**Q1 and 2) Run the following classifiers on all downloaded datasets: a. LDA (linear discriminant analysis) b. Quadratic (quadratic discriminant analysis) c. Naïve Bayes: Choose a distribution and explain why you chose it d. SVM: Choose a kernel and explain why you chose it. For each classifier, provide the following five performance measures: PPV, NPV, Specificity, Sensitivity and Accuracy.**

**A1 and 2)**

Step1: - We download all the datasets provided in the resources which are following circles0.3, moons1, spiral1, twogaussians33, twogaussians42, and halfkernel.

We have used the following algorithms: -

**a. LDA (linear discriminant analysis):-** Used in linear datasets usually and gives higher accuracy for gaussian datasets i.e. **gussian33 and guassian42** as per the given data

**b. Quadratic (quadratic discriminant analysis): -** It does not use assumption of LDA that covariance of classes is same, however when covariance is different QDA works better

**c. Naïve Bayes: -** We select gaussian distribution because it functions well on continuous values and functions well if attributes are conditionally independent. Naïve Bayes is a conditional probability model, given a problem instance to be classified, represented by a vector **x** = (x1, …, xn) representing some n features (independent variables), it assigns to this instance probabilities for each of K possible outcomes or classes.

p(Ck|x1,.....,xn)p(Ck|x1,.....,xn)

**d. SVM: -** It is used to classify both linear and non linear data, SVM uses the concept of classification of tuples into classes so that it can be used to decide the separating hyperplane also it is less prone to overfitting. Thus, we select RBF as kernel (Radial Basis Function) depends on the distance from the origin or from some point and thereby gives curve to separate non linear data hence we choose RBF.

**In order to run the following algorithms**, we need to perform some pre requisite steps

1)Read the csv file in order to import the dataset of the following provided

**CODE: -**

**dataset1 = pd.read\_csv('C:\Python37\datasets\circles0.3.csv')**

**dataset2 = pd.read\_csv('C:\Python37\datasets\halfkernel.csv')**

**dataset3 = pd.read\_csv('C:\Python37\datasets\moons1.csv')**

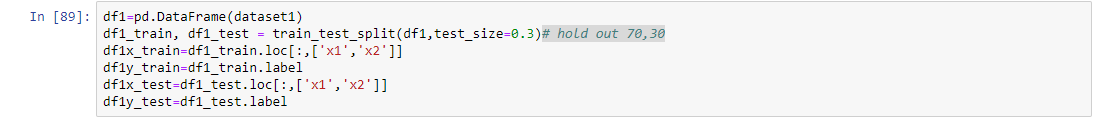
**dataset4 = pd.read\_csv('C:\Python37\datasets\spiral1.csv')**

**dataset5 = pd.read\_csv('C:\Python37\datasets\twogaussians33.csv')**

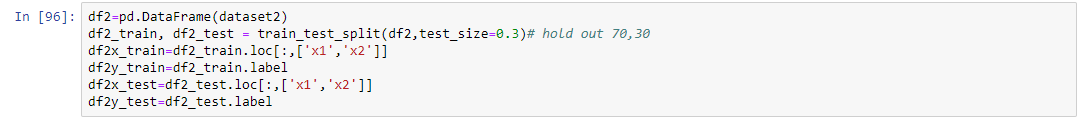
**dataset6 = pd.read\_csv('C:\\Python37\\datasets\\twogaussians42.csv')**

2)We perform the hold out using train and test set in 70 and 30 ratio and determine training and testing data input and output for every model

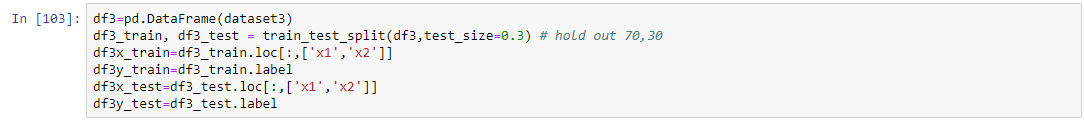
Dataset:- Circles 0.3



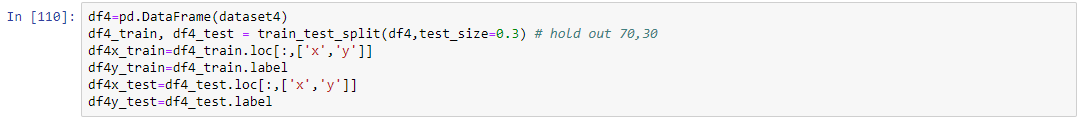
Dataset:-Half Kernel



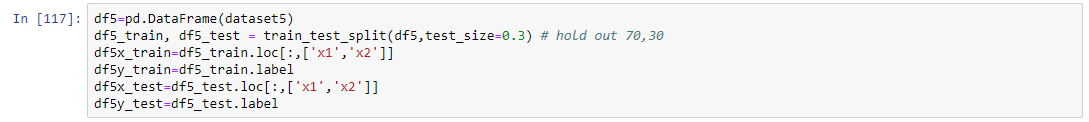
Dataset:-Moons1



Dataset:-Spiral1



Dataset:-Twogaussians33



Dataset:-Twogaussians42



3)As per the question based on the requirement, we plot the scattering graph using different colors and markers

**Screenshots of plots are given below**

4)We thereby specify the type of algorithm which is to be implemented and train and predict the data for output.

5)We specify the metrics which needs to be calculated using confusion matrix and access them to calculated PPV, NPV, Specificity, Sensitivity, Accuracy.

