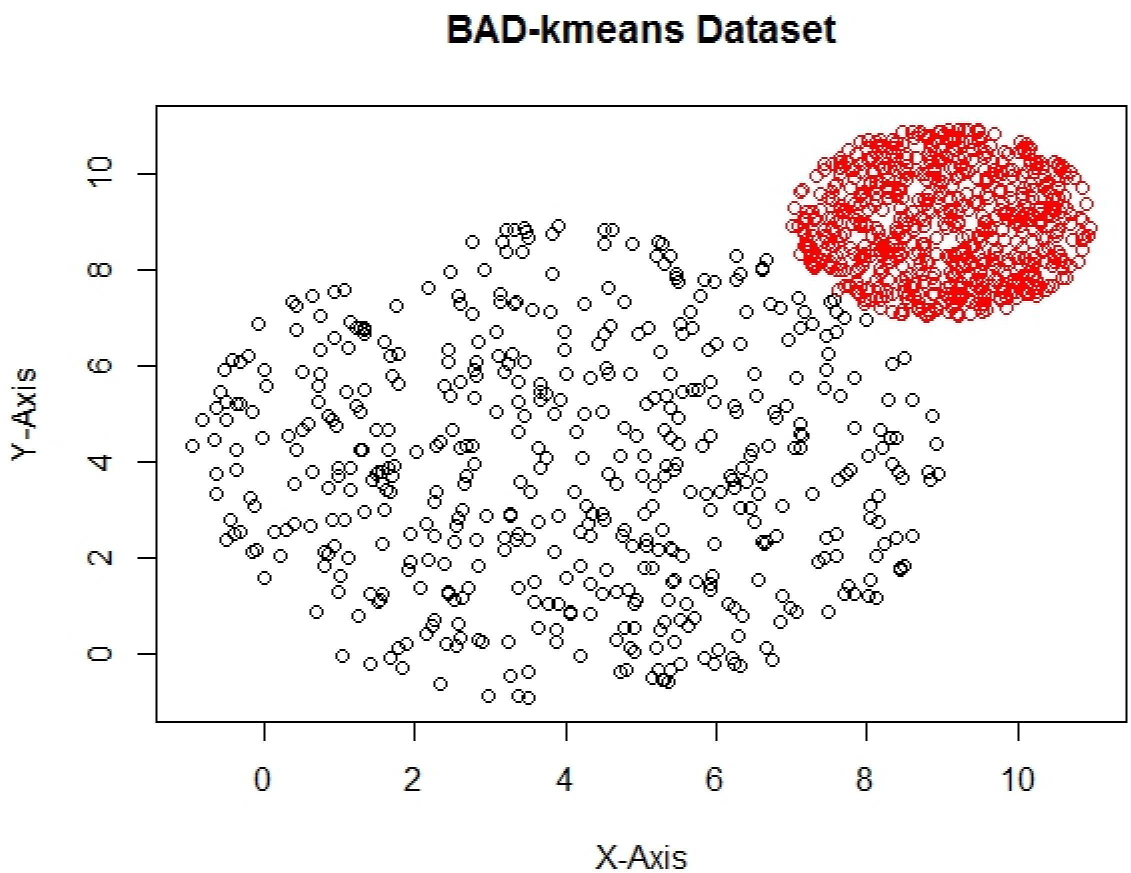
Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

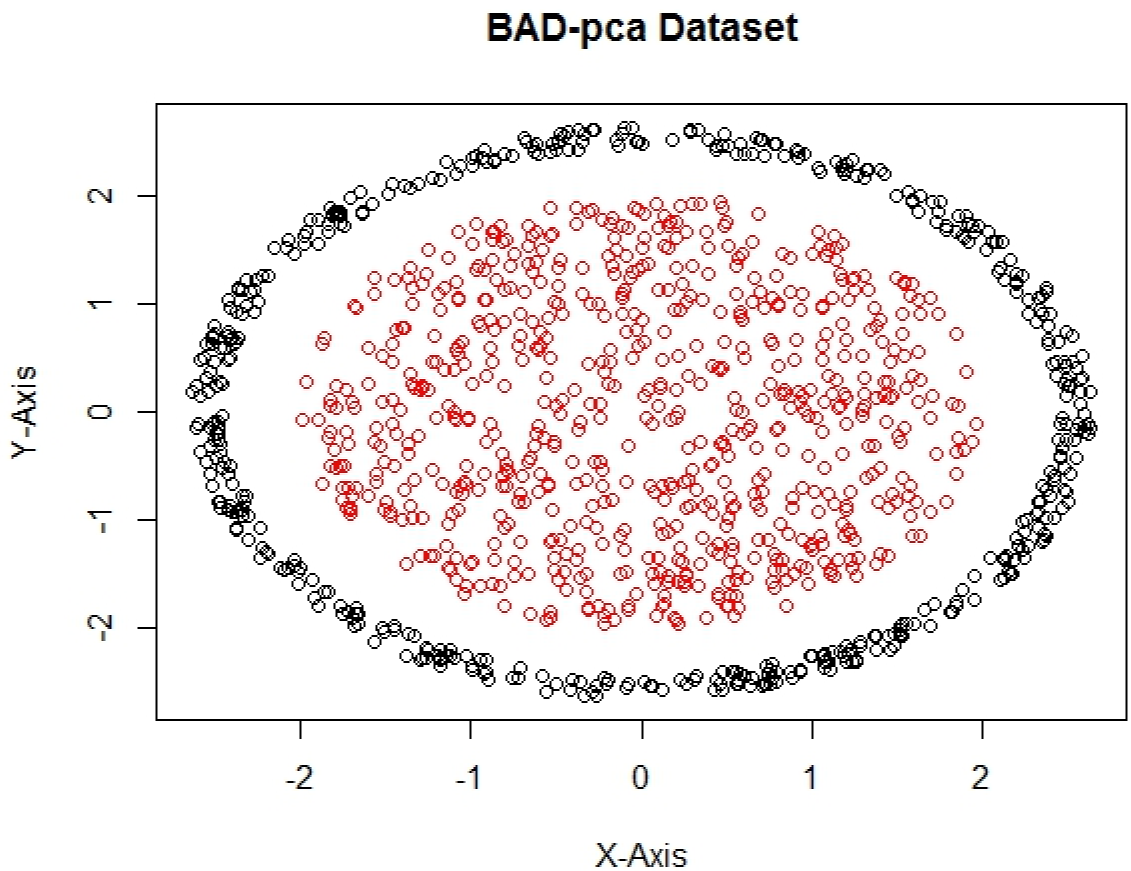
Exercise 1: Generating the data sets. Write a script (in R, Matlab, or SAS) that generates three data sets in a 2-dimensional space, defined as follows :

(a) BAD_kmeans: The data set for which the k means clustering algorithm will not perform well.



