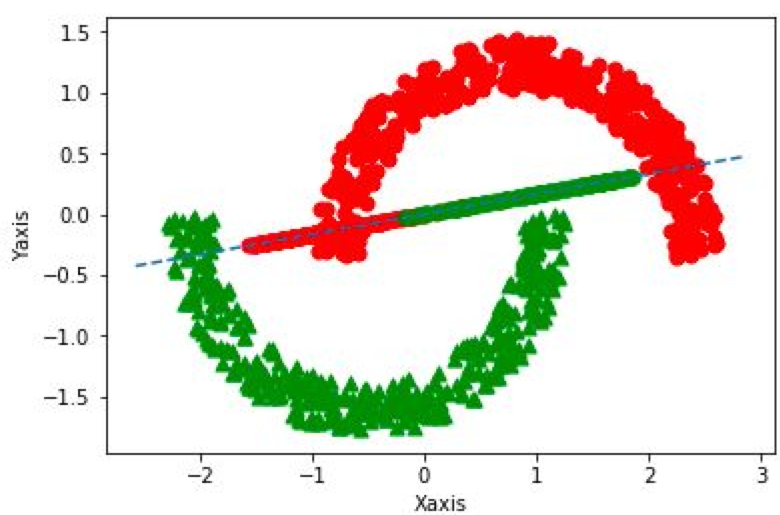
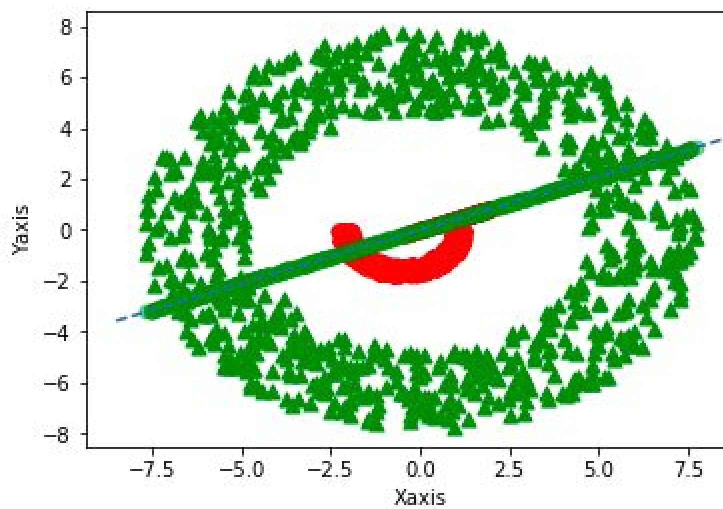


Figure 1.1.13 Non Linear Separable data of Class 1 and Class 3 is projected on W using FDA.



1.3 BoVW data:

FDA is implemented to projected the 32D BoVW features to 1D.

The FDA gives the direction in which the discrimination between the classes is maximum. The results slightly better (49%) than the average value obtained using PCA technique. Because that direction in which maximum separation is observed is used for projection and on that data, GMM is built.

Number of GMM Components = 1

Table 1.3.1 The Confusion Matrix for the classifier after FDA with K = 1

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*23*	*21*	*6*
	Class 2	*24*	*24*	*2*
	Class 3	*21*	*3*	*26*

Table 1.3.2 The Performance Matrix for the classifier after FDA with K = 1

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*33.82*	*46.0*	*38.98*
Class 2	*50.0*	*48.0*	*48.98*
Class 3	*76.47*	*52.0*	*61.9*
Mean Value	*53.43*	*48.67*	*49.96*

Class Accuracy: 48.667%

Number of GMM Components = 2

Table 1.3.3 Confusion Matrix for the classifier after FDA with K = 2

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*30*	*15*	*5*
	Class 2	*26*	*14*	*10*
	Class 3	*15*	*9*	*26*

Table 1.3.4 Performance Matrix for the classifier after FDA with K = 2

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*42.25*	*60.0*	*49.59*
Class 2	*36.84*	*28.0*	*31.82*
Class 3	*63.41*	*52.0*	*57.14*
Mean Value	*47.5*	*46.67*	*46.18*

Class Accuracy: 46.667%

Number of GMM Components = 4

Table 1.3.5 Confusion Matrix for the classifier after FDA with K = 4

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*24*	*18*	*8*
	Class 2	*17*	*22*	*11*
	Class 3	*19*	*3*	*28*

Table 1.3.6 Performance Matrix for the classifier after FDA with K = 4

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*40.0*	*48.0*	*43.64*
Class 2	*51.16*	*44.0*	*47.31*
Class 3	*59.57*	*56.0*	*57.73*
Mean Value	*50.25*	*49.33*	*49.56*

Class Accuracy: 49.333%

2. Perceptron Algorithm

Simple perceptron algorithm classifies a pair of linear separable classes. We have 3 classes. So we built classifiers for every pair of classes. Every test point is tested for every pair of classes and Class label is allocated based on final voting scheme.

Figure shows decision region obtained for three linearly separable classes. We got 99.7% accuracy. The one misclassified point is also due to the fact that perceptron does not give the weight vectors of optimal separating hyperplane. If we know, one hyperplane can be drawn to separate the two classes, then Infinite hyperplanes can be drawn separating the two classes. But, perceptron does not come up with the optimal separating hyperplane. So during test, some points can be misclassified.