* A Confusion Matrix is a table that is often used to describe the performance of a model which is to be classified or classifier on a set of test data for which the test values are known, we thereby access the elements and determine the elements and find PPV, NPV, Specificity, Sensitivity, and Accuracy

|  |  |
| --- | --- |
| True Positive(TP) | False Negative(FN) |
| False Positive(FP) | True Negative(TN) |

1.Precision or PPV=TP/(TP+FP)

2.Specificity=TN/(TN+FP)

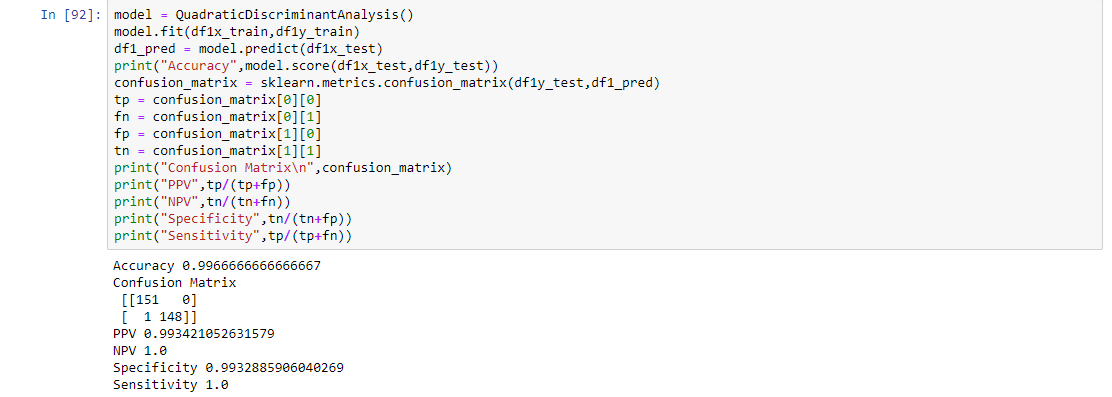
3.NPV=TN/(TN+FN)

4.Accuracy=(TP+TN)/n

5.Senstivity or Recall=TP/(TP+FN)