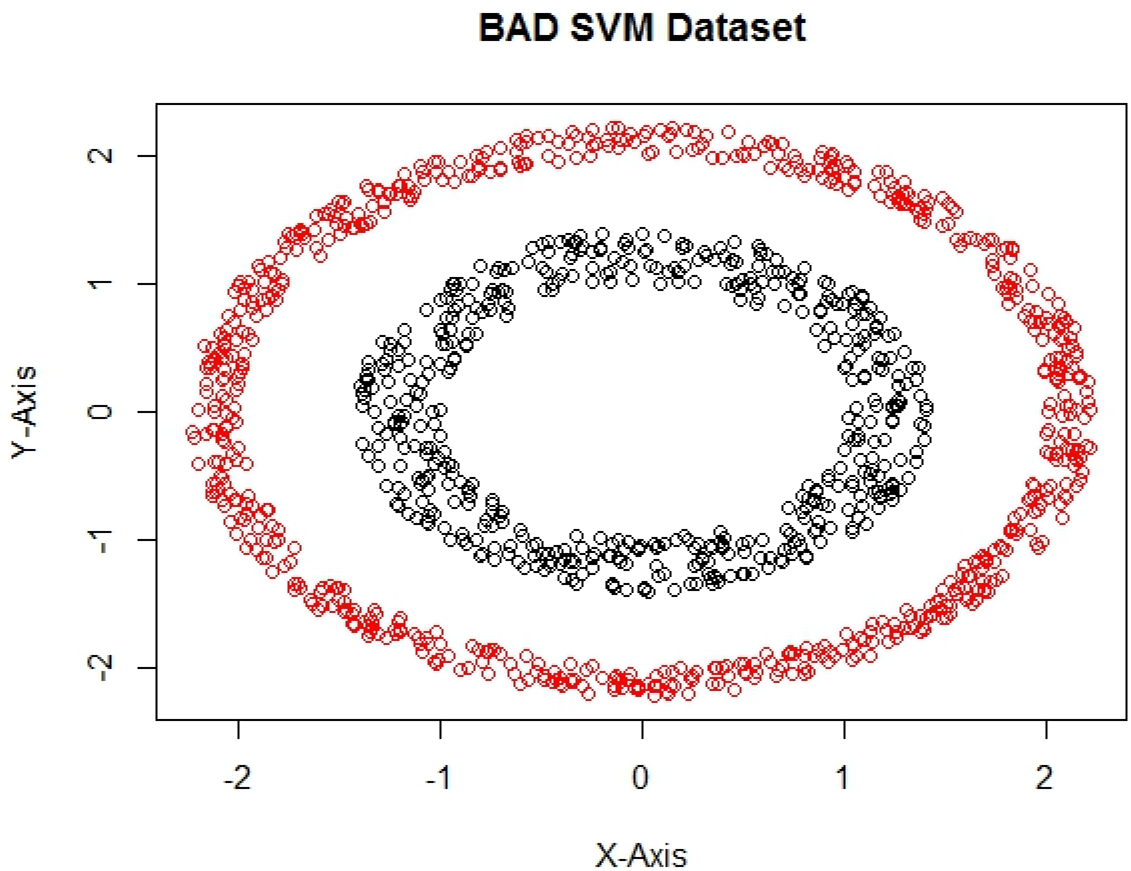
Above dataset is a bad data for K-means clustering algorithm because K-means works best for equal density cluster and in the given data, there are two clusters of varying density. So k-means algorithm won't work properly on the given data and inspite of two separate clusters.

(b) BAD_pca: The data set for which the Principal Component Analysis (PCA) dimension reduction method upon projection of the original points into 1-dimensional space (i.e., the first eigenvector) will not perform well.



Above dataset is bad for principal component analysis because when reducing from two dimensions to one dimension , maximum variability of data is not captured and much of the information is lost.

(c) BAD_svm: The data set for which the linear Support Vector Machine (SVM) supervised classification method using two classes of points (positive and negative) will not perform well.



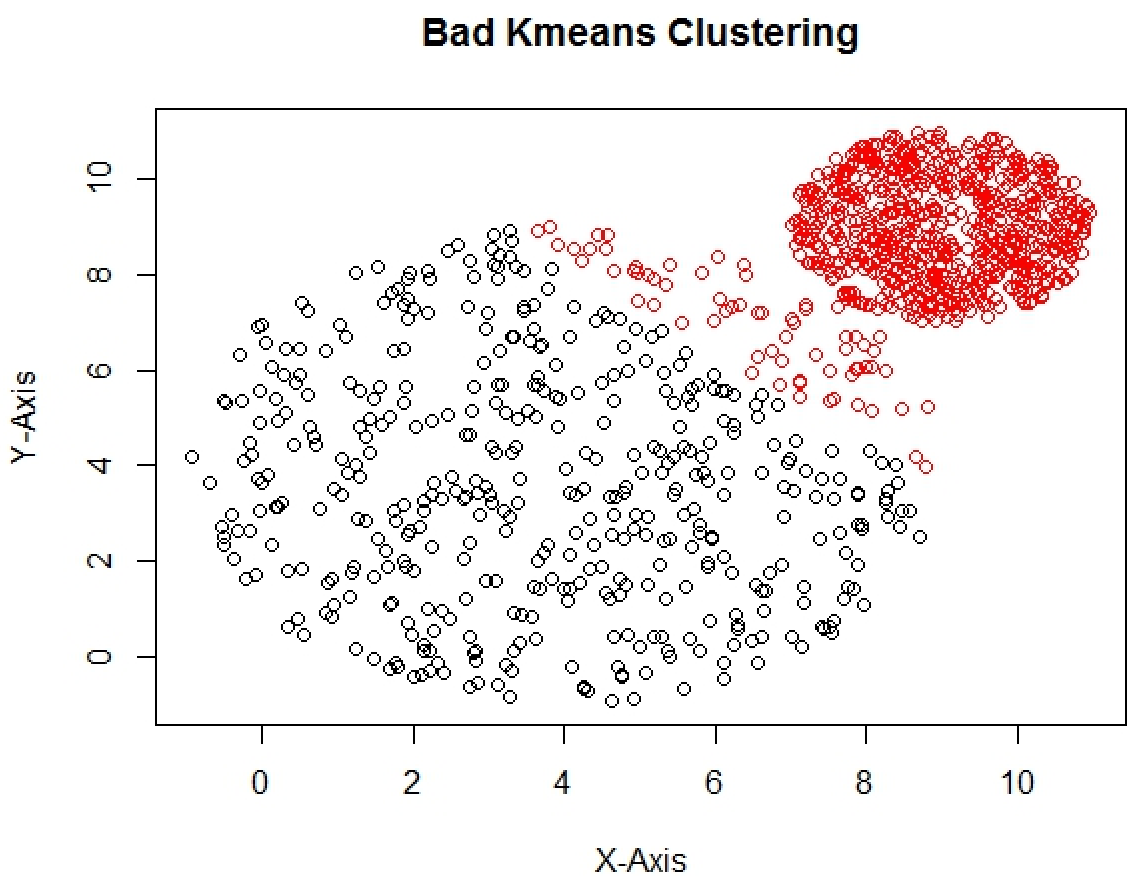
Above dataset is bad for Support Vector Machines because, there is no linear decision boundary that can separate the two clusters.

Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

Exercise 2: Evaluating the "badness" of the data mining methods. Write a script that uses the BAD data set in Exercise 2, runs the corresponding data mining method, produces the output from the method, and evaluates how bad the performance of this method is. You may use various performance metrics to assess each method (e.g., the variance, precision, recall, F1 measure).

1. **BAD_kmeans Clustering, Kernel Tricks and Performance Metrics**

Following is the output of the k-means clustering on the above bad kmeans dataset:



Performance of the above k-means clustering:

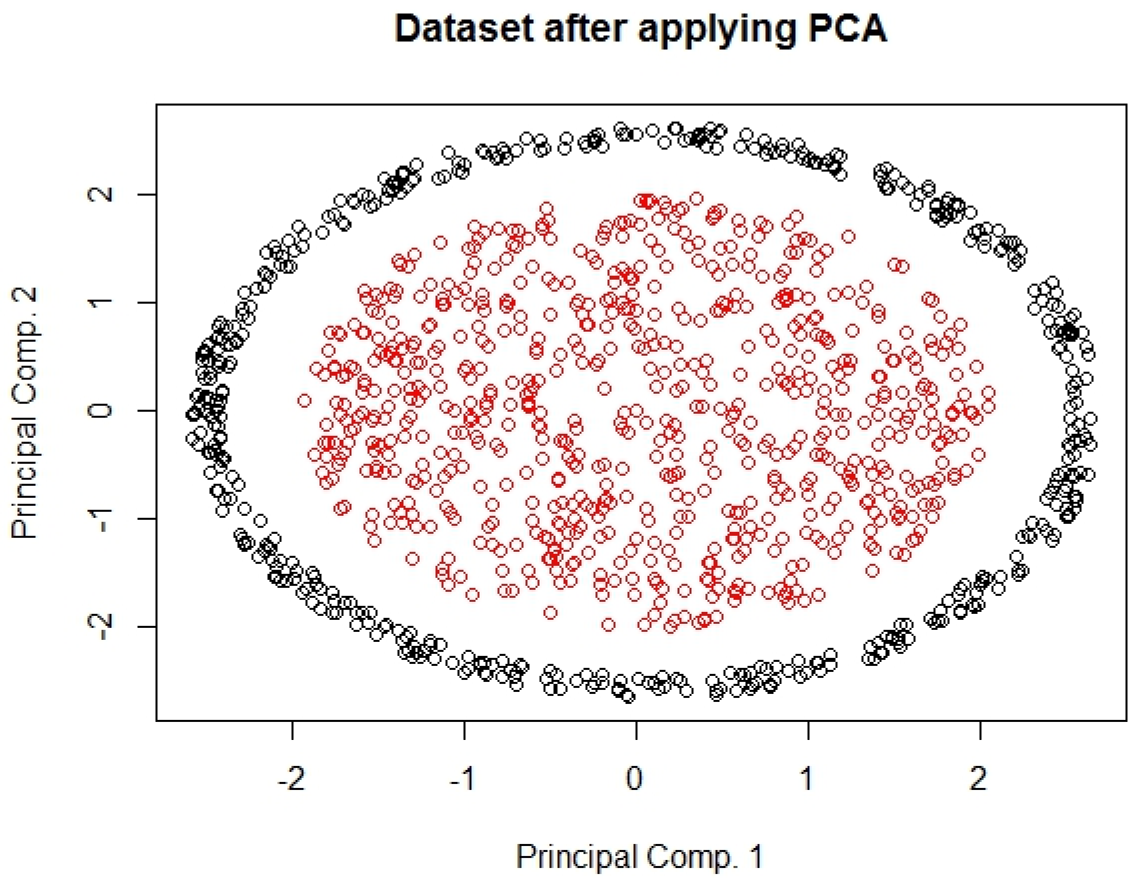
Accuracy	Precision	F-Measure	Jaccard Coefficient	Error Rate
0.9491667	0.878	0.9350373	0.9198423	0.05083333