However, we observed that the perceptron algorithm converges and weight vector is obtained in finite number of iterations as shown below with a **learning rate of 0.1**:

Between Class 0 and Class 1:
Number of Iterations = 280
Weight Vector = [-331.1, -22.36642901, -176.5313441]

Between Class 0 and Class 2:
Number of Iterations = 356
Weight Vector = [-419.5, -28.19497601, -223.7793425]

Between Class 1 and Class 2:
Number of Iterations = 66
Weight Vector = [-10.6, -431.782445, -15.02704]

Table 2.1 Confusion Matrix for the classifier

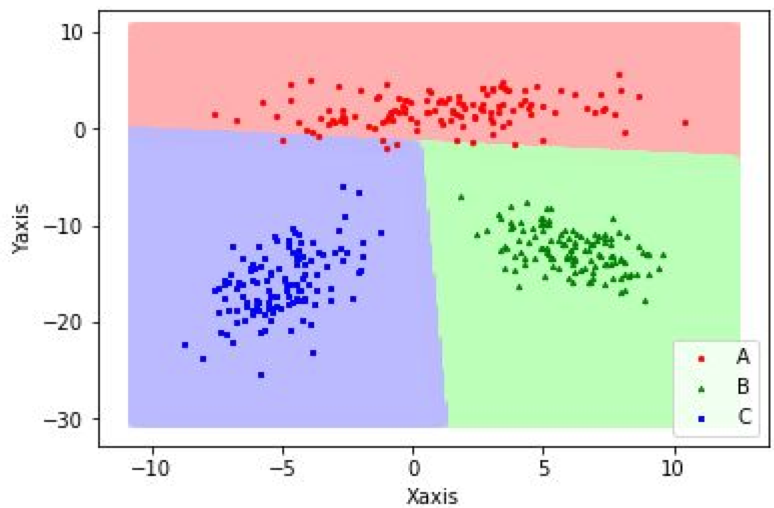
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*124*	*0*	*1*
	Class 2	*0*	*125*	*0*
	Class 3	*0*	*0*	*125*

Table 2.2 Performance Matrix for the classifier

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*100.0*	*99.2*	*99.6*
Class 2	*100.0*	*100.0*	*100.0*
Class 3	*99.21*	*100.0*	*99.6*
Mean Value	*99.74*	*99.73*	*99.73*

class accuracy: 99.73%

Figure 2.3 Decision Boundary of LS data using Perceptron technique



3. SUPPORT VECTOR MACHINE

SVM is supervised machine learning algorithm, can be used for both classification and regression problems. SVM performs some kernel trick techniques, which is transforming the lower dimension data to higher dimension and then it finds the separating hyperplane between the data. We used Scikit Learn library to perform the experiments.

We have used following kernel methods in the experiments:

1. Linear Kernel:

$$K(X_m, X_n) = X_m^T X_n$$

2. Polynomial Kernel:

$$K(X_m, X_n) = (aX_m^T X_n + b)^p$$

3. Gaussian Kernel or Radial Basis Function:

$$K(X_m, X_n) = \exp(-\|X_m - X_n\|^2 / \sigma)$$

4. Sigmoid Kernel:

$$K(X_m, X_n) = \tanh(\gamma X_i^T X_j + r)$$

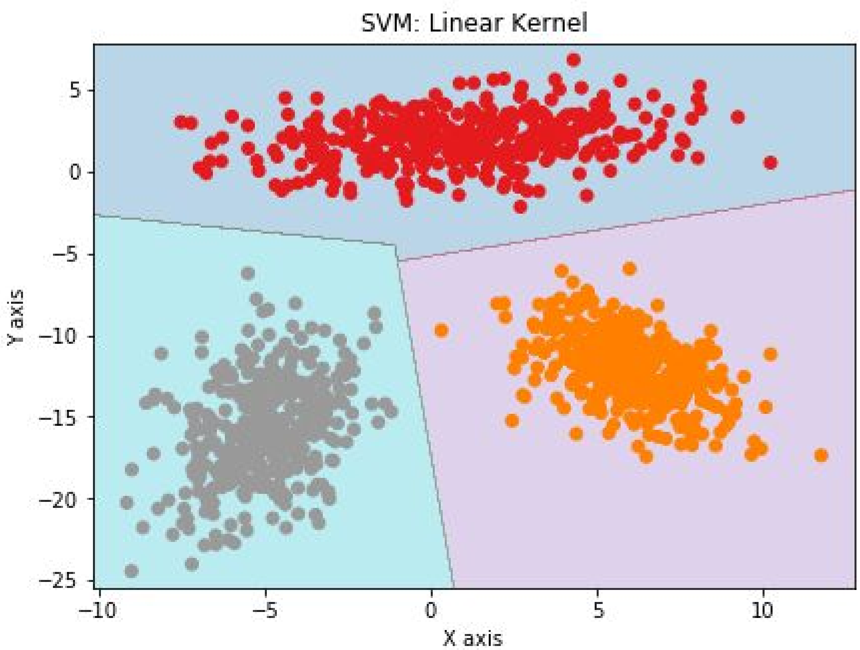
3.1 Linearly Separable data:

SVM is applied with 4 different kernels on Linearly Separable data. Parameters are adjusted to get better results. Support vectors are shown and corresponding hyperplane between every pair of classes are also shown. When linear kernel was used on linearly separable data, we got 100% accuracy with C = 1. With RBF and Polynomial kernel also, in all sub cases, we got 100% accuracy as we can see below from the corresponding decision boundary figures. Decision boundary is highly non linear. With lower C, boundary gets smoother.