**Dataset1: - Circles0.3 1)LDA**

**2)QDA**



**3)NAÏVE BAYES**

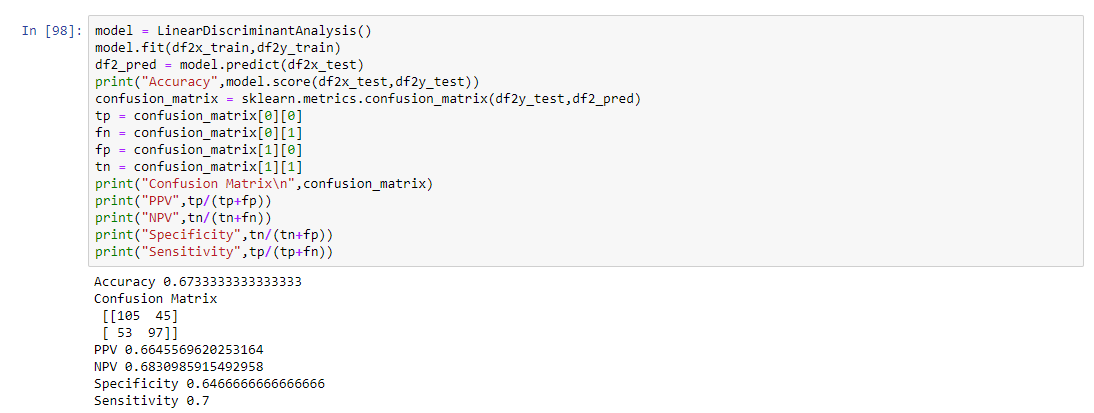


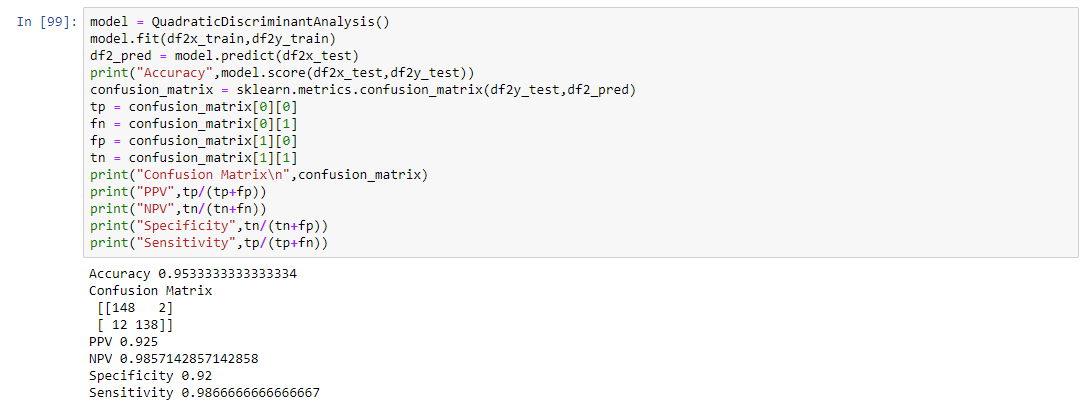
**4)SVM**

****

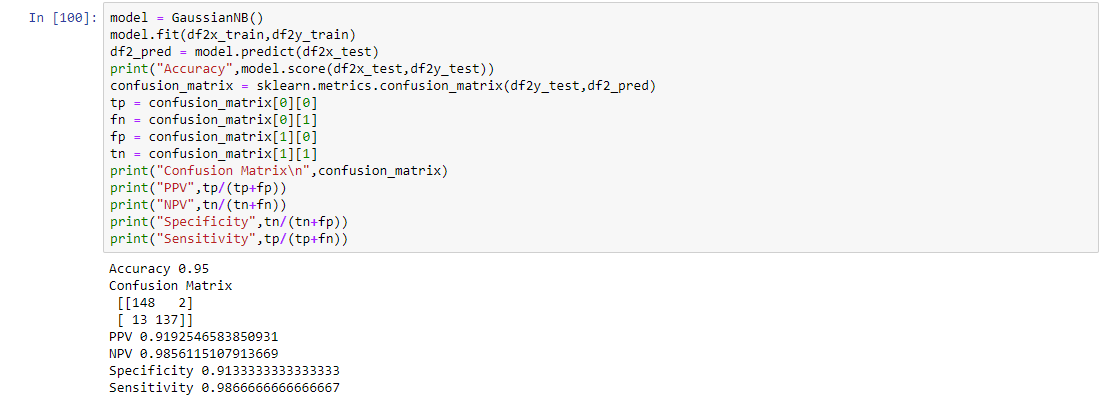
**Dataset2:-Half Kernel**

**1)LDA**

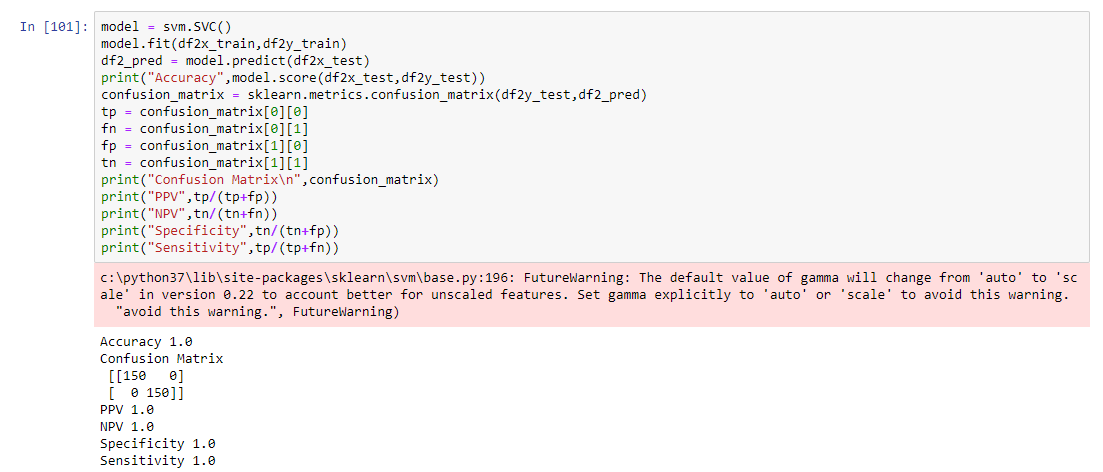


**2)QDA**

**3)NAÏVE BAYES**

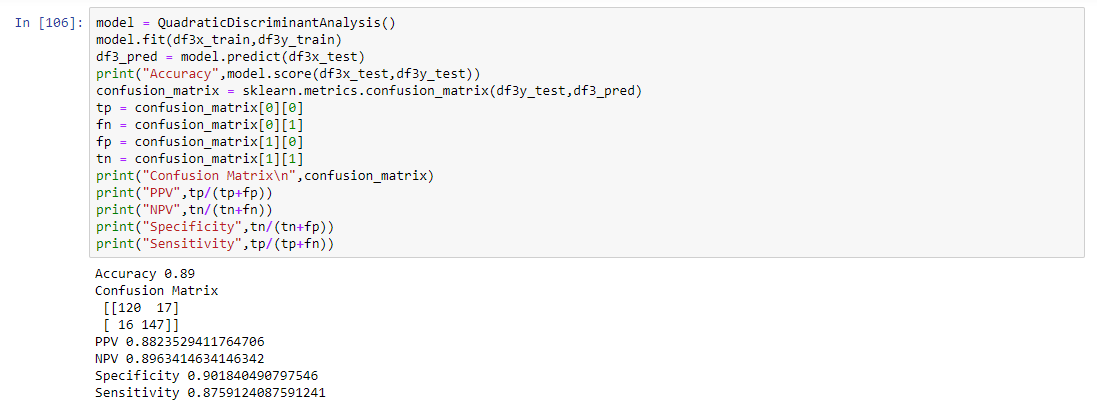


**4)SVM**

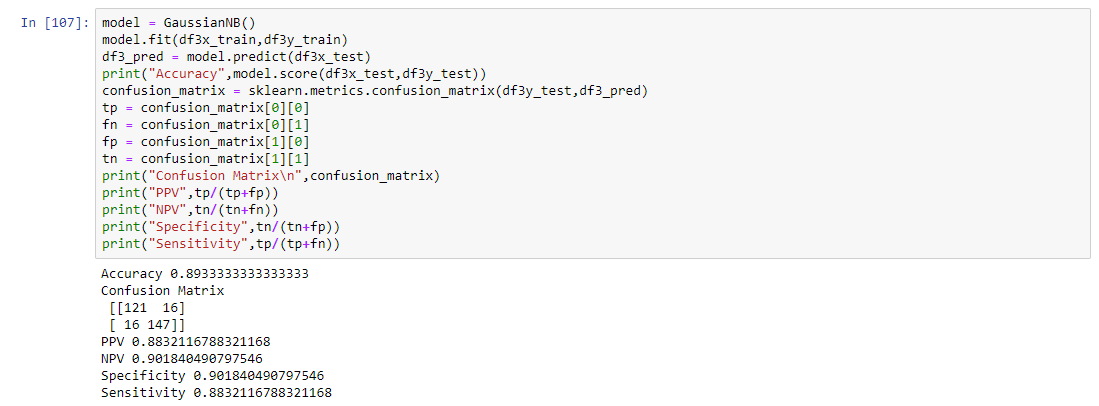


**Dataset3:-Moons1**

**1)LDA**

**2)QDA**

**3)NAÏVE BAYES**

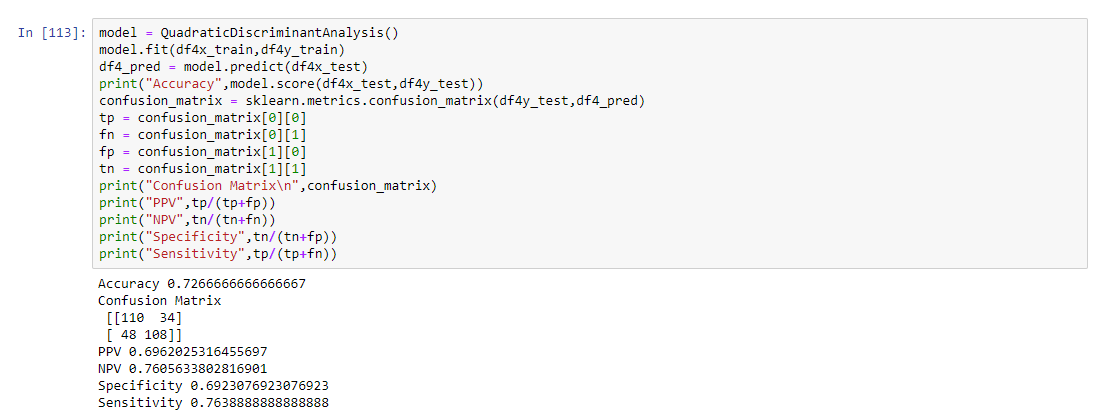


