Pattern Recognition & Machine Learning

<u>Lab-9</u>

“Support Vector Machines”

Objectives

Following were the required tasks to be fulfilled in Lab-9 :-

1. Use 70-20-10 as Training,Validation and Test Split Percentages. Compare the Performances of Nearest Neighbors, Perceptron and SVM Classifiers, by using validation data for Grid Search and incorporating Normalization process in comparisons too. Also, Implement any of 2 from OVA/OVO/DAG configurations and compare the results.

2. Using the same split percentages again, compare the performances of SVM Classifier with different kernels like Linear, Polynomial and RBF (Radial Basis Function). Use Grid-Search on Validation Data to tune hyper-parameters. Report the number of support vectors in each case and if possible, plot the 2-D representation of Hyperplanes, for separating classes.

Datasets

Following were the Datasets, used in Lab-9 related Tasks :-

1. MNIST Dataset - Using Sklearn’s (fetch OpenML)
2. Diabetes Dataset - dataset

Dependencies

1. Pandas
2. Sklearn
3. Matplotlib
4. Mlxtend

Preprocessing Methods

1. For Comparison Purposes, Dataset was analysed originally as well as Standardized and MinMaximised, using Sklearn. Their Respective Formulas are as following :-

Standardization:

$$z = \frac{x - \mu}{\sigma}$$

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Where :-

X -----> Feature
Myu (u) ---> Mean of Feature
Sigma -----> Standard Deviation of Feature
Z ------> Standardized Feature
X (min) -----> Minimum value of Feature
X (max) ----> Maximum value of Feature
X (scaled) -----> MinMaximised Feature

2. For Hyperplane and DataPoint representations, Principal Component Analysis was used.

Support Vector Machine on MNIST Dataset

Procedures

1. Picked classes {0,1,2,3,4} and their data points, from other classes.

2. Made 3 types of Datasets :- Original, Standardized and Min-Maximised. Then, they were splitted into the above-mentioned split percentages.

3. Using Validation Data of each kind, Grid Searches were performed for Nearest Neighbors, Perceptron and SVM Classifiers.

4. Picked One-v/s-All and One-v/s-One Configurations and Compared the results of the above estimators.

Hyper-Parameters for each Classifier

For Nearest-Neighbor Classifier :-

1. N_neighbors : [2,3,4,5,6]
2. Weights : ‘Uniform’, ‘Distance”

For Perceptron :-

1. Kernel : ‘Linear’, ‘Polynomial’, ‘Radial Basis Function’
2. C : [0.8,1.0,1.2]
3. Tol : [0.001,0.01,0.1,1]

For Support Vector Machines :-

1. Penalty : ‘L1’, ‘L2’
2. Class_Weight : ‘None’, ‘Balanced’
3. Tol : [0.001,0.01,0.1,1]

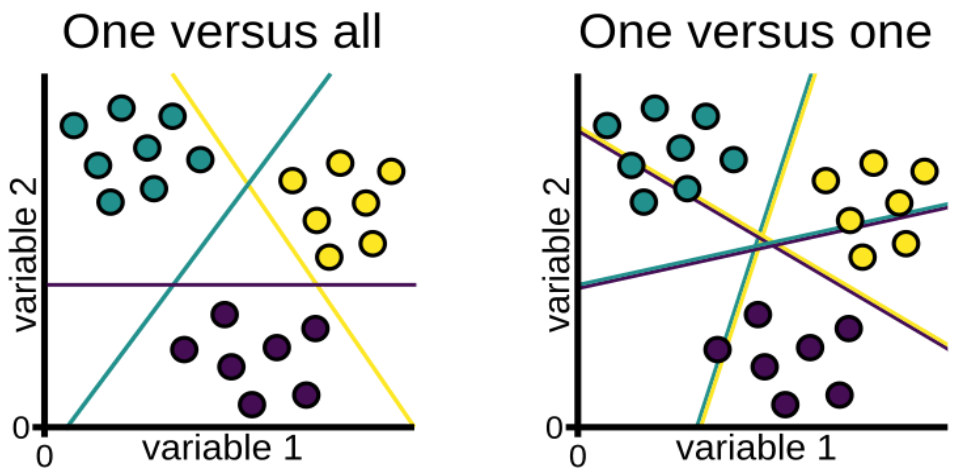
Following was the table obtained, after running multiple Grid-Searches on Validation Data :-

