Course Title: Statistical and Syntactic Pattern Recognition (Comp5107)

Final Project Report

Project Title : Classification on Diabetic Retinopathy in an image set

Submitted to – Dr. B. John Oommen

Submitted By – Tansin Jahan (ID: 101065087)

For code reference I am including my GitHub link where the source code of this project can be found - https://github.com/tansinjahan/FinalProject_classifiers

In this final project, I have tried to build a PR system with real life data. We used four classifiers for training and testing purposes.

Traditional classifiers – i. Quadtratic classifier ii. K Nearest neighbour

Linear classifiers – i. The Ho-Kashyap rule ii. Fisher's Discriminant Method

Dataset Description for The Project

I have used Diabetic Retinopathy Debrecen Dataset [1] from https://archive.ics.uci.edu/ml/datasets/Diabetic+Retinopathy+Debrecen+Data+Set for the project. This dataset contains features extracted from the Messidor image set to predict whether an image contains signs of diabetic retinopathy or not.

Data Set Characteristics	Attribute Characteristics:	Number of Instances:	Number of Attributes:	Missing Values:	Associated Tasks:
Multivariate	Integer, Real	1151	20	N/A	classification

Attribute/Feature information:

The dataset contains 20 features where each feature represents information as following –

0) The binary result of quality assessment. 0 = bad quality 1 = sufficient quality.
1) The binary result of pre-screening, where 1 indicates severe retinal abnormality and 0 its lack.
2-7) The results of MA detection. Each feature value stand for the number of MAs found at the confidence levels.
8-15) contain the same information as 2-7) for exudates.
16) The euclidean distance of the center of the macula and the center of the optic disc to provide important information regarding the patients condition. This feature is also normalized with the diameter of the ROI.
17) The diameter of the optic disc.
18) The binary result of the AM/FM-based classification.
19) Class label. 1 = contains signs of DR (Accumulative label for the Messidor classes 1, 2, 3), 0 = no signs of DR.For this project, I have chosen feature set [] compared to the variations and relevancy of information provided in other features. Class 1, which represents the signs of DR contains 611 row whereas Class 2, contains 536 example.
Class 1 = contains signs of DR = 611 examples out of 1151 instances
Class 2 = contains no signs of DR = 536 examples out of 1151 instances

Implementation

Quadratic classifier: A quadratic classifier is used to separate measurements of two or more classes of objects or events by a quadric surface. It is a parametric way of classification.

$$X^TAX + B^TX + C$$

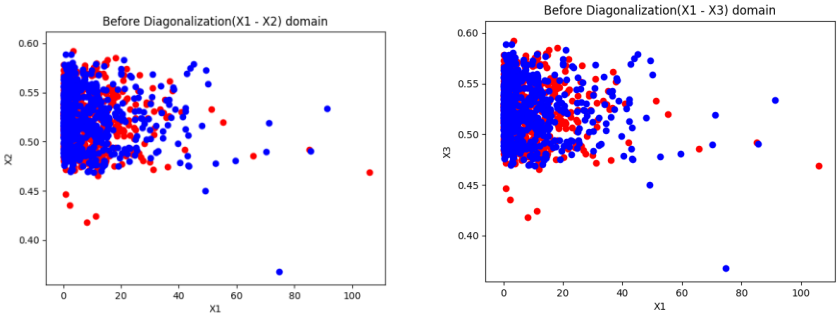
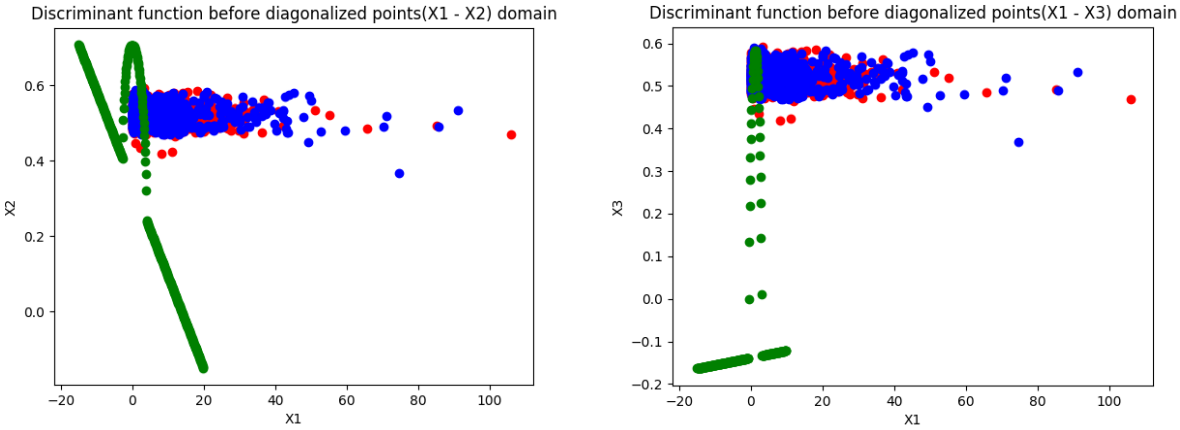