**4)SVM**



**Dataset4:-Spiral1**

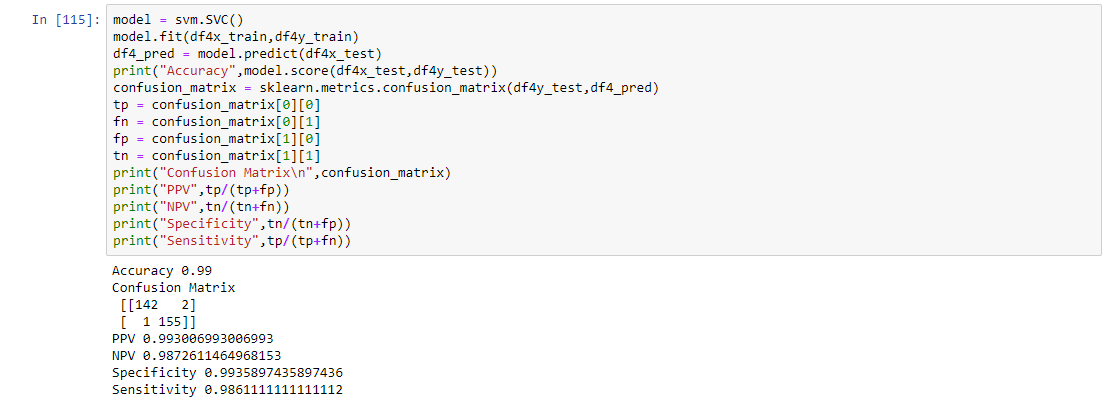
**1)LDA**

**2)QDA**

**3)NAÏVE BAYES**

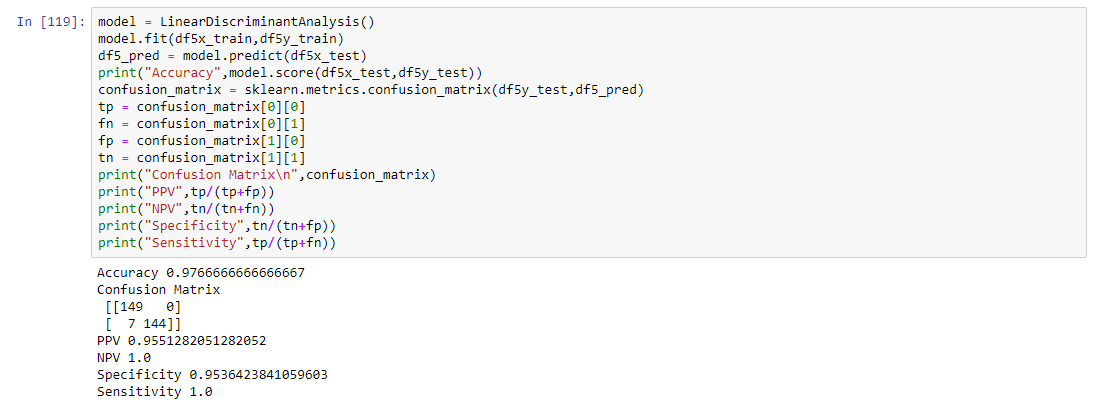


**4)SVM**



**Dataset5:-Twogaussians33**

**1)LDA**



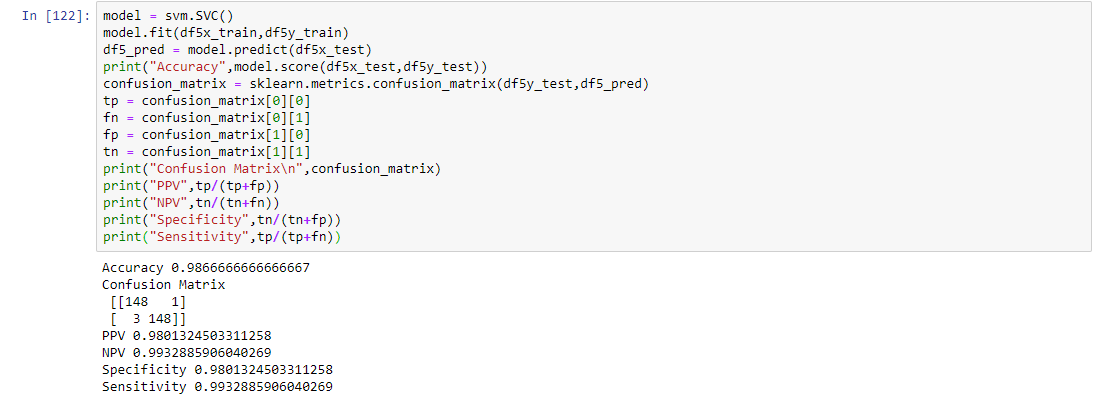
**2)QDA**



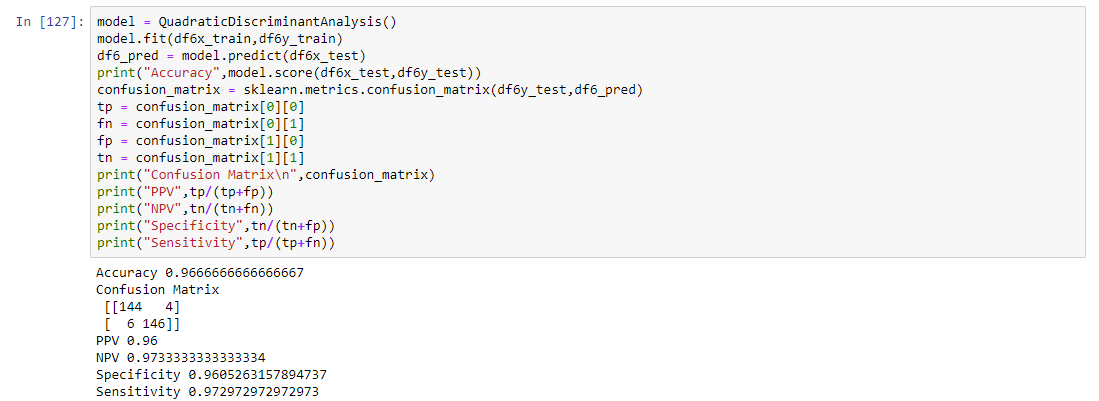
**3)NAÏVE BAYES**



**4)SVM**



**Dataset 6:-Twogaussians42**

**1)LDA****2)QDA**

**3)NAÏVE BAYES**



**4)SVM**

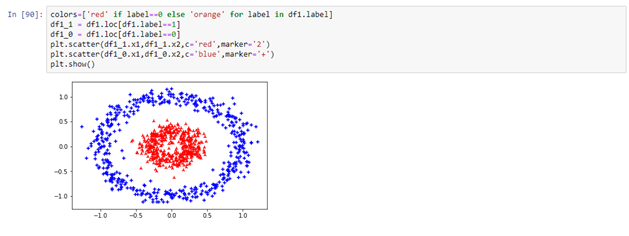