Sigmoid kernel gives very low accuracy comparatively.

3.1.1 Linear Kernel

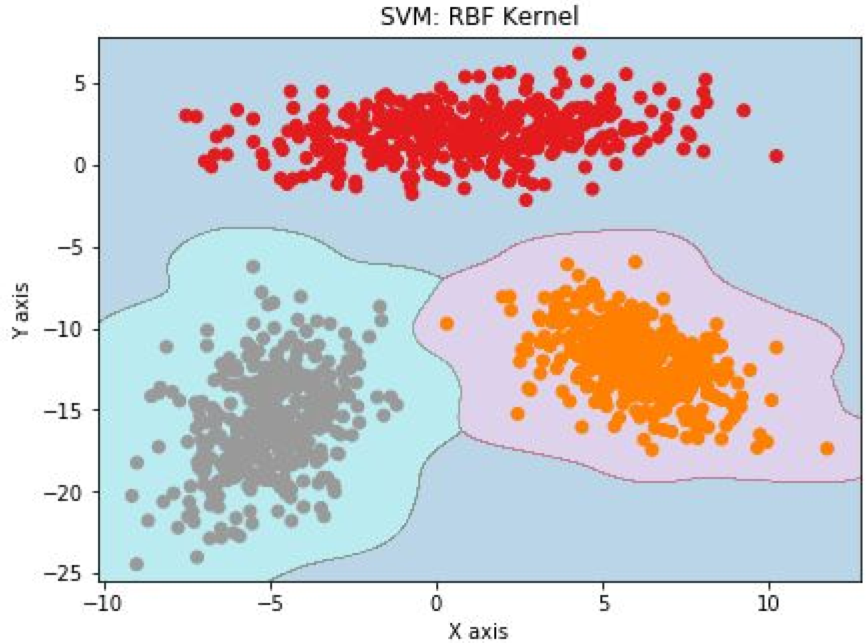
Figure 3.1.1.1 C = 1.0 Linear Kernel on Linearly Separable data



All 100 % accuracy

3.1.2 RBF Kernel

Figure 3.1.2.1 Gamma = 0.5, c = 1.0 on Linearly Separable data



Accuracy 100%

Figure 3.1.2.2 Gamma = 0.5, c = 10.0 on Linearly Separable data

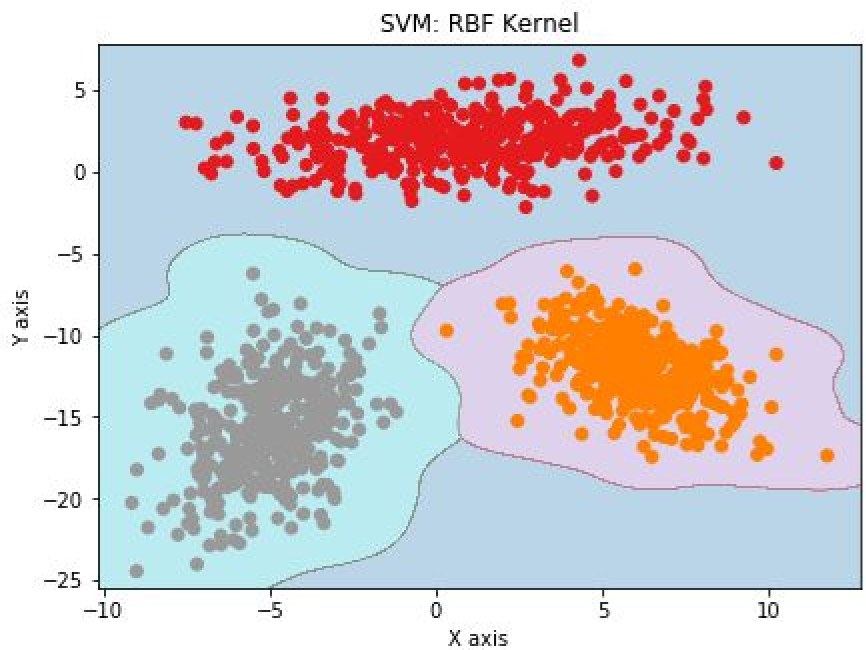
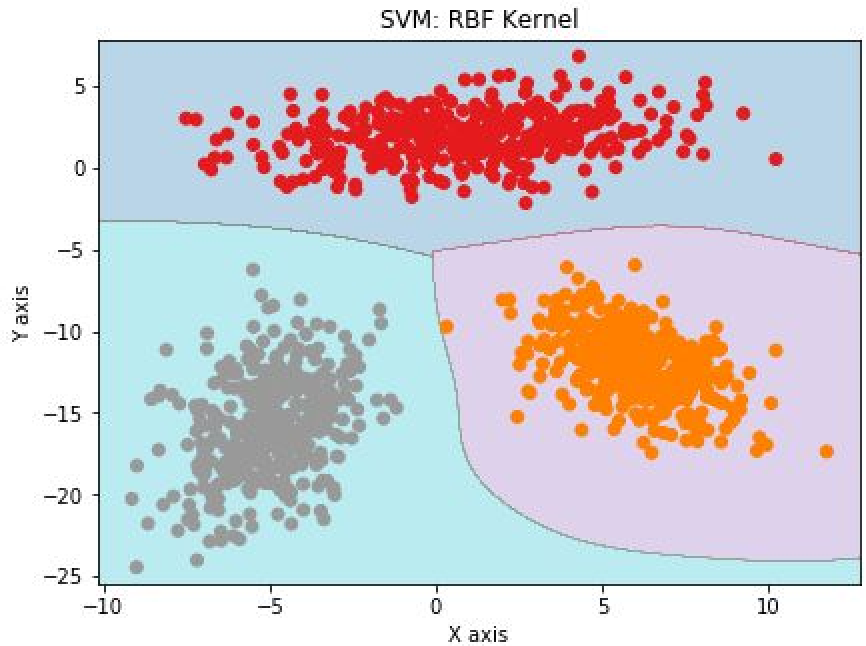