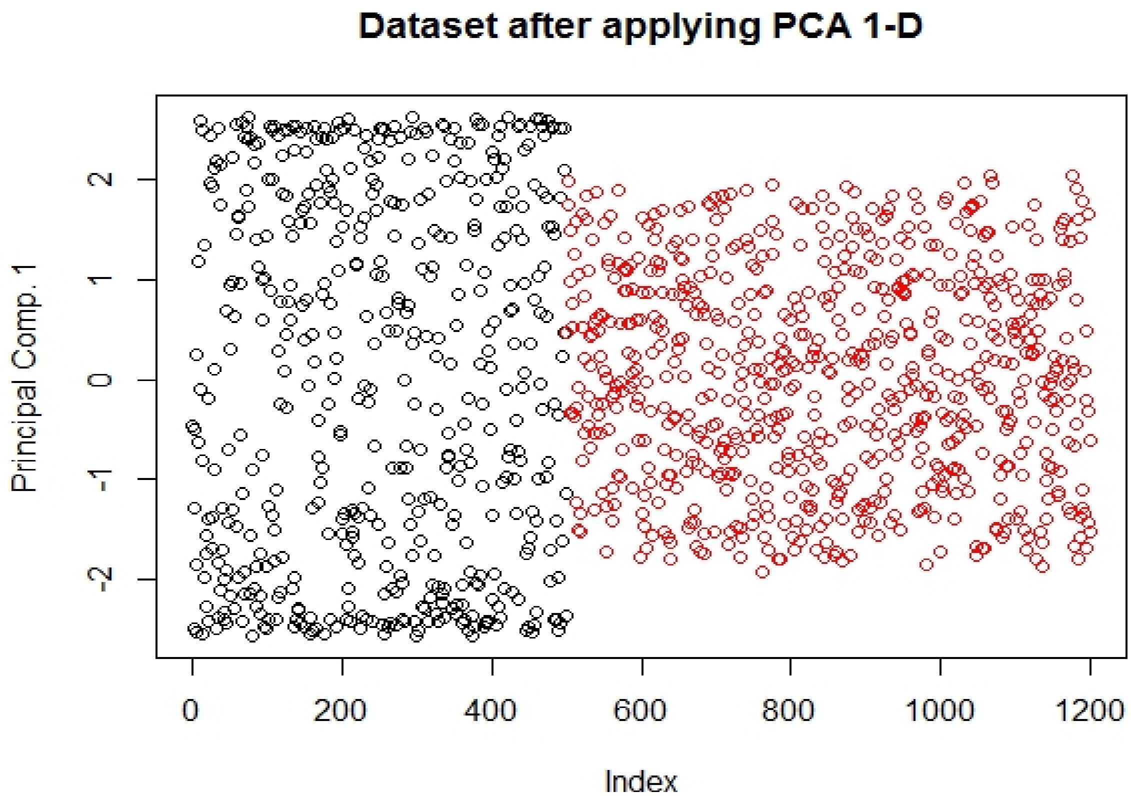
Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

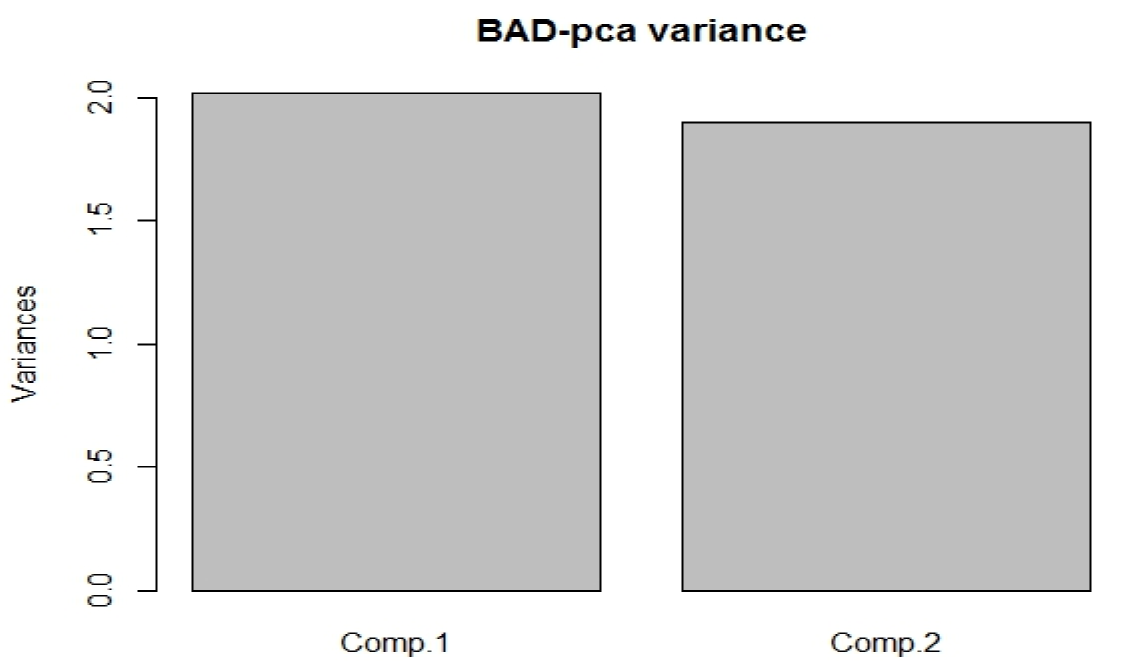
Confusion Matrix:

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	700	61
Ground Truth Cluster 2	0	439

2. **BAD_pca, Kernel Tricks and Performance Metrics:**
Following is the output of the simple pca applied on the above mentioned bad_pca dataset:




Dataset after applying PCA 1-D
Principal Comp. 1
Index
0
200
400
600
800
1000
1200
-2
-1
0
1
2