**Q2. Plot the samples of the all datasets (in separate plots) as points in the 2D space, using a different color and point shape for each class.**

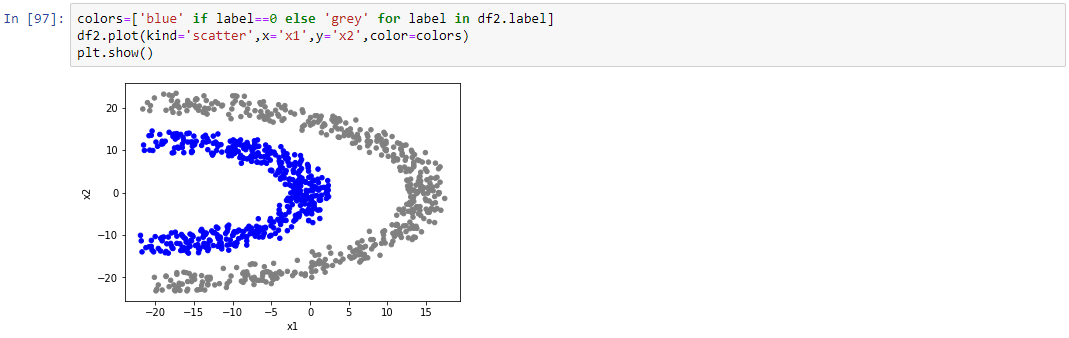
**A2)** To plot the samples of all data sets in 2D space we make use of pandas and matplotlib in order to draw the scatter graph of different colors we use different **markers** and **labels** to show the colors

Thus, after applying matplotlib we make the scatter on the system as following for different datasets