Figure 3.1.2.3 Gamma = 0.05, c = 1.0 on Linearly Separable data



3.1.3 Polynomial Kernel

C = 1.0, gamma = 0.5, degree = 4 on LS Data

Table 3.1.3.1 Confusion Matrix

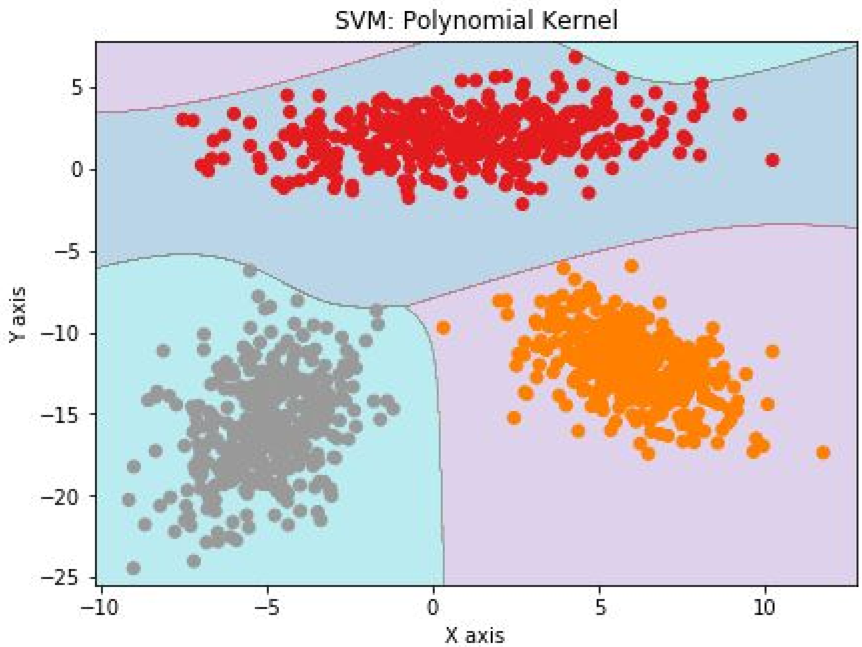
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*124*	*0*	*1*
	Class 2	*0*	*125*	*0*
	Class 3	*2*	*0*	*123*

Table 3.1.3.2 Performance Matrix

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*98*	*99*	*99*
Class 2	*100*	*100*	*100*
Class 3	*33.3*	*100.0*	*50.0*
Mean Value	*99*	*98*	*99*

Class Accuracy: 99.7%

Figure 3.1.3.3 Decision Boundary



<u>*C = 1, gamma = 0.5, degree = 8*</u>

Table 3.1.3.4 Confusion Matrix

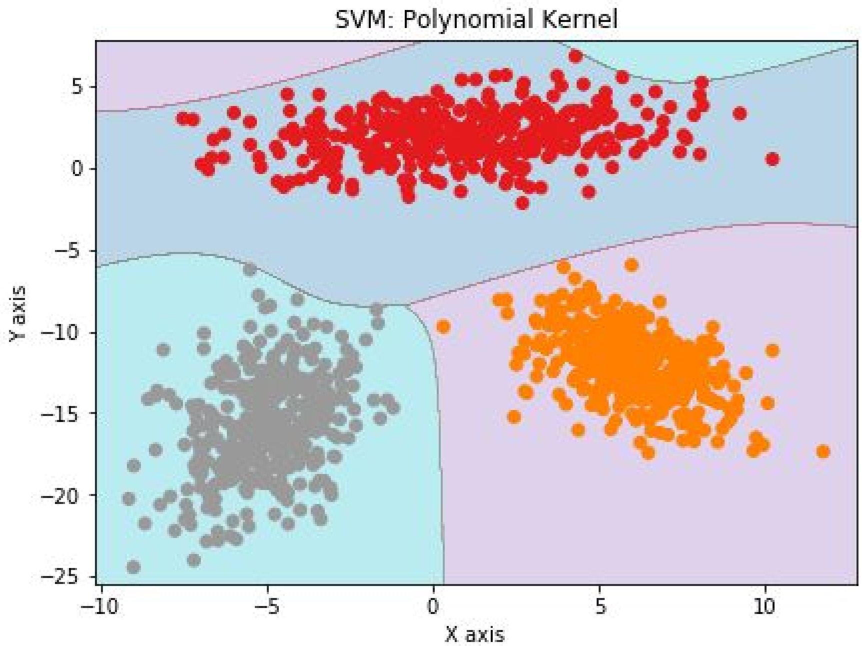
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*124*	*0*	*1*
	Class 2	*0*	*125*	*0*
	Class 3	*2*	*0*	*123*

Table 3.1.3.5 Performance Matrix

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*98.41*	*99.2*	*98.8*
Class 2	*100.0*	*100.0*	*100.0*
Class 3	*99.19*	*98.4*	*98.8*
Mean Value	*99.2*	*99.2*	*99.2*