Dataset Type	Nearest Neighbor Classifier	Perceptron	Support Vector Machines
Original	*n_neighbors: 4* *weights: distance*	*class_weight: None* *penalty: L1* *tol:* ***(No Effect)***	*C: 1.2* *kernel: rbf* *tol: 0.1*
Standardized	*n_neighbors: 2* *weights: distance*	*class_weight: None* *penalty: L1* *tol:* ***(No Effect)***	*C: 1.2* *kernel: rbf* *tol:* **0.001/0.01**
Min-Maximised	*n_neighbors: 2* *weights: distance*	*class_weight: None* *penalty: L1* *tol:* ***(No Effect)***	*C: 1.2* *kernel: rbf* *tol: 0.1*

One-v/s-All Classifier V/S One-v/s-One Classifier

One v/s All Classifier constructs the Binary Classifiers in such a way that a Single Binary Classifier separates Kth Class and Non Kth Classes (Where K is any integer less than Total Number of Classes). Here, the number of such classifiers equals to N-1.

One v/s One Classifier constructs the Binary Classifiers in such a way that a Single Binary Classifier separates a Couple of Consecutive Classes. Here, the number of such classifiers equals to N(N-1)/2 .



Performance Metrics

During Cross-Validation, the Performance Metrics were built for Multiple Grid-Searches as following :-

Dataset Type	Nearest Neighbor Classifier	Perceptron	Support Vector Machines
Original	***Mean_Score :0.98460638*** ***Std_Score :0.00351432***	***Mean_Score :0.95493988*** ***Std_Score :0.00628369***	***Mean_Score :0.98586669*** ***Std_Score :0.00422933***
Standardized	***Mean_Score :0.98390688*** ***Std_Score :0.00290323***	***Mean_Score :0.95004164*** ***Std_Score :0.0054523***	***Mean_Score :0.98334705*** ***Std_Score :0.00184836***
Min-Maximised	***Mean_Score :0.98586581*** ***Std_Score :0.00252107***	***Mean_Score :0.95801817*** ***Std_Score :0.002429***	***Mean_Score :0.98544603*** ***Std_Score :0.00301626***

Regarding Test Performance, following were the key-findings for Both Configurations :-

Dataset Type	Nearest Neighbor Classifier	Perceptron	Support Vector Machines
Original	***OVO Test Accuracy :99.07%*** ***OVA Test Accuracy :99.04%***	***OVO Test Accuracy :96.05%*** ***OVA Test Accuracy :94.20%***	***OVO Test Accuracy :99.32%*** ***OVA Test Accuracy :99.24%***
Standardized	***OVO Test Accuracy :99.3%*** ***OVA Test Accuracy :99.3%***	***OVO Test Accuracy :96.81%*** ***OVA Test Accuracy :95.13%***	***OVO Test Accuracy :99.18%*** ***OVA Test Accuracy :99.18%***
Min-Maximised	***OVO Test Accuracy :99.04%*** ***OVA Test Accuracy :99.04%***	***OVO Test Accuracy :96.41%*** ***OVA Test Accuracy :95.88%***	***OVO Test Accuracy :98.99%*** ***OVA Test Accuracy :99.04%***

Support Vector Machine on Diabetes Dataset

Procedures

1. Made 3 types of Datasets :- Original, Standardized and Min-Maximised. Then, they were splitted into the above-mentioned split percentages.

2. Using Validation Data of each kind, Grid Searches were performed for different types of SVM Classifiers and Preprocessing Types

3. Using Principal Component Analysis, 2-D Hyperplanes were Visualised, according to the type of Data-Preprocessing.

4. After Finalising best estimators for each kind, Number of Support Vectors were obtained for each kind of SVM Classifiers.