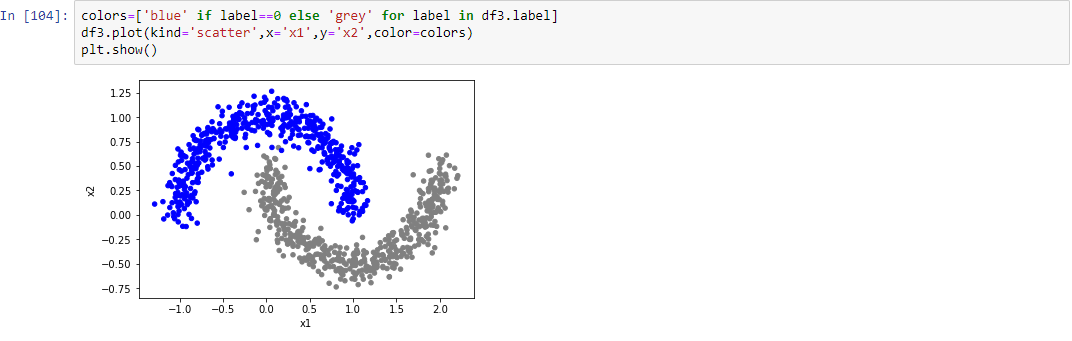
Dataset: -Circles0.3



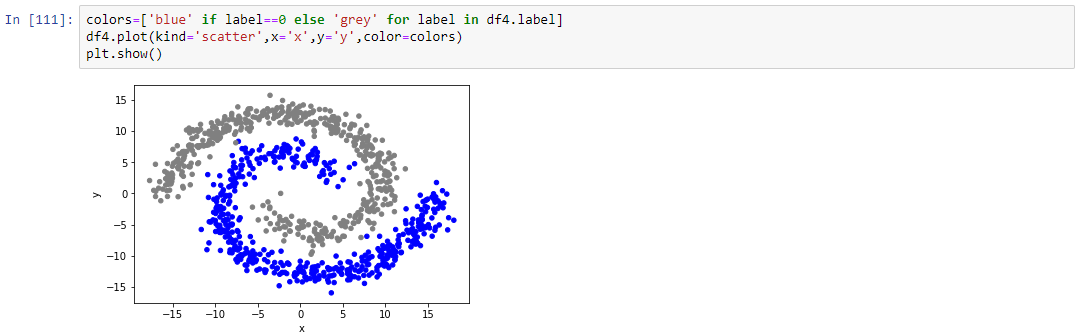
Dataset: -Half Kernel



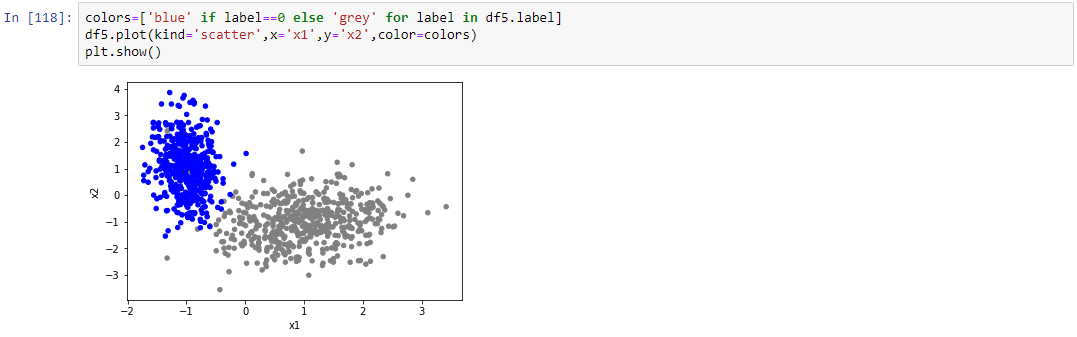
Dataset: -Moons1



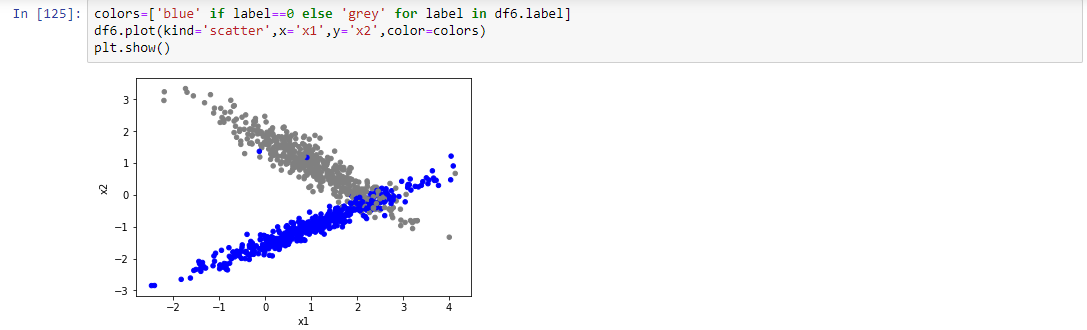
Dataset: -Spiral1



Dataset: -Twogaussian33



Dataset: -Twogaussians42



**Q3. Briefly discuss the performance (accuracy only) for each classifier and dataset individually: Why do you think it is good/poor?**

**A3)**

**Dataset1: -Circles0.3**

* When we apply **LDA** on circles0.3 dataset it gives an accuracy of 0.48
* When we apply **QDA** on circles0.3 dataset it gives an accuracy of 0.99
* When we apply **NAÏVE BAYES** on circles0.3 dataset it gives an accuracy of 0.99
* When we apply **SVM** on circles0.3 dataset it gives an accuracy of 1

**The above dataset is visualized as concentric circles therefore LDA performs poorly as it is used to classify linear datasets and also since variance values are different naïve bayes behaves like QDA which gives similar accuracy and on performing SVM thereby gives the best accuracy of 1 which stands the best in this case.**

**Dataset2: -Half Kernel**

* When we apply **LDA** on half kernel dataset it gives an accuracy of 0.67
* When we apply **QDA** on half kernel dataset it gives an accuracy of 0.95
* When we apply **NAÏVE BAYES** on half kernel dataset it gives an accuracy of 0.95
* When we apply **SVM** on half kernel dataset it gives an accuracy of 1

**The above dataset is visualized as two concentric semi circles and therefore LDA performs poorly as it is used to classify linear datasets and also since variance values are different naïve bayes behaves like QDA which gives similar accuracy and on performing SVM thereby gives the best accuracy of 1 which stands the best in this case as it takes similarity of one datapoint with rest of the points in dataset. So the new dataset is m x m where m is the number of data in the original dataset.**

**Dataset3: -Moons1**

* When we apply **LDA** on moons1 dataset it gives an accuracy of 0.89
* When we apply **QDA** on moons1 dataset it gives an accuracy of 0.89
* When we apply **NAÏVE BAYES** on moons1 dataset it gives an accuracy of 0.89
* When we apply **SVM** on moons1 dataset it gives an accuracy of 0.99

**The above dataset is visualized as two semi circles opposite to each other and therefore LDA performs poorly as it is used to classify linear datasets and also since variance values for the dataset is same thereby gives the similar accuracy for all three LDA, QDA and Naïve Bayes i.e: - they perform and gives a similar accuracy of 0.89 also SVM gives the best result in the above case 0.99 where it compares m x m number of points in the original dataset.**

**Dataset4: -Spiral1**

* When we apply **LDA** on spiral1 dataset it gives an accuracy of 0.72
* When we apply **QDA** on spiral1 dataset it gives an accuracy of 0.72
* When we apply **NAÏVE BAYES** on spiral1 dataset it gives an accuracy of 0.71
* When we apply **SVM** on spiral1 dataset it gives an accuracy of 0.99