Class Accuracy: 99.7%

Figure 3.1.3.6 Decision Boundary



3.1.4 Sigmoid kernel

<u>*C =1, gamma = 0.5*</u>

3.1.4.1 Confusion Matrix

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*84*	*25*	*16*
	Class 2	*0*	*125*	*0*
	Class 3	*0*	*125*	*0*

Table 3.1.4.2 Performance Matrix

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*100*	*67.2*	*80.38*
Class 2	*45.45*	*100.0*	*62.5*
Class 3	*0*	*0*	*0*
Mean Value	*48.48*	***55.73***	*47.63*

Class Accuracy: 55.7%

Figure 3.1.4.3 Decision Boundary

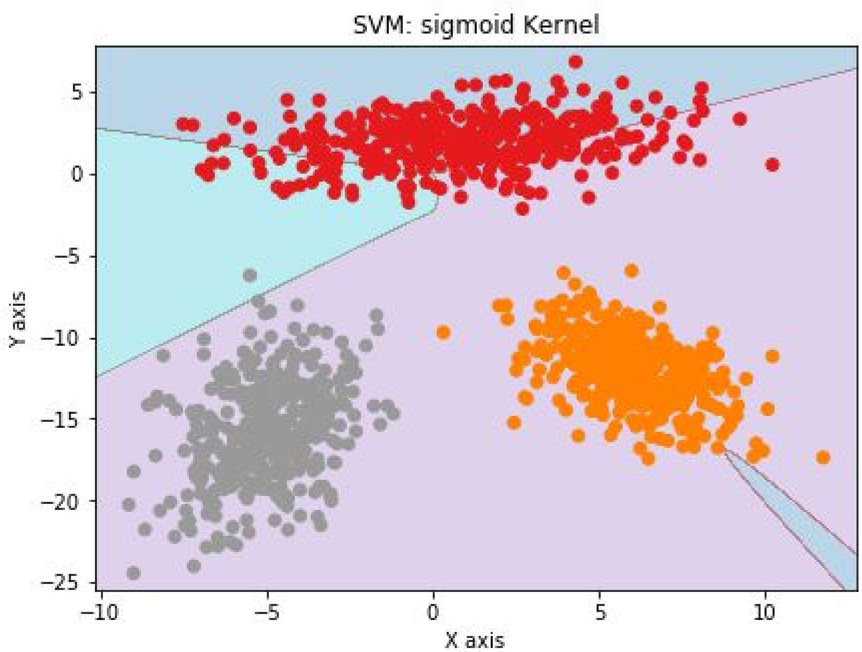
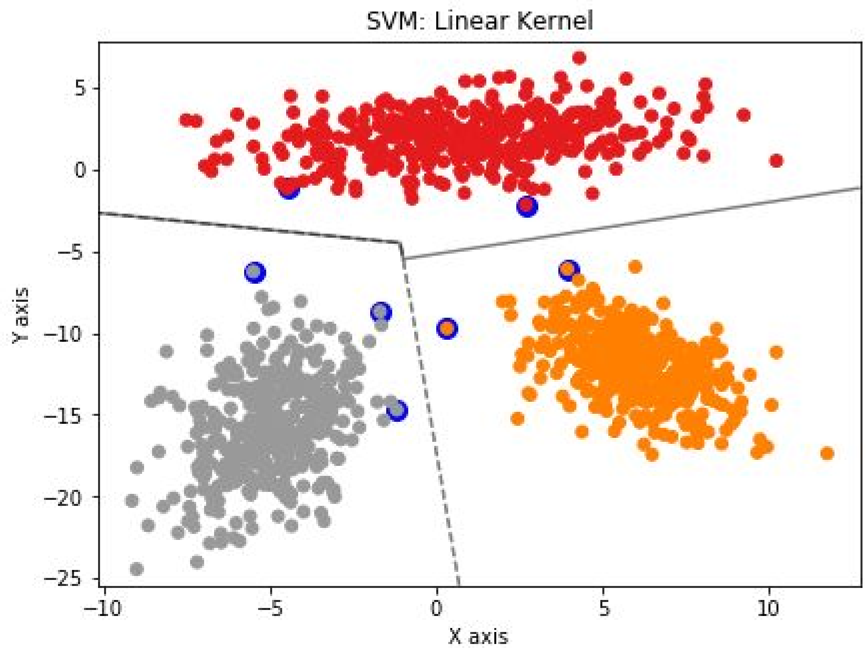


Figure 3.1.5 Linear Kernel with support vectors.



3.2 Non-Linear Separable data

With linear kernel we got 33.33% accuracy. Using a linear kernel on non linearly separable data is highly inefficient. All data points got classified as Class 1.

While RBF kernel gives 100% accuracy. We observe that, the decision boundary is highly non linearly in nature and covers all the points.

With Polynomial kernel, accuracy went down to 85%. Sigmoidal performs even worser in non linearly separable case.

Number of support vectors can be changed by changing the value of C.