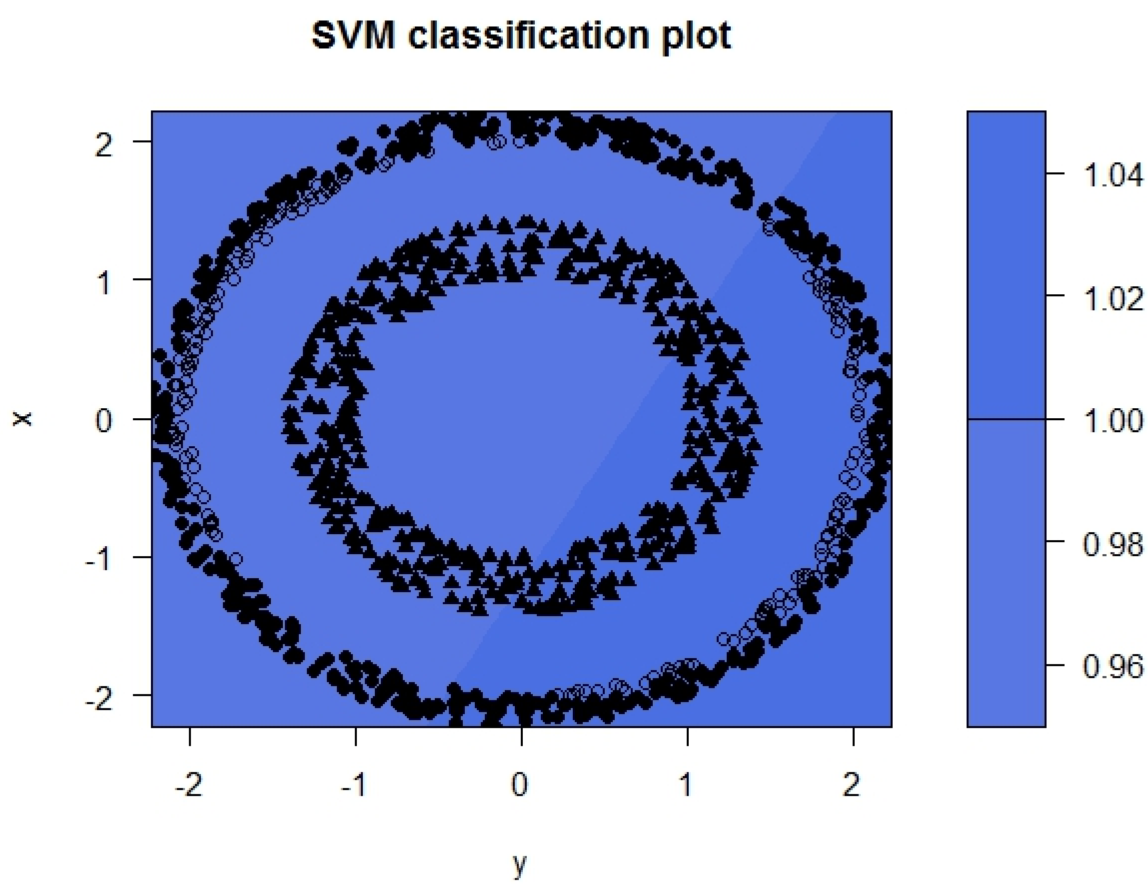

BAD-pca variance
Variances
0.0
0.5
1.0
1.5
2.0
Comp.1
Comp.2

Performance Metrics:

	Principal Comp. 1	Principal Comp. 2
Standard Deviation	1.4204530	1.3772486
Proportion of Variance	0.5154391	0.4845609
Cumulative Proportion	0.5154391	1.0000000

This performance metric clearly states that Principal Component 1 captures almost 51% of the original data variability while Principal Component 2 captures the rest of the data variability.

3. **BAD_SVM, Kernel Tricks and Performance Metrics:**
   Following is the output of the simple linear svm applied on the above mentioned bad_svm dataset:



This plot clearly shows that Linear SVM is not able to draw a linear decision boundary to divide two clusters of data.

Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

Performance Metrics:

- Number of Support Vectors: 1003 (83.5%)

Accuracy	Precision	F-Measure	Jaccard Coefficient	Error Rate
0.583333	0	Undefined	0	0.4166667

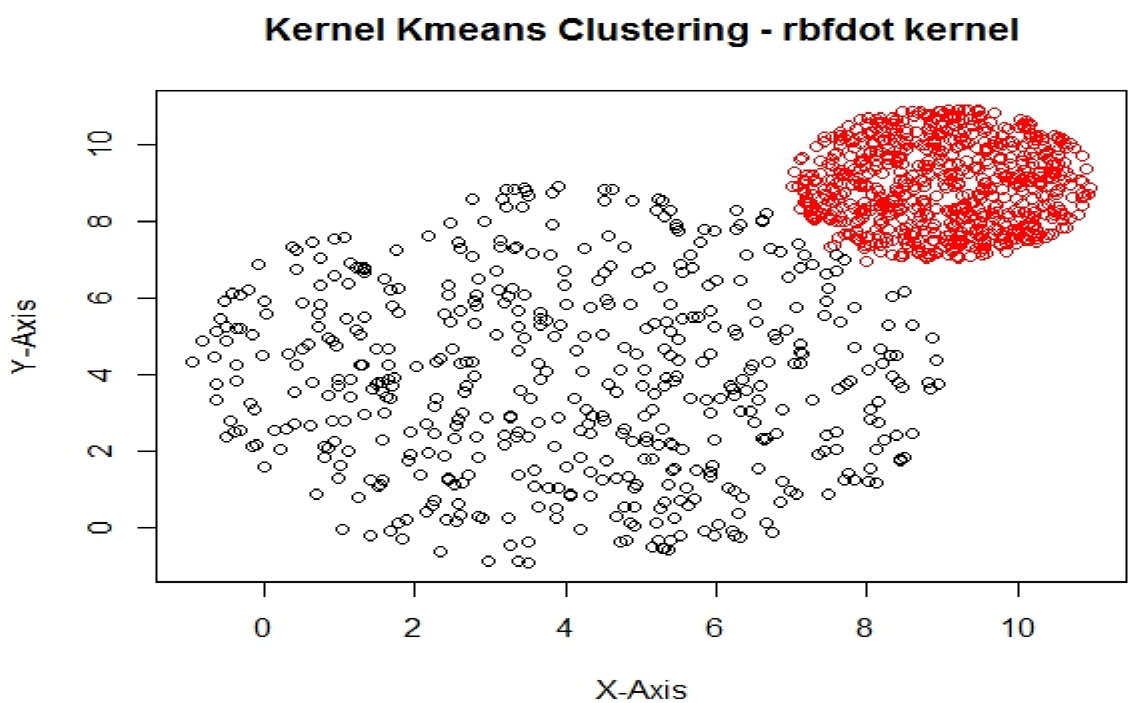
Confusion Matrix:

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	0	0
Ground Truth Cluster 2	700	500

Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

Exercise 3: Kernelizing the methods. Write a script that uses the kernelized version of each of the data mining method in Exercise 2
(a) Choose at least two kernels for each of the methods.
(b) Use the same performance metrics as in Ex. 2, and compare the performance obtained by the methods after applying the kernel trick versus the original un-kernelized versions of the techniques.
(c) Do you observe the difference in performance when you use different kernels?
(d) What are the best performance results do you get by playing with different kernels and kernel parameters? Also, make sure to report the number of support vectors for the SVM (the good rule of thumb is to strive for no more than 35%-50% support vectors to avoid model overfitting.

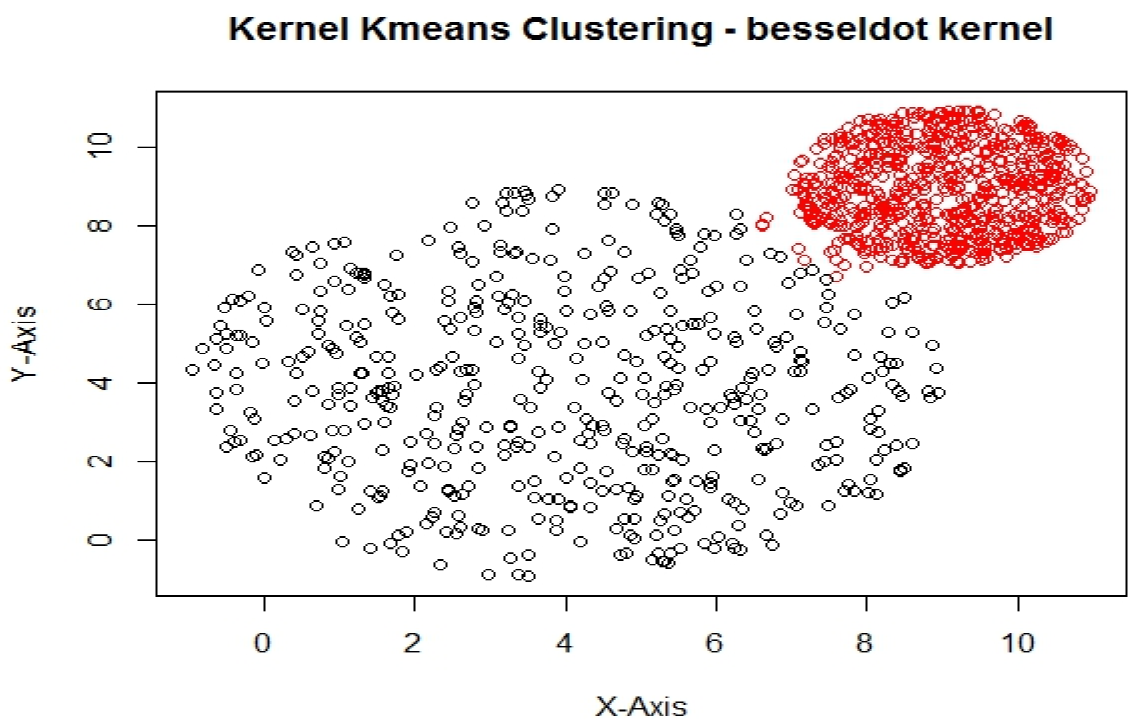
1. Kernelized K-means on bad_kmeans dataset and Performance Evaluation:



Confusion Matrix:

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	700	10
Ground Truth Cluster 2	0	490

Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2



Confusion Matrix:

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	700	11
Ground Truth Cluster 2	0	489

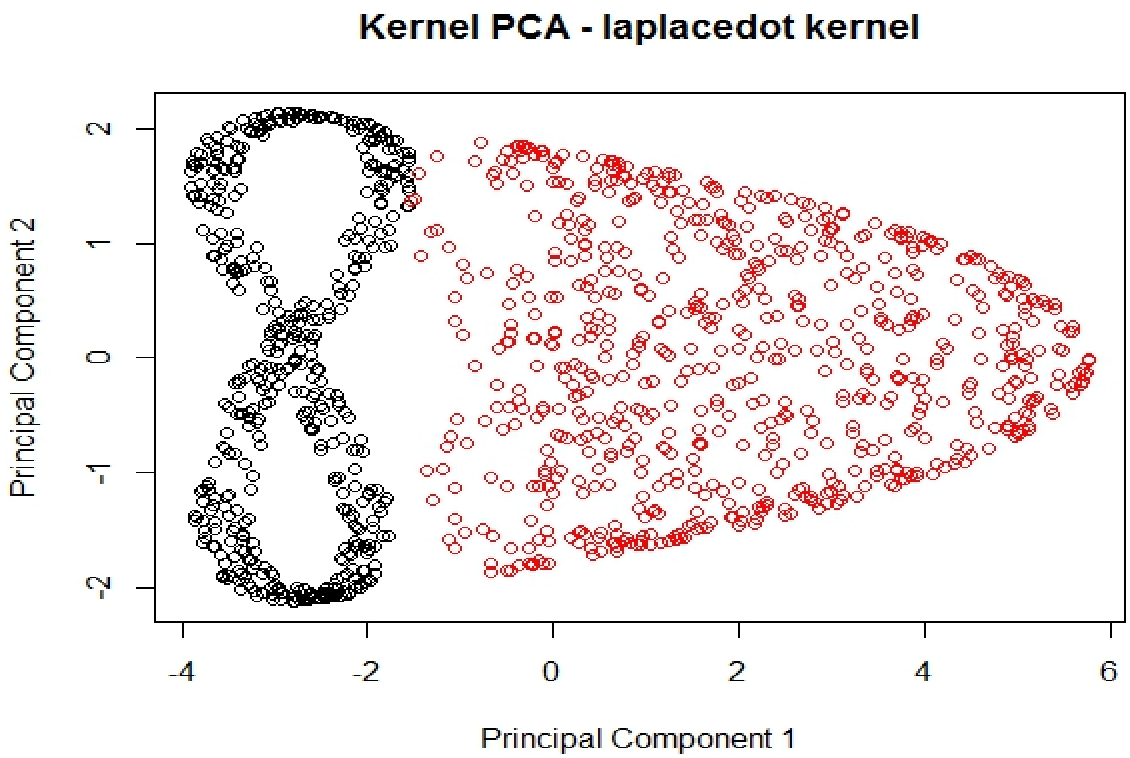
Performance Evaluation:

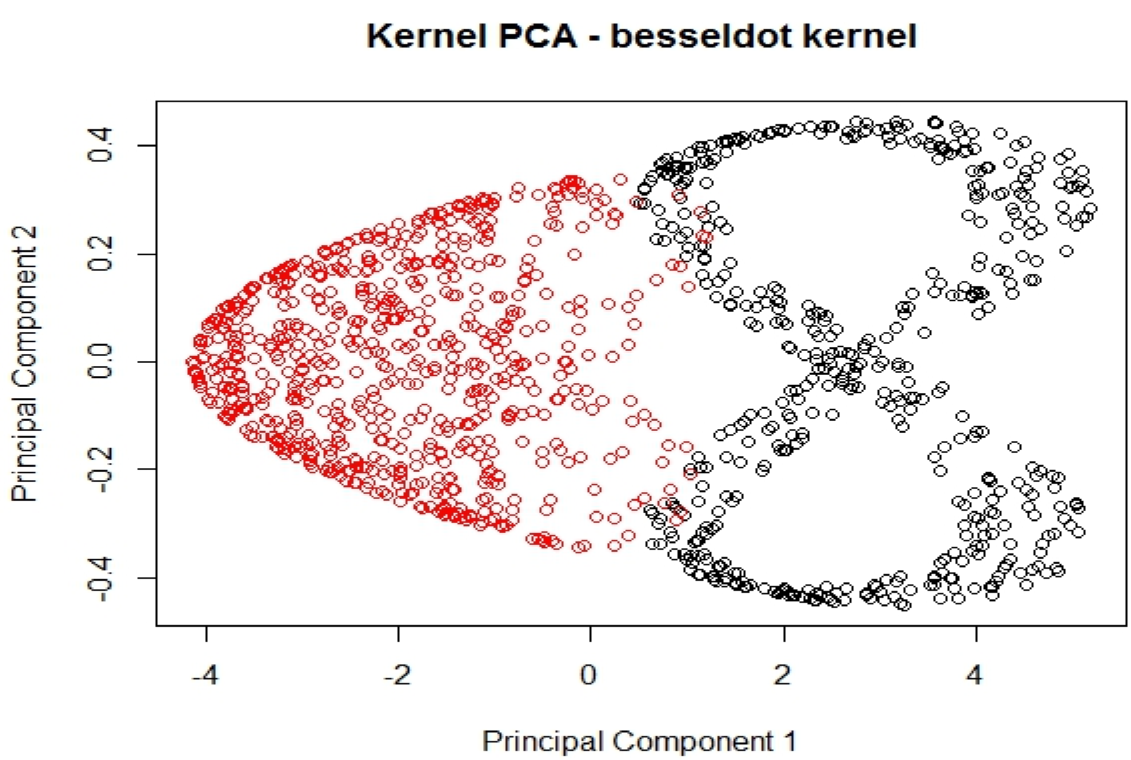
K-Means Type	Accuracy	Precision	F-Measure	Jaccard Coefficient	Error Rate
UnKernelized	0.9491667	0.878	0.9350373	0.9198423	0.05083333
rbfdot kernel	0.9916667	0.98	0.989899	0.9859155	0.008333333
besseldot kernel	0.9908333	0.978	0.9888777	0.9845288	0.009166667

A marginal difference is observed in the performance when using different kernels for k-means. Kernelized k-means very beautifully classifies both the different density clusters. As observed in the above table, Radial basis kernel and besseldot kernel outperforms the un-kernelized k-means version and almost correctly identifies both the clusters 100% accurately.

Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

2. Kernelized K-pca on bad_pca dataset and Performance Evaluation:





Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

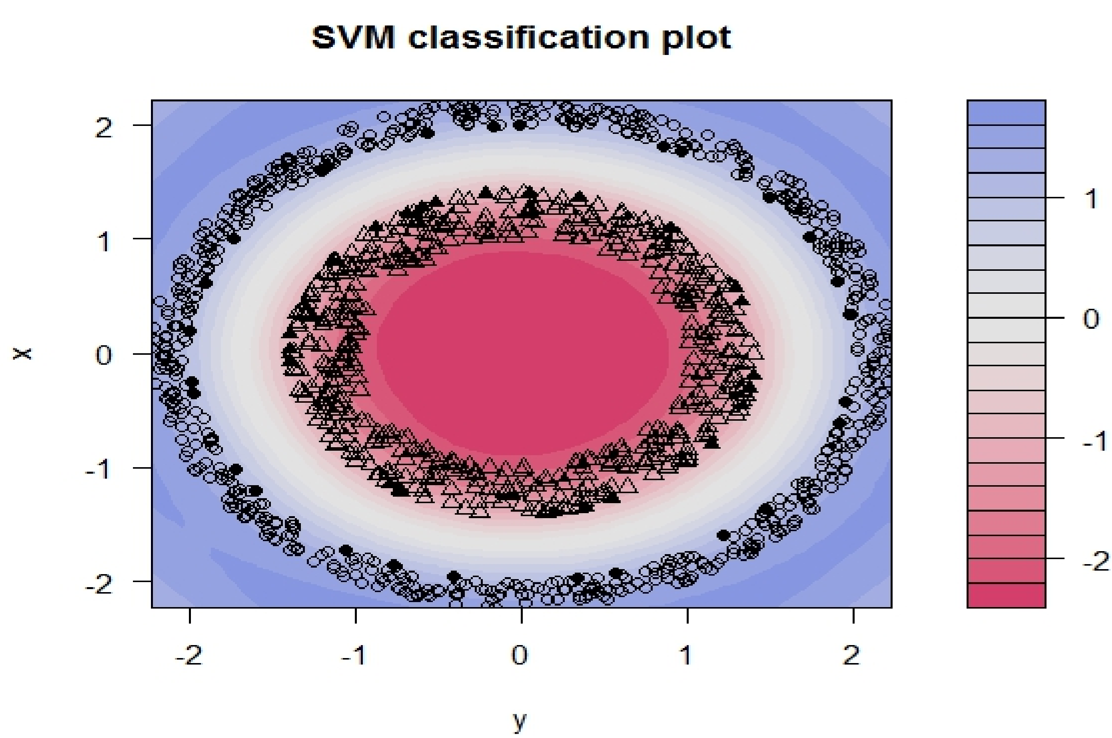
Performance Evaluation:
As observed above, after applying kernel tricks, when we transform the 2 - dimensional data into some high dimensional space, the two concentric circles which were previously linearly inseparable, can now be easily separated using the original SVM or other data mining techniques to draw a decision boundary between two clusters.

There is a great performance difference after using the Laplace and Bessel kernel method for Principal Component Analysis(PCA).

Proportion of Variation	Principal Component 1	Principal Component 2
Un-Kernelized PCA	0.5154391	0.4845609
LaplaceDot Kernel	0.4374287	0.4078973
BesselDot Kernel	0.5099064	0.4792058

As seen in the above table, the kernelized pca captures almost 98% of the variability of data in the high dimension space and yet made the bad_pca data linearly separable.

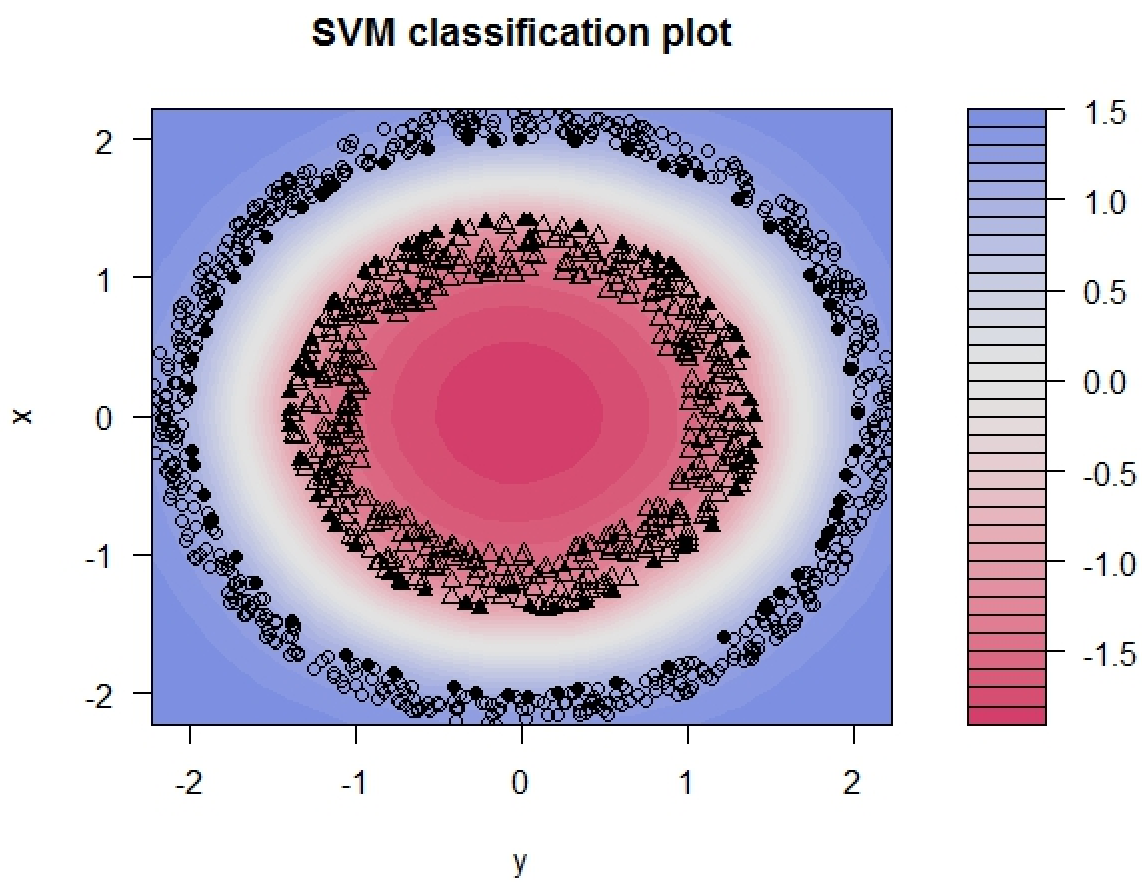
3. **Kernelized SVM on bad_svm dataset and Performance Evaluation:**



- Number of support vectors: 120 (10%)

Confusion matrix: (Above Laplace Dot Kernel)

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	500	0
Ground Truth Cluster 2	0	700



- Number of support vectors: 50 (4.2%)

Confusion matrix: (Above RBF Dot Kernel)

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	500	0
Ground Truth Cluster 2	0	700

Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

Performance Evaluation:

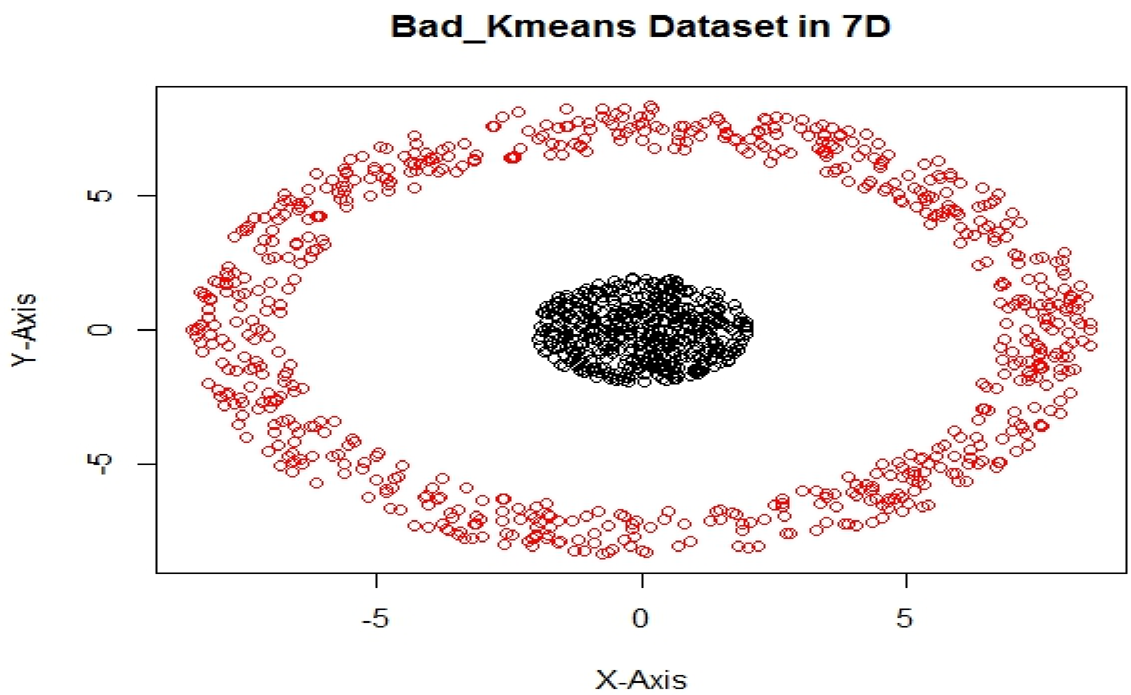
As seen in the above two SVM classification plot using laplace and rbf kernel, the classification is 100% accurate , which is way better than the Un-Kernelized (or Linear/ Vanilla Dot) SVM