**The above dataset is visualized as spiral design, LDA performs poorly as it is used to classify linear datasets and also since variance values for the dataset is same thereby gives the similar accuracy for all three LDA, QDA and Naïve Bayes i.e: - they perform and gives a similar accuracy of 0.72 also SVM gives the best result in the above case 0.99 where it compares m x m number of points in the original dataset.**

**Dataset5: -Twogaussians33**

* When we apply **LDA** on twogaussians33 dataset it gives an accuracy of 0.97
* When we apply **QDA** on twogaussians33 dataset it gives an accuracy of 0.98
* When we apply **NAÏVE BAYES** on twogaussians33 dataset it gives an accuracy of 0.98
* When we apply **SVM** on twogaussians33 dataset it gives an accuracy of 0.99

In the above dataset LDA gives good performance with accuracy 0.97 since it is linearly separable also QDA gives a good performance since it gives a curved boundary and helps in better classification also naïve bayes makes use of gaussian distribution and hence on continuous data it gives best performance also SVM makes use of m x m comparison for independent points on the dataset for the above system.

**Dataset6: -Twogaussians42**

* When we apply **LDA** on twogaussians42 dataset it gives an accuracy of 0.92
* When we apply **QDA** on twogaussians42 dataset it gives an accuracy of 0.966
* When we apply **NAÏVE BAYES** on twogaussians42 dataset it gives an accuracy of 0.92
* When we apply **SVM** on twogaussians42dataset it gives an accuracy of 0.96

In the above dataset LDA gives good performance with accuracy 0.92 since it is linearly separable also QDA gives a good performance since it gives a curved boundary and helps in better classification also naïve bayes makes use of gaussian distribution and hence on continuous data it gives best performance also SVM makes use of m x m comparison for independent points on the dataset for the above system.

**Accuracy Table: -**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Accuracy | Circles0.3 | HalfKernel | Moons1 | Spiral1 | Twogaussins33 | Twogaussians42 |
| **LDA** | 0.48 | 0.67 | 0.89 | 0.72 | 0.97 | 0.92 |
| **QDA** | 0.99 | 0.95 | 0.89 | 0.72 | 0.98 | 0.96 |
| **NaïveBayes** | 0.99 | 0.95 | 0.89 | 0.71 | 0.98 | 0.92 |
| **SVM** | 1.00 | 1.00 | 0.99 | 0.99 | 0.99 | 0.96 |

The following table thereby summarizes the accuracy values of all the algorithms the following result which can be interpreted is that SVM serves the best role in classification for all the dataset in general as it is flexible with all the types of dataset may it be linear or non linear

LDA serves well only for the linear data and thus provides **good** result only for twogaussians33 and twogaussians42

QDA serves well for non linear data and also provides quadratic decision boundary for classification which can be used of better results for varying covariances using curved boundaries.

Naïve Bayes serves well for conditionally independent and also continuous type of data because of the class independence assumption, naive Bayes classifiers can quickly learn to use high dimensional features with limited training data compared to more sophisticated methods. This can be useful in situations where the dataset is small compared to the number of features, such as images or texts.

SVM serves the best result for the above datasets in all cases as it handles both linear and nonlinear kind of data also it takes similarity of one data point and compares it with other points in the dataset which is equivalent of m x m points in the original dataset and almost provides 100% accuracy in each plots above and gives the best result.

**Q4. For each dataset: which classifier is the best for that particular dataset, why? Give very strong reasons, including mathematical formulas and/or arguments.**

For the above system SVM serves the best classifier for all datasets

In SVM algorithm we find the points closest to the line from both the classes which are called as support vectors. Now, we compute the distance between the line and the support vectors this distance is called the margin. The goal is to maximize the margin. The hyperplane for which the margin is maximum is the optimal hyperplane. Hence it makes sure that two classes are separated from each other for the best result.

Also in complex cases where it is not linearly separable we convert the data into higher dimension and add a third dimension from 2D to 3D format which mathematically can be given as

z = x²+y²

Tuning of parameters is one of the most important features in SVM which involves

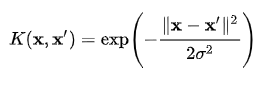
1. C
2. Gamma
3. Degree

C: - Refers to the cost, suppose if SVM cannot separate the data it thereby gives flexibility to produce some mistakes and adjust as per this through the algorithm, more the c value means more training points are correct.

Gamma: - It is solely responsible for the shape of the kernel, more the value of gamma more they are dependent on points which are close to line.

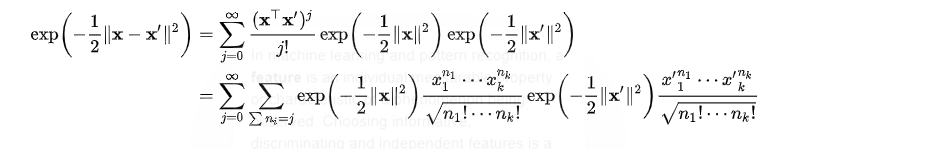
Degree: - This parameter is usually absent in case of RBF kernel because they have infinite values.

SVM usually has greater tuning factor with different parameters when it comes to RBF, thus following is the mathematical formula for RBF



Referred from <https://en.wikipedia.org/wiki/Radial_basis_function_kernel>

Assume that σ=1, RBF where thereby will have infinite dimensional representation



Hence x and x’ represent feature vectors in input space and the square of Euclidean distance is calculated where sigma is a free parameter which determine the scatter and geometrical shape of the kernel, also RBF provides the infinite space functioning it has better performance feature as compared to other working models.