3.2.1 Linear Lernel:

Table 3.2.1.1 Confusion Matrix

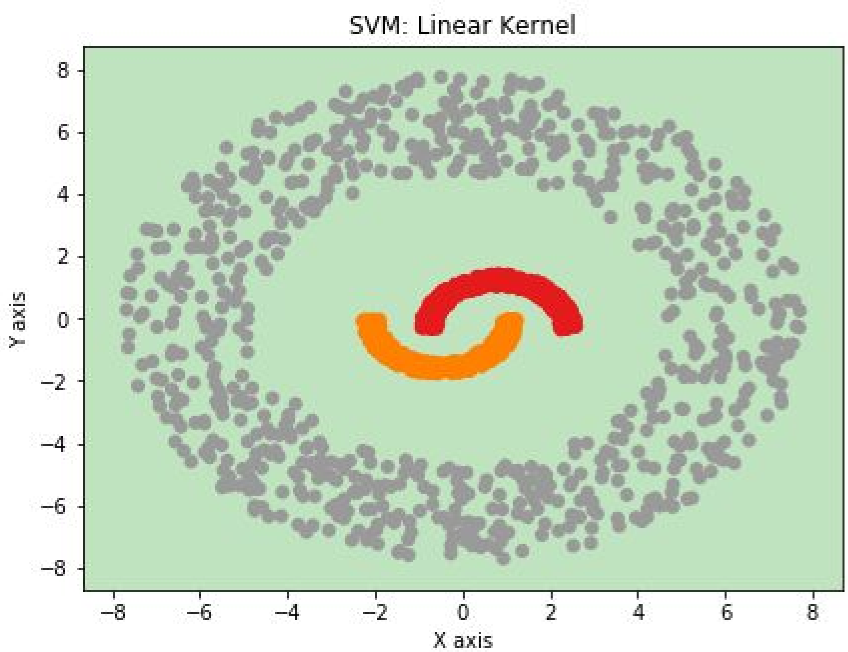
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*0*	*0*	*125*
	Class 2	*0*	*0*	*125*
	Class 3	*0*	*0*	*250*

Table 3.2.1.2 Performance Matrix

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*0*	*0*	*0*
Class 2	*0*	*0*	*0*
Class 3	*50.0*	*100.0*	*66.67*
Mean Value	*16.67*	*33.33*	*22.22*

class accuracy: 33.33%

Figure 3.2.1.3 Decision Boundary



3.2.2 RBF Kernel

Table 3.2.2.1 Confusion Matrix for the classifier

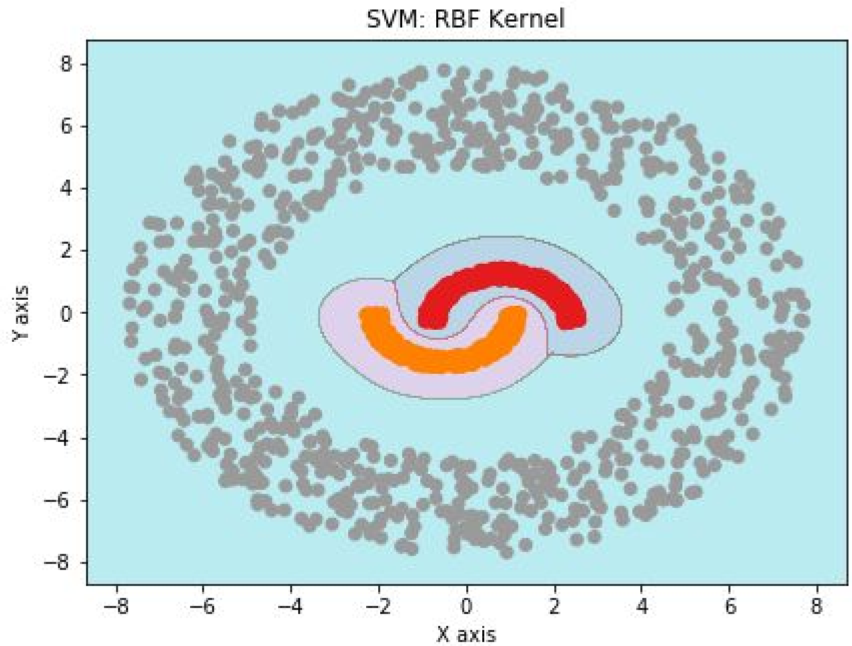
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*125*	*0*	*0*
	Class 2	*0*	*125*	*0*
	Class 3	*0*	*0*	*250*

Table 3.2.2.2 Performance Matrix for the classifier

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*100.0*	*100.0*	*100.0*
Class 2	*100.0*	*100.0*	*100.0*
Class 3	*100.0*	*100.0*	*100.0*
Mean Value	*100.0*	*100.0*	*100.0*