Figure 1: 2 classes before diagonalization



1. Maximum Likelihood(ML) estimation for training –

a) Mean (M) : $1/\mathrm{N}\sum Xi$

where N = Number of total instances and X_i = each point at a particular time

This is estimated(ML) Mean for class1 and class2:
[0.94029851 23.24900456 8.29521293 0.04240483 0.52288005 0.10887204]
[0.898527 23.09773791 9.12073021 0.36271143 0.52343545 0.10791079]

b) Covariance (∑): $1/N\sum[Xi - M][Xi - M]T$

where N = Number of total instances and X_i = each point at a particular time

This is estimated(ML) covariances for class1: [[5.61372243e-02 2.74547957e-01 1.36057947e-01 6.25139341e-04
6.45199098e-05 -4.36837213e-04]
[2.74547957e-01 3.87025306e+02 1.89803157e+02 6.52907482e-01 -8.01677459e-02 -5.34985915e-02]
[1.36057947e-01 1.89803157e+02 1.11743242e+02 4.20550726e-01 -3.79232178e-02 -2.89853768e-02]
[6.25139341e-04 6.52907482e-01 4.20550726e-01 2.45520073e-02 -1.37434024e-05 -1.27554611e-04]
[6.45199098e-05 -8.01677459e-02 -3.79232178e-02 -1.37434024e-05 8.11119479e-04 -8.65847327e-05]
[-4.36837213e-04 -5.34985915e-02 -2.89853768e-02 -1.27554611e-04 -8.65847327e-05 3.00523375e-04]]
This is estimated(ML) covariances for class2:
[[9.11762264e-02 3.52887181e-01 2.20761707e-01 1.61479589e-02 1.73892039e-05 -3.82893764e-04]
[3.52887181e-01 5.35338189e+02 2.64750041e+02 8.70738655e+00 -7.51932256e-02 -1.62282172e-02]
[2.20761707e-01 2.64750041e+02 1.53007379e+02 7.06386577e+00 -4.45393029e-02 -1.16589910e-02]
[1.61479589e-02 8.70738655e+00 7.06386577e+00 2.03356395e+00 -8.48533908e-03 -1.87373272e-03]
[1.73892039e-05 -7.51932256e-02 -4.45393029e-02 -8.48533908e-03 7.64302448e-04 -5.25543356e-05]
[-3.82893764e-04 -1.62282172e-02 -1.16589910e-02 -1.87373272e-03 -5.25543356e-05 3.33335930e-04]]

2. Bayesian Method (BL) estimation for training –

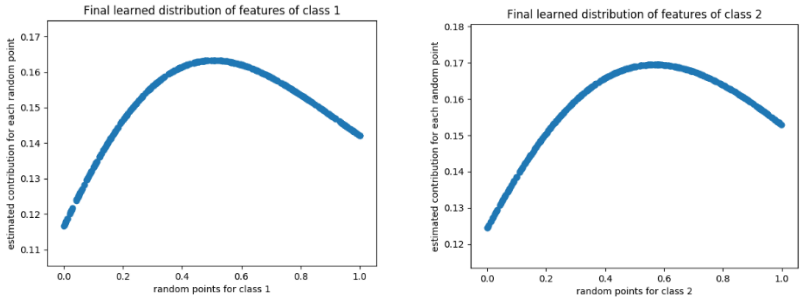
c) Mean (M_n) : $\frac{1}{N}\sum\left[\left(\frac{1}{N}\right)\sum + \sum o\right]^{-1} m_o + \sum o\left[\left(\frac{1}{N}\right)\sum + \sum o\right]^{-1} + \frac{1}{N}\sum Xi$