Hyper-Parameters for Support Vector Machines

Following were the Hyper-Parameters, taken for Grid-Search Cross Validation :-

1. C : `[0.001,0.01,0.1,1,10,100,1000]`
2. Degree : `[2,3,4,5,6]` (Works for Kernel = 'Polynomial')
3. Tol : `[0.01,0.1,1,10,100]` (Insignificant for Kernel = 'Polynomial')

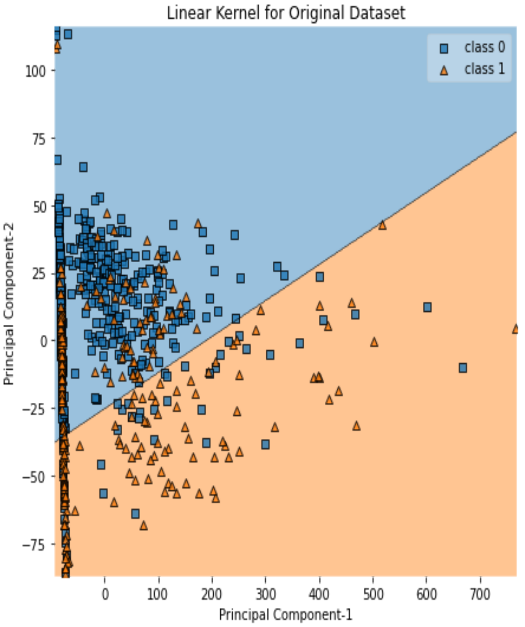
Following was the table obtained, after running multiple Grid-Searches on Validation Data :-

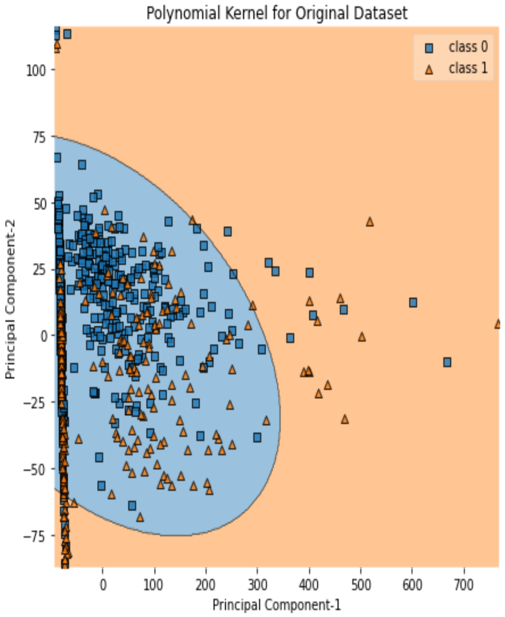
Dataset Type	Support Vector Machine with Linear Kernel	Support Vector Machine with Polynomial Kernel	Support Vector Machine with Radial Basis Function Kernel
Original	*C : 10* *Tol : 1*	*C : 100* *Degree: 2* *Tol: 0.001*	*C : 100* *Tol : 1*
Standardized	*C : 1* *Tol : 1*	*C : 10* *Degree: 2* *Tol: 0.001*	*C : 1* *Tol : 1*
Min-Maximised	*C : 100* *Tol : 0.1*	*C : 1* *Degree: 2* *Tol: 0.001*	*C : 10* *Tol : 0.01*

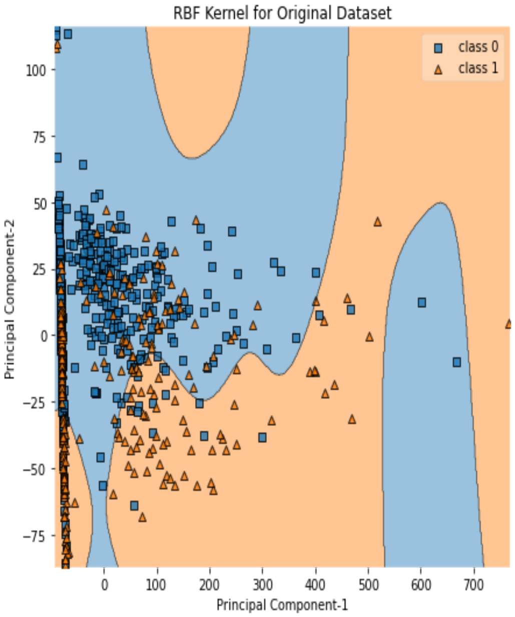
2-D HyperPlane Visualisations

Following were the Graphs for every configurations of Data-Preprocessing with Kernels :-