	Accuracy	Precision	F-Measure	Jaccard Coefficient	Error Rate
Linear SVM	0.583333	0	Undefined	0	0.4166667
Laplace Kernel SVM	1	1	1	1	0
RBF Kernel SVM	1	1	1	1	0

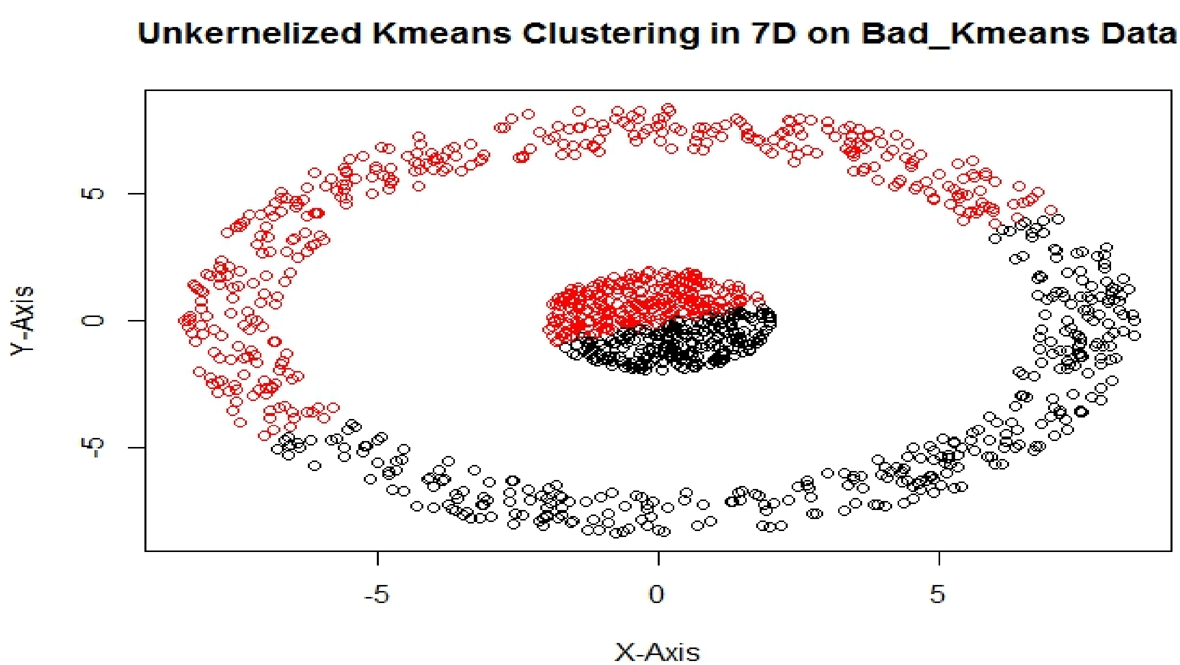
Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

Exercise 4: Pipelining. Dimension reduction is often used as the key data preprocessing step to other data mining techniques downstream of end-to-end data analysis.

(a) Generalize your BAD_kmeans data set to very high-dimensional space (d>>2).



(b) Show that the kmeans clustering method does not perform well on that data.



Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

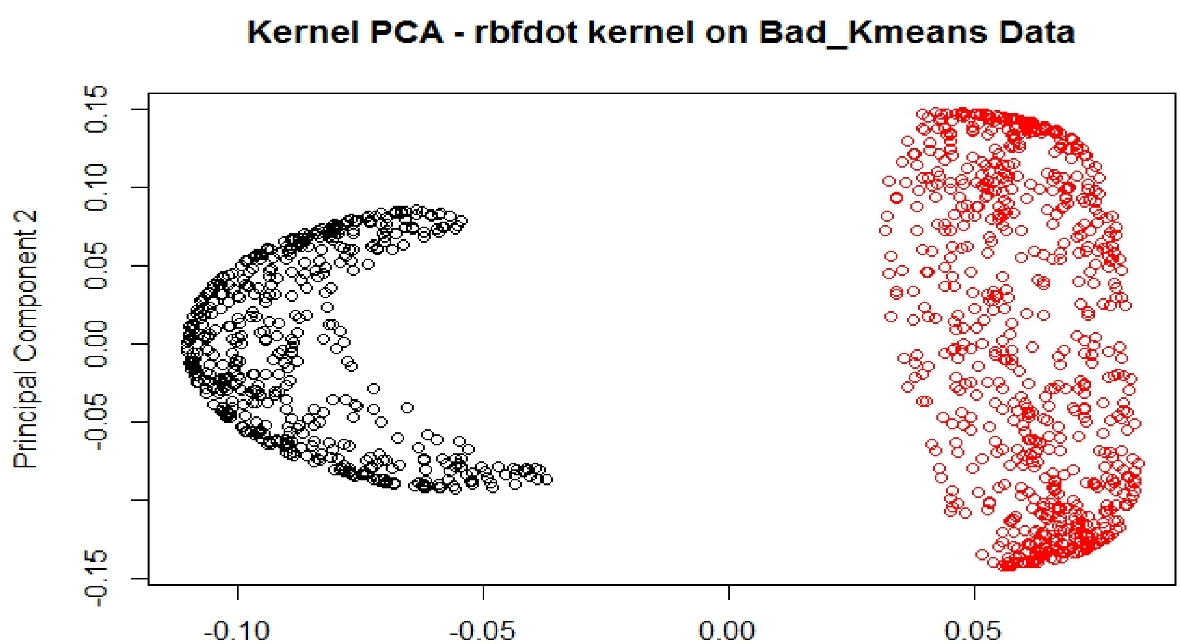
K-means performs horribly on the above high dimensional bad-k means data. Below is the confusion matrix and the performance metrics of k means.
Confusion matrix: (Un-kernelized K-means)

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	234	337
Ground Truth Cluster 2	266	363

Performance Metrics:

K-Means Type	Accuracy	Precision	F-Measure	Jaccard Coefficient	Error Rate
Un Kernelized	0.4975	0.468	0.4369748	0.2795699	0.5025