Where $\sum_o$= Identity matrix and $m_o = 0$

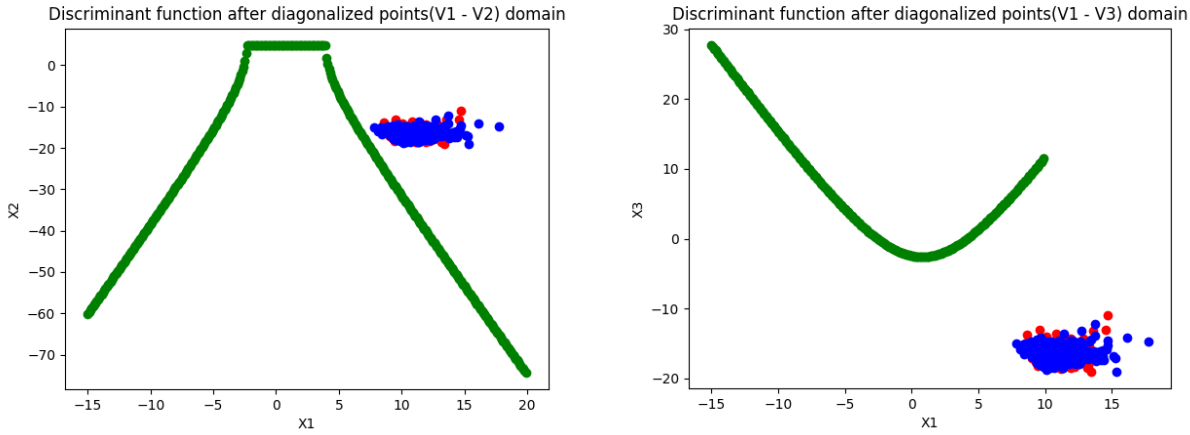
This is estimated(BL) mean for class1 and class2:

[0.93282609 12.86409702 3.09455003 0.02430476 0.52502215 0.11032415]
[0.89053052 11.54982465 3.29008703 0.15953032 0.52509821 0.10828137]

Parzen Window Calculation for one feature of class 1 and 2 :



3. After repeating the above (1-2) for diagonalized points following results are plotted –



4. 5-Fold cross Validation :

For 1151 total datapoints , we used five-fold cross validation. According to the k-fold cross validation rule, for first iteration 921 points is used for training and 230 points are used for testing and it will take next 230 points for validation in each 4 iterations and so on. After testing ,accuracy using estimated parameters are described in following table:

Table: Accuracy table for Quadratic Classifier

Before Diagonalization accuracy using estimated parameters	
Accuracy with estimated ML parameters	51.1769%
Accuracy with estimated BL parameters	52.3331%
After Diagonalization accuracy using estimated parameters	
Accuracy with estimated ML parameters	51.1769%

Accuracy with estimated BL parameters	52.3331%

K-Nearest Neighbour :

The KNN algorithm search through the training dataset for the k-most similar instances.The prediction attribute of the most similar instances is summarized and returned as the prediction of the unseen instance. For the dataset of this project, **1151** instances of **Diabetic Retinopathy** from **2** classes is used and there are **6** features for each observations. I have given input of the whole dataset to the classifier with the class known for each observation. After that the dataset is splitted into training and testing points with a split of **67%** and **33%** for consecutively training and testing.Then for each testing points, the Euclidean distance is measured from that point to every training points and stored the lowest distance point as the neighbour of that testing point. In addition the nearest neighbour value **k =3** is chosen for the project.

Find similarity using Euclidean Distance	Find k-Nearest Neighbour

def euclideanDistance(instance1, instance2, length):
  distance = 0
  for x **in** range(length):
  distance += pow((instance1[x] - instance2[x]), 2)
  return math.sqrt(distance)
	def getNeighbors(trainingSet, testInstance, k):
	  distances = []
	  length = len(testInstance) - 1
	  for x **in** range(len(trainingSet)):
	  dist = euclideanDistance(testInstance, trainingSet[x], length)
	  distances.append((trainingSet[x], dist))
	  distances.sort(key=operator.itemgetter(1))
	  neighbors = []
	  for x **in** range(k):
	  neighbors.append(distances[x][0])
	  return neighbors