class accuracy: 100%

Figure 3.2.2.3 Decision Boundary



3.2.3 Polynomial Kernel

Table 3.2.3.1 Confusion Matrix

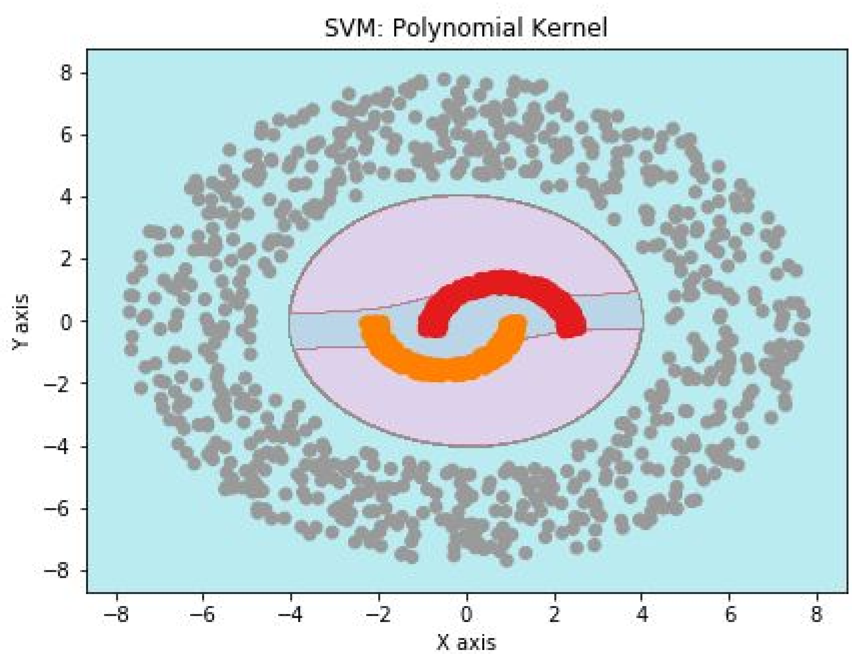
	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*107*	*18*	*0*
	Class 2	*54*	*71*	*0*
	Class 3	*0*	*0*	*250*

Table 3.2.3.2 Performance Matrix

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*66.46*	*85.6*	*74.83*
Class 2	*79.78*	*56.8*	*66.63*
Class 3	*100.0*	*100.0*	*100.0*
Mean Value	*82.08*	*80.8*	*80.39*

class accuracy: 85.6%

Figure 3.2.3.3 Decision Boundary



3.2.4 Sigmoidal Kernel

Table 3.2.4.1 Confusion Matrix

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*47*	*5*	*73*
	Class 2	*30*	*33*	*62*
	Class 3	*123*	*56*	*71*

Table 3.2.4.2 Performance Matrix

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*23.5*	*37.6*	*28.92*
Class 2	*35.11*	*26.4*	*30.14*
Class 3	*34.47*	*28.4*	*31.14*
Mean Value	*31.02*	*30.8*	*30.07*

class accuracy: 30.2%

Figure 3.2.4.3 Decision Boundary

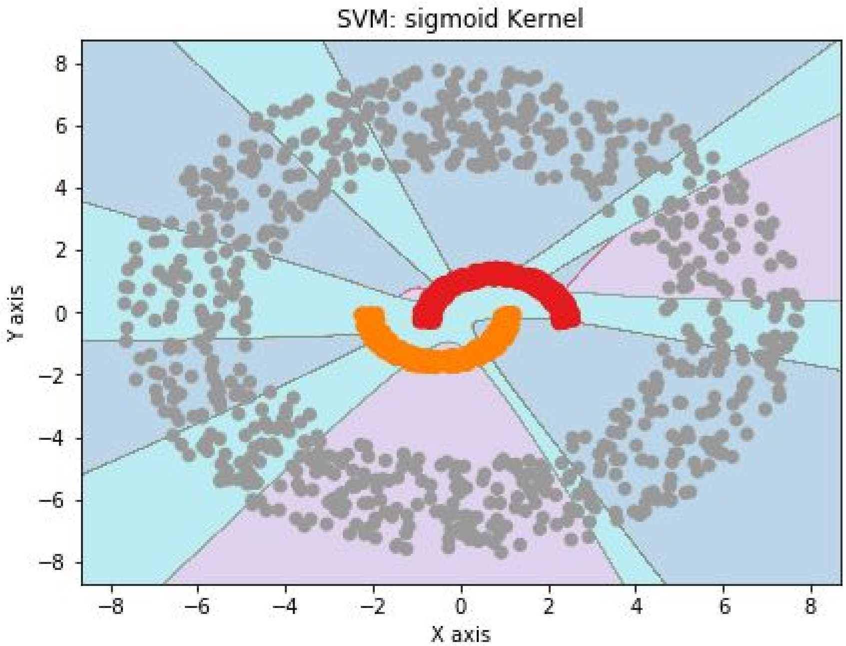
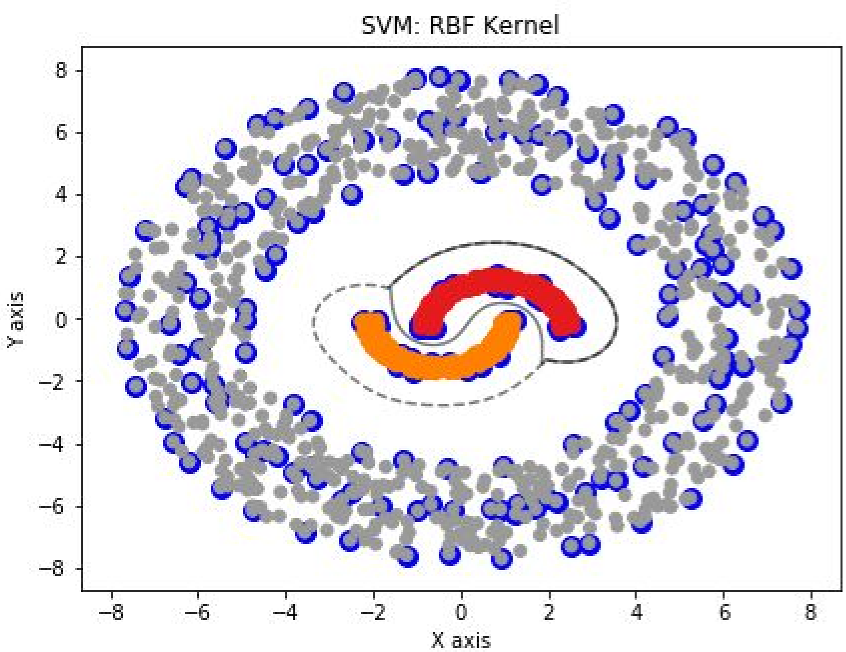


Figure 3.2.5 RBF kernel with support vectors..