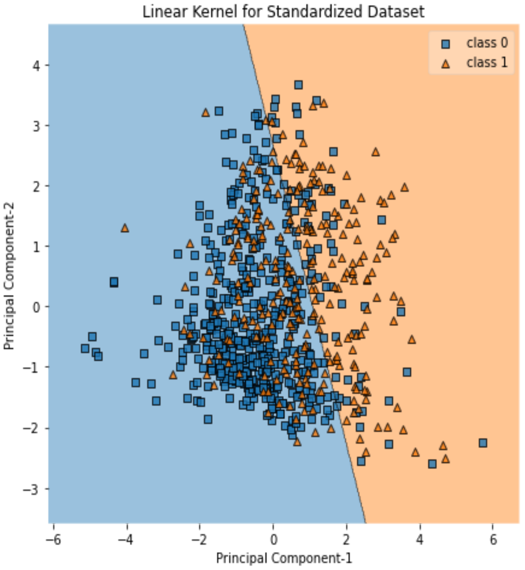
1. For Original Dataset :-

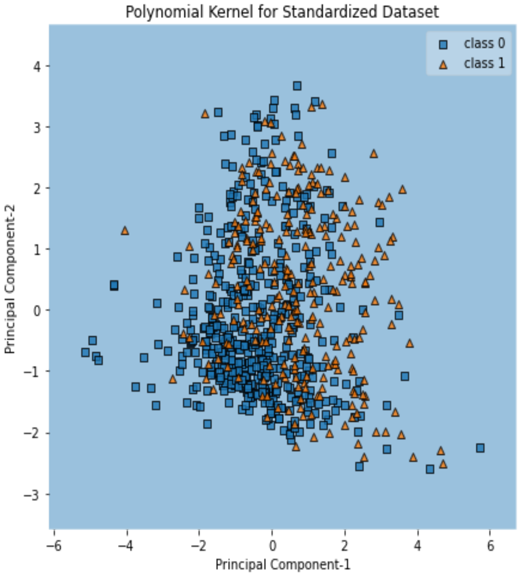


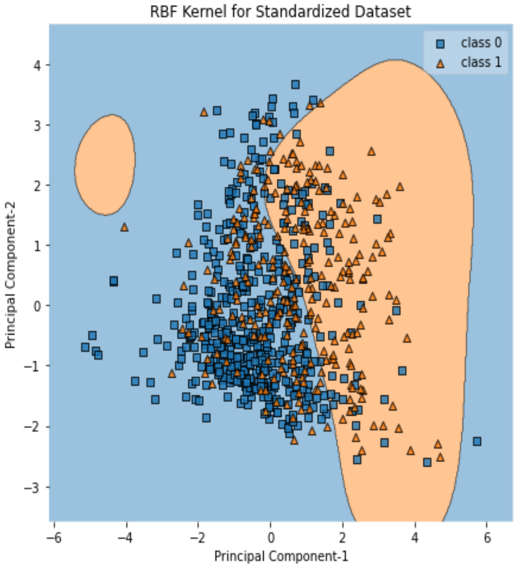




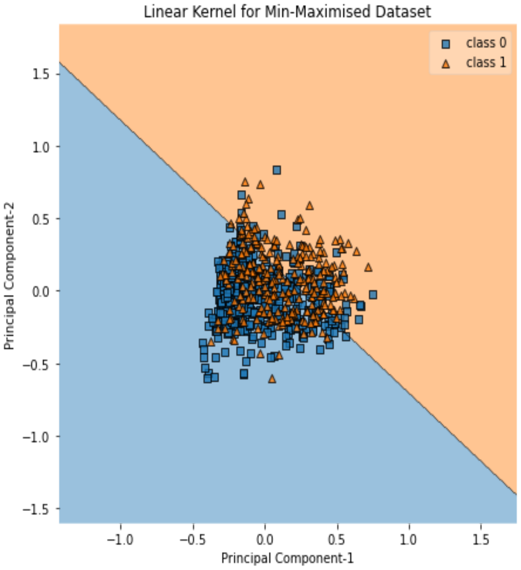
2. For Standardized Dataset :-

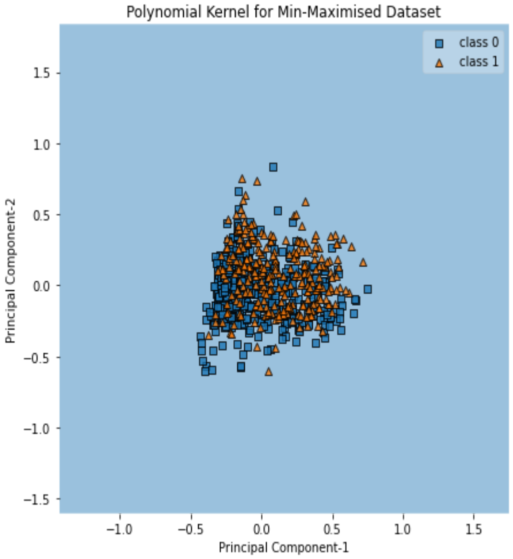


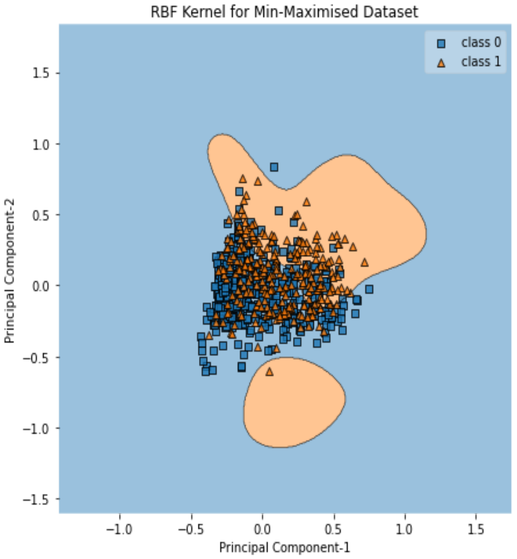




3. For Min-Maximised Dataset :-







Performance Metrics

During Cross-Validation, the Performance Metrics were built for Multiple Grid-Searches as following :-

Dataset Type	Support Vector Machine with Linear Kernel	Support Vector Machine with Polynomial Kernel	Support Vector Machine with Radial Basis Function Kernel
Original	***Mean_Score :0.82473118*** ***Std_Score :0.08479229***	***Mean_Score :0.81806452*** ***Std_Score :0.07112527***	***Mean_Score :0.79139785*** ***Std_Score :0.0466027***
Standardized	***Mean_Score :0.74602151*** ***Std_Score :0.07240129***	***Mean_Score :0.73892473*** ***Std_Score :0.05994718***	***Mean_Score :0.70666667*** ***Std_Score :0.05806213***
Min-Maximised	***Mean_Score :0.82387097*** ***Std_Score :0.04723552***	***Mean_Score :0.84301075*** ***Std_Score :0.04448547***	***Mean_Score :0.8172043*** ***Std_Score :0.03215755***

Regarding Number of Support Vectors, following were the key-findings for each combination :-

Dataset Type	Support Vector Machine with Linear Kernel	Support Vector Machine with Polynomial Kernel	Support Vector Machine with Radial Basis Function Kernel
Original	***304***	***288***	***316***
Standardized	***288***	***338***	***274***
Min-Maximised	***297***	***293***	***292***