1. KNN after Diagonalization

After diagonalization of **1151** dataset, it is given as the input to the classifier and the algorithm similarly calculates the 3-Nearest Neighbour of each diagonalized point in the testing set which is splitted by **67/33** ratio. The 5-fold cross validation result for both diagonalized and non-diagonalized points are discussed in the following section

2. 5-fold cross validation for accuracy measurement of KNN –

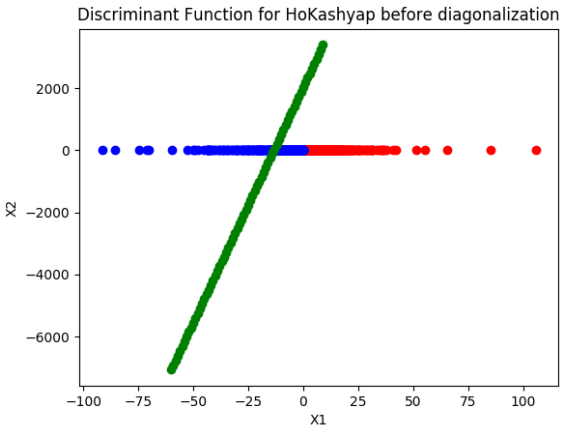
For each observation in validation set, I checked if the class prediction of the testing point is matched with the actual class or not. If matched then, a variable(correct) is increased by **1** and at the end **[correct/length(validation set)] * 100** formula returns the percentage accuracy of the classifier for the Diabetic Retinopathy dataset.

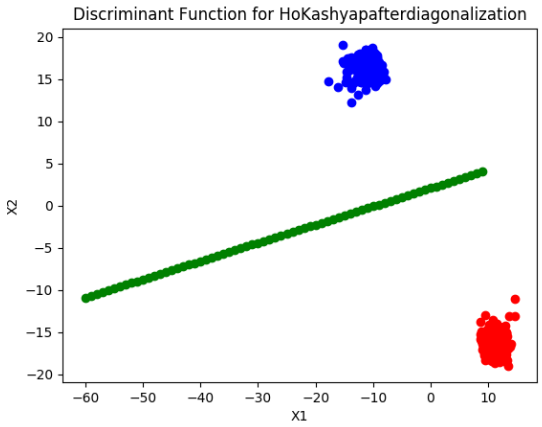
```
def getAccuracy(class_points, predictions):
  for training_x1, validation_x1 in k_fold_cross_validation(class_points,5):
    for x in range(0,len(validation_x1)):
      correct = 0
      if validation_x1[x][-1] == predictions[x]:
        correct += 1
  return (correct / float(len(validation_x1))) * 100.0
```

Before Diagonalization of 1151 instances , accuracy in percentage	
Accuracy	50.56%
After Diagonalization of 1151 instances , accuracy in percentage	
Accuracy	53.159%

Ho-Kashyap :

HoKashyap is used to classify points if they are not Linearly separable unlike perceptron algorithm. For HoKashyap, a separating hyperplane **A** needs to find out for bias **b** and learning rate **n** . In first iteration, a and b is chosen arbitrary**, A = [1,1,…1]** and **b = [1,1,…1,1]** where **b** has one more dimension than **A**. The algorithm continues to diverge until $e^{(k)} >= 0$ where $\mathbf{e^{(k)} = Y. A^{(k)} – b^{(k)}}$





1. 5- fold Cross Validation (HoKashyap):

With the formula, 5 – fold validation is performed for both diagonalized and non diagonalized points