Class accuracy 100%

3.3 Bovw

Here we applied SVM on 32D BoVW features for scene image data set. With linear kernel we got accuracy = 51.3% using C = 1 and accuracy = 49% using C = 10. Not much differed here.

With RBF, the accuracy is lower compared to PCA and FDA. This may be because of the overlapping nature of the projected data. This might mean, that 32 dimensional BoVW data is not able to differentiate between the images of 3 classes.

Compared to techniques employed in previous assignments, accuracy obtained using polynomial kernel in BoVW's case is highest so far (61.3%). This is due to the ability of SVM to capture even the sparse data or variability in the data.

3.3.1 Linear kernel

C = 1.0

Table 3.3.1.1 Confusion Matrix

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*30*	*13*	*7*
	Class 2	*12*	*23*	*15*
	Class 3	*24*	*2*	*24*

Table 3.3.1.2 Performance Matrix

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*45.45*	*60.0*	*51.72*
Class 2	*60.53*	*46.0*	*52.27*
Class 3	*52.17*	*48.0*	*50.0*
Mean Value	*52.72*	*51.33*	*51.33*

class accuracy: 51.33%

C = 10

Table 3.3.1.3 Confusion Matrix

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*30*	*13*	*7*
	Class 2	*12*	*22*	*16*
	Class 3	*26*	*2*	*22*

Table 3.3.1.4 Performance Matrix for the classifier

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*44.12*	*60.0*	*50.85*
Class 2	*59.46*	*44.0*	*50.57*
Class 3	*48.89*	*44.0*	*46.32*
Mean Value	*50.82*	*49.33*	*49.25*

class accuracy: 49.33%

3.3.2 RBF

C = 1 , gamma = 0.5

Table 3.3.2.1 Confusion Matrix for the classifier

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*4*	*0*	*46*
	Class 2	*1*	*1*	*48*
	Class 3	*0*	*0*	*50*

Table 3.3.2.2 Performance Matrix for the classifier

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*80.0*	*8.0*	*14.55*
Class 2	*100.0*	*2.0*	*3.92*
Class 3	*34.72*	*100.0*	*51.55*
Mean Value	*71.57*	*36.67*	*23.34*

class accuracy: 36.66%

C = 10, gamma = 0.5

Table 3.3.2.3 Confusion Matrix for the classifier

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*1*	*1*	*48*
	Class 2	*0*	*2*	*48*
	Class 3	*0*	*0*	*50*

Table 3.3.2.4 Performance Matrix for the classifier

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*100.0*	*2.0*	*3.92*
Class 2	*66.67*	*4.0*	*7.55*
Class 3	*34.25*	*100.0*	*51.02*
Mean Value	*66.97*	*35.33*	*20.83*

class accuracy: 35.33%

3.3.3 Polynomial kernel

Table 3.3.3.1 Confusion Matrix for the classifier

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*33*	*13*	*4*
	Class 2	*23*	*20*	*7*
	Class 3	*9*	*2*	*39*

Table 3.3.3.2 Performance Matrix for the classifier

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*50.77*	*66.0*	*57.39*
Class 2	*57.14*	*40.0*	*47.06*
Class 3	*78.0*	*78.0*	*78.0*
Mean Value	*61.97*	*61.33*	*60.82*

class accuracy: 61.33%

3.3.4 Sigmoid kernel

Table 3.3.4.1 Confusion Matrix for the classifier

	Class assigned by the Classifier			
Actual Values		*Class 1*	*Class 2*	*Class 3*
	Class 1	*0*	*0*	*50*
	Class 2	*2*	*3*	*45*
	Class 3	*0*	*0*	*50*

Table 3.3.4.2 Performance Matrix for the classifier

	Precision (%)	*Recall Rate (%)*	*F Score (%)*
Class 1	*0*	*0*	*0*
Class 2	*100*	*6*	*11.32*
Class 3	*34.48*	*100.0*	*51.28*
Mean Value	*44.83*	*35.33*	*20.87*

class accuracy: 35.33%

4. Conclusion

In the first part of the assignment, we implemented PCA to reduce the dimensions of 32D BoVW to different lower dimensional space, built GMM and performed classification on it. We got 51.3% accuracy with 12 principal components.

Next we performed FDA and we got slightly better results on BoVW data. If the data is not overlapping or concentric (one inside the other), FDA is better than PCA because it finds out better representation by finding the direction of higher separation. However, FDA gives only one direction in which separation is maximum. So, clearly we could see that FDA didn't work properly on non linearly separable data because projected data in the chosen direction were overlapping. Using GMM, we got 100% accuracy on the same dataset.

We also implemented Perceptron algorithm on linearly separable data and got 99.7% accuracy. Although the data is linearly separable, perceptron does not give 100% accuracy because it does not choose optimal separating hyperplane between the classes.

In order to get an optimal separating hyperplane, SVM is used. SVM on nonlinearly separable data gives 100% accuracy. But on BoVW, even SVM with RBF kernel could not provide good accuracy because features were overlapping. Even in the higher dimensions, features are overlapping and not well separated.