Regarding Test Performance, following were the key-findings for Both Configurations :-

Dataset Type	Support Vector Machine with Linear Kernel	Support Vector Machine with Polynomial Kernel	Support Vector Machine with Radial Basis Function Kernel
Original	***Test Accuracy :75.64%***	***Test Accuracy :80.76%***	***Test Accuracy :82.05%***
Standardized	***Test Accuracy :78.20%***	***Test Accuracy :71.79%***	***Test Accuracy :71.79%***
Min-Maximised	***Test Accuracy :75.64%***	***Test Accuracy :75.64%***	***Test Accuracy :71.79%***

Conclusion

Following were the key-points regarding Lab-9 :-

1. Amongst different classifier types, Nearest Neighbor Classifier performed better than others and Perceptron performed the worst.

2. Most of the time, One v/s One Configuration was better than One v/s All Configuration, despite the computationally expensive run-time by former rather than later.

3. Amongst Hyper-Parameters, Tolerance was least significant in affecting the performance of the model, when compared to Weight Distribution, C, Degree, N_Neighbors, Penalty, Kernel etc.

4. With Difference in Data-Preprocessing Techniques, the visualisations for Data-Points differed too, because Principal Components change with changes in the type of Data Distribution. And in one of the instances, it was hard to plot the Decision Boundary, despite classifying the points upto some extent.

5. It was hard to compare the performance of various kernels, because of varying Data-Preprocessing types. But according to Cross-Validation Scores and somewhat Test Performances, the Linear Kernel performed better than other kernels in “Diabetes Dataset”.