$\mathbf{Y^T . A > 0}$ then **class 1**

$\mathbf{Y^T . A < 0}$ then **class 2**

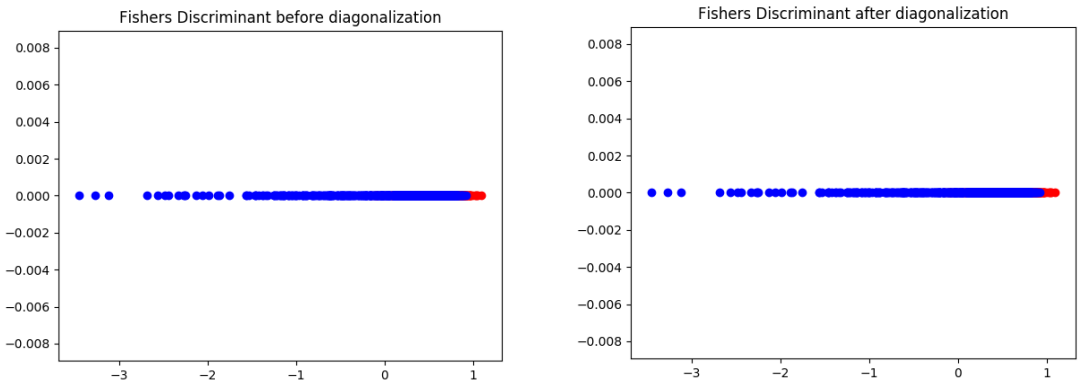
```
def k_fold(class_points,k,a):
  acc = 0
  for training_x1, validation_x1 in k_fold_cross_validation(class_points, k):
     accuracy = 0
     for i in range(0, len(validation_x1)):
        temp = np.transpose(a)
        temp1= np.dot(temp,validation_x1[i])
        if temp1 > 0 and validation_x1[i][0] > 0 :
           accuracy = accuracy +1
     acc = acc + (accuracy/float(len(validation_x1)))*100
  return (acc/k)
```

Before Diagonalization of 1151 instances , accuracy in percentage	
Accuracy	43.94%
After Diagonalization of 1151 instances , accuracy in percentage	
Accuracy	48.69%

Fisher's Discriminant

Fisher's Discriminant is a dimension reduction technique which is used for classification. Formula –

Class within distance, $\mathbf{S_w = \sum_1 + \sum_2}$, weight vector, $\mathbf{W = S_w^{-1} (M_1 - M_2)}$, fisher's point = $\mathbf{W^T X}$



1. 5- fold Cross Validation (Fisher's Discriminant):

After getting the fisher's mean of both classes $M_1 = W^T . M_o$ and $M_2 = W^T . M_o$ [M_o is the estimated mean of class 1 and class 2] , a threshold is generated for 5 – fold validation.

Threshold, $T = (M_1 + M_2) / 2$

```
def k_fold(class_points,k,a, m1,m2):
  acc = 0
  for training_x1, validation_x1 in k_fold_cross_validation(class_points, k):
     accuracy = 0
     for i in range(0, len(validation_x1)):
        T = (m1 + m2)/2
        if temp1 <  T :
           accuracy = accuracy +1
     acc = acc + (accuracy/float(len(validation_x1)))*100
  return (acc/k)
```

Before Diagonalization of 1151 instances , accuracy in percentage	
Accuracy	46.94%
After Diagonalization of 1151 instances , accuracy in percentage	
Accuracy	47.38%

Discussion:

1. Explain how you have tested the quality of your classifier ?
   - ⇨ For each classifier, I have chosen 5-fold cross validation method where the whole dataset is divided into training and testing set five times. With this process, each data points acts as both training and testing points for several times which makes the testing quality of the classifier higher compared to other methods.
2. Comment on accuracy, why it is not good .
   - ⇨ From the figure 1, we can observe that my data points are very close and overlaps with each other. In Addition,the 6 features that I have chosen is also pretty close which makes the

performance of the classification in lower quality. Besides, each classifier follows it's own formula to find 5-fold cross validation. Therefore, variations in the data points could make the performance of classifiers more better.

	A	B	C	D	E	F
1	1	17.775994	5.27092	0.006864	0.486903	0.100025
2	1	23.799994	3.325423	0.003903	0.520908	0.144414
3	1	18.445954	9.118901	0.272434	0.483284	0.11479
4	1	9.497786	1.22366	0	0.576318	0.071071
5	1	50.977459	17.293722	0	0.546008	0.112378
6	0	13.949995	0.436232	0	0.551682	0.139657

Figure 2 : Dataset for class 1

3. Explain how you can extend the system to be a complete statistical PR system
   ⇨ Though the project uses real dataset but the variations in dataset is not very good and in a complete PR system, data whitening is always a mandatory task before performing classification. With this dataset, there was not any need to cleaning the data and the observation is in total 1151 points which is good but in real system there might be more than thousand data. And these dataset might be always updated according to the necessity of the system. Therefore, this system can be extended by looking more deeply into these variations of data points.

References:

1. Antal, Bálint, and András Hajdu. "An ensemble-based system for automatic screening of diabetic retinopathy." *Knowledge-based systems* 60 (2014): 20-27.