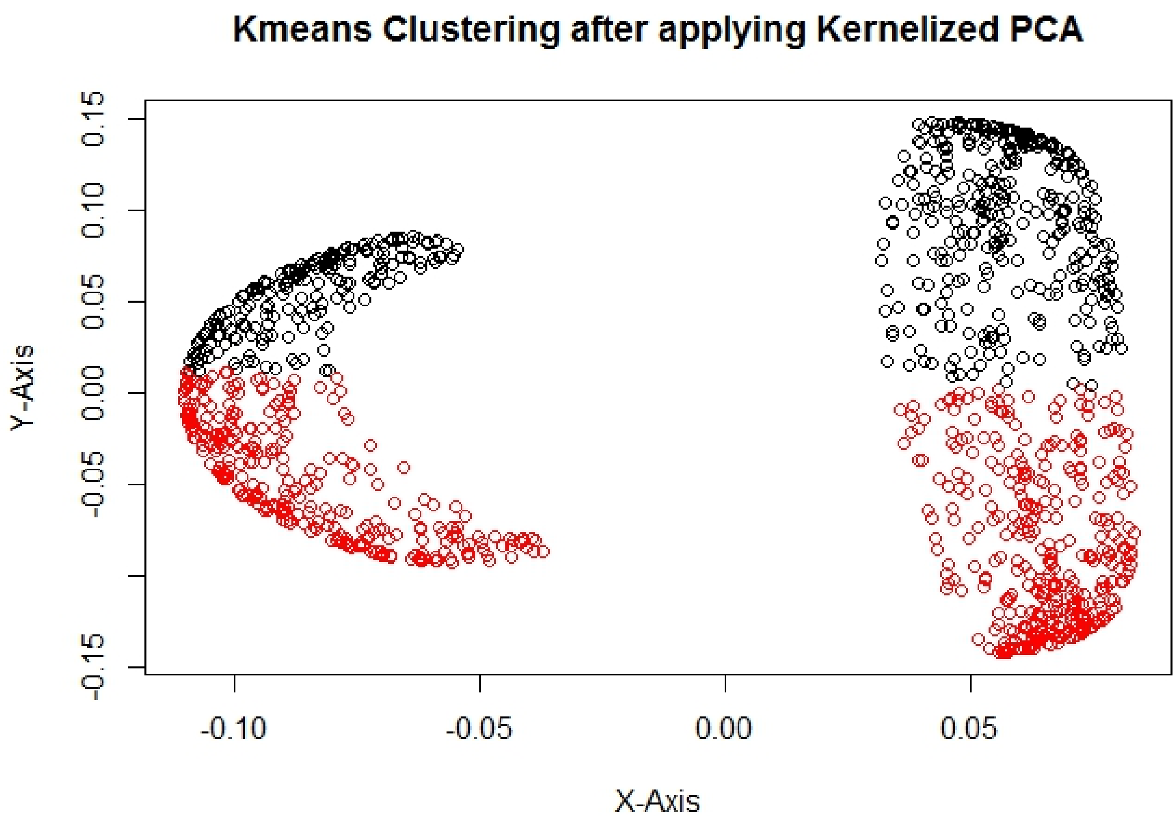
(c) Apply the kernel PCA method to this high dimensional data and identify the number (m<<d) of principal components (i.e., eigenvectors) that provide a reasonably good low-dimensional approximation to your data (i.e., based on eigenvalue distribution). How much total variability of the data will be preserved upon using this low-dimensional representation?



Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

After applying rbfdot Kernel PCA on the given bad k means data, the two clusters are clearly separated in the low dimensions. Also much of the data is preserved.
Variability preserved using this low dimensional data is : ~63% (0.6305741) (m = 3 << 7)

(d) Project your original data onto the top m eigenvectors corresponding the largest eigenvalues.
(e) Run the k means clustering algorithm on the projected low dimensional data.



(f) Compare the performance of the k-means on d-dimensional original data vs. the m-dimensional projected data. Has the performance improved?

Confusion matrix: (Kernelized K-means)

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	204	344
Ground Truth Cluster 2	296	356

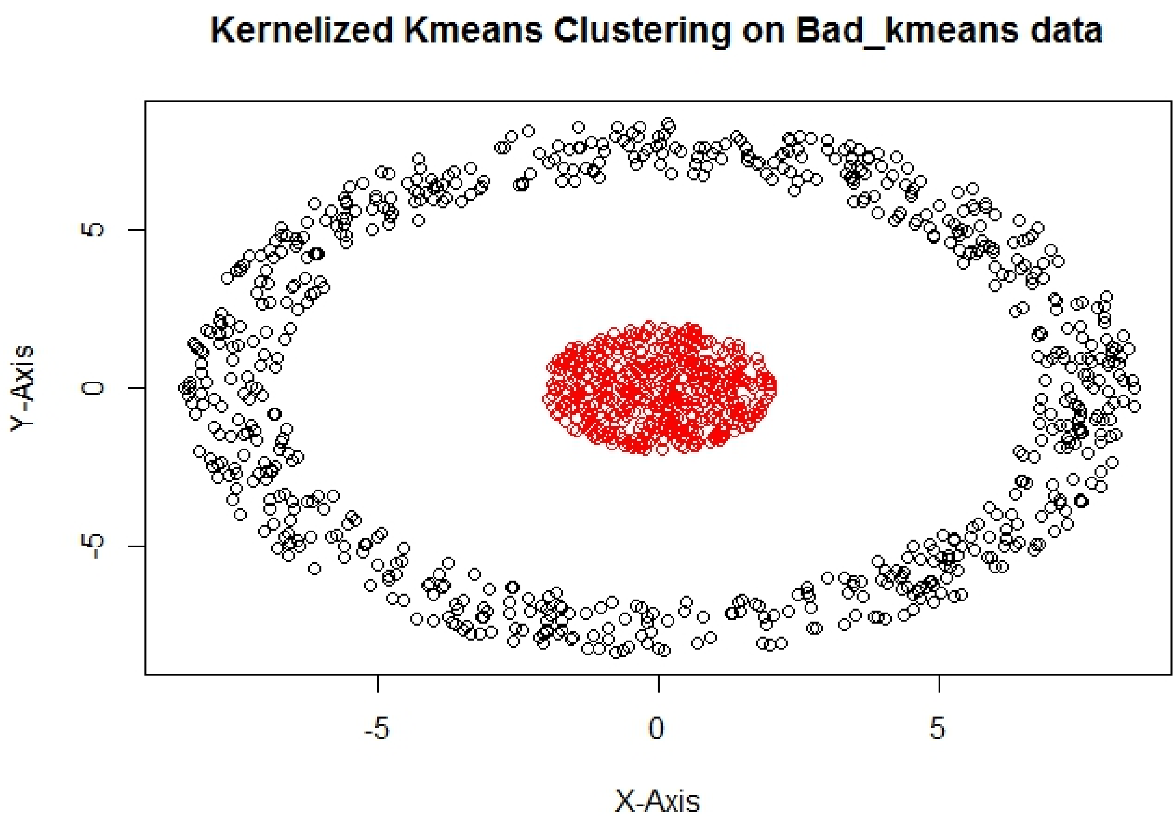
Performance Metrics:

Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

K-Means Type	Accuracy	Precision	F-Measure	Jaccard Coefficient	Error Rate
Kernelized	0.4666667	0.408	0.389313	0.2417062	0.5333333
Un Kernelized	0.4975	0.468	0.4369748	0.2795699	0.5025

Above performance metrics clearly shows that, Kernel PCA in the data preprocessing step helps in separating clusters, but applying k-means on the kernelized data doesnt improve much performance. This is because of the limitation of the k-means to cluster only the globular and equal density clusters.

(g) If you run the kernel k means clustering method on the original data, will get better/worse performance? Can you discuss the pros and cons of using kernel k means on the original data directly versus applying the kernel pca as the pre-processing step and then running the k means on the low-dimensional data.



Applying kernelized k-means on the bad_kmeans data performs amazingly compared to that of the k means on the kernelized pca data as seen in the performance metrics below.

Project Report
Project: Kernelization, Kernel Tricks
Unity Id: ndgandh2

Confusion matrix: (Kernelized K-means)

	Predicted Cluster 1	Predicted Cluster 2
Ground Truth Cluster 1	700	0
Ground Truth Cluster 2	0	500

Performance Metrics:

K-Means Type	Accuracy	Precision	F-Measure	Jaccard Coefficient	Error Rate
Kernelized	1	1	1	1	0

Pros and cons of using kernel k means on the original data directly versus applying the kernel pca as the pre-processing step and then running the k means on the low-dimensional data are as follows:

1. Kernel k-means uses the 'kernel trick' (i.e. implicitly projecting all data into a non-linear feature space with the use of a kernel) in order to deal with one of the major drawbacks of k-means that is that it cannot capture clusters that are not linearly separable in input space. So kernel k means performs excellently on the given bad_kmeans data.
2. Using Kernel PCA as a data preprocessing step, to separate clusters which are non linearly bounded, improves the performance. But k means has a limitation of clustering only the globular and equal density clusters. So even after applying kernel PCA, k-means performs poorly. But in case, SVM was used to separate the clusters, job would be done excellently as SVM uses linear decision boundaries.

References:

[1] http://cran.r-project.org/web/packages/kernlab/vignettes/kernlab.pdf
[2]http://www.csc.ncsu.edu/faculty/samatova/practical-graph-mining-with-R/slides/pdf/Performance_Metrics_For_Graph_Mining_Tasks.pdf