Other classifiers: -

**LDA: -** It is linear classifier which is used to separate only linear data and hence fails to perform in dataset which are not linearly separable but continues to give a good accuracy in twogaussian33 and twoguassian42 as they are linearly separable. **QDA:** Quadratic Discriminant Analysis which helps in forming a curved boundary i.e. quadratic mostly for varying covariances. **Naïve Bayes: -** Naive Bayes classifiers can quickly learn to use high dimensional features with limited training data compared to more sophisticated methods. This can be useful in situations where the dataset is small compared to the number of features, such as images or texts. Also using gaussian formula when the dataset is scattered it tends to give less accuracy.

**Q5-. Explain mathematically (sketch of proof) how Naïve Bayes that uses Gaussian distributions is a simplified form of the quadratic (optimal Bayesian) classifier.**

**A5)** Bayesian Classification for training data “D” probability of hypothesis ‘h’ the theorem can be written as: -

P(h/D) = (P(D/h).P(h))/P(D)

Where

P(h) = The probability of hypothesis ‘h’

P(D) = The probability of the dataset ‘D’

P(h/D) = The conditional probability of h given will be D

P(D/h) = The conditional probability of D will be h.

Let us take into consideration Multivariate Gaussian Distribution:-

Mathematically it can be represented as

**f(x)= 1\*{exp(-1/2\*(x-μ)T∑-1(x-μ))}/(√[2π]d x det [∑])**

where:-

**μ** is the mean.

**∑** is the covariance matrix

**D** is the dimension of x.

If the covariance for the output classes is not the same, then

p(x|t = 1) = p(x|t = 0) (\*Using, p(x|t) = 1\*exp(−1/2 (xj − µjt)T Σ t−1 (xk − µkt)) **/**√(2π)D det(Σt))

=>log π1 − 1/2 (x − µ1) T Σ1-1 (x − µ1) = log π0 − 1/2 (x − µ0) T Σ0-1 (x − µ0)

On Labelling the constant terms as a and solving further, we get,

=>xT( Σ1−1 − Σ0-1) x −2 (µ1T Σ1-1 − µ0T Σ0-1 ) x + (µ0T Σ0µ0 − µ1TΣ1µ1 ) = a

Assume,

P=( Σ1−1 − Σ0-1)

vT= (µ1T Σ1-1 − µ0T Σ0-1 )

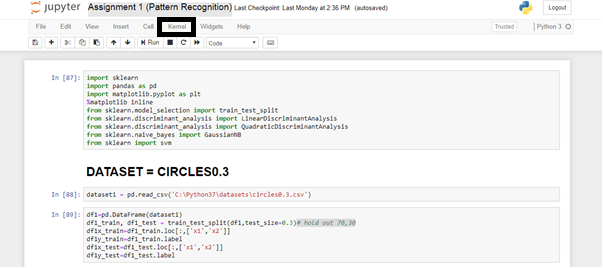
We see that the below equation is a quadratic equation.

xTPx – 2vT x + c = 0

Covariance are different, the above becomes a quadratic equation.

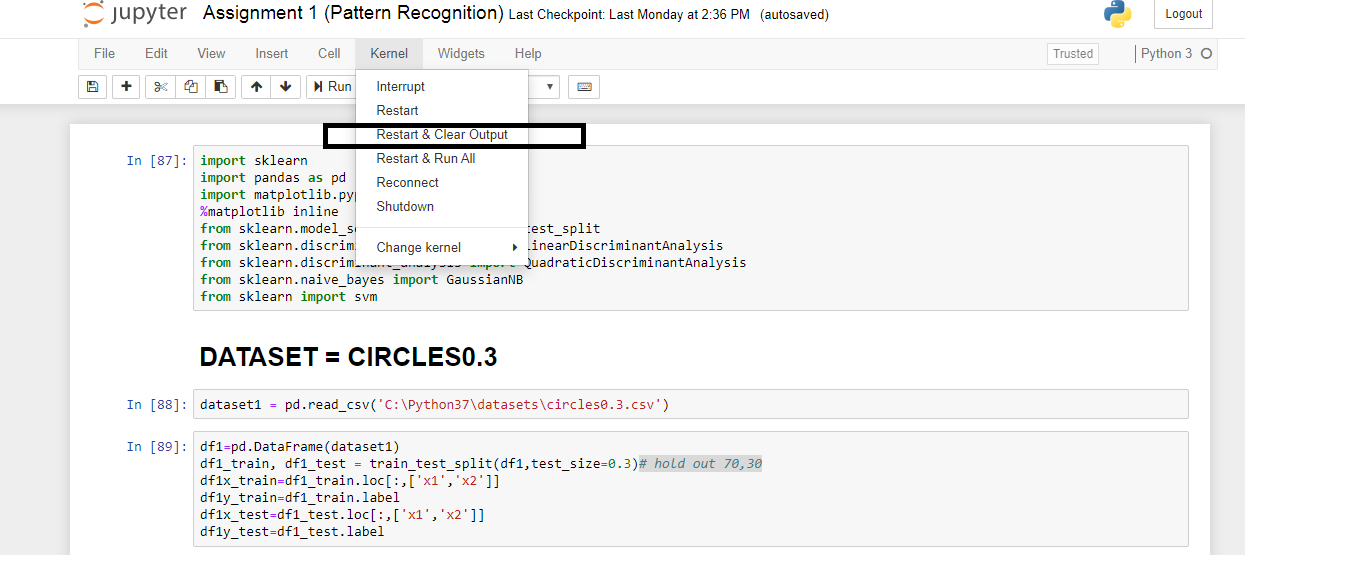
**Steps to run the code: -**

* **Open python jupyter book in terminal of your folder in which there is python program**
* **Create a new file of assignment 1 and follow the code**
* **Each cell has different functions either run the cells by using shift + enter or run entire code by selecting the kernel option**

****

**The black box shows the following option**

* **Select restart and run all option from the following**

****

**The black box shows the following option**

* **Hence each cell runs showing the accuracy along with the following results as needed**