Introduction

Machine learning is a method of data analysis that automates analytical model building. So I am going to perform some machine learning tasks for better understanding. To perform Data science I need data points that I can analyze and fit into models. So for that, I will generate some random points and with the help of these random points, I will also generate target data which will be the y variable. I am going to do this task in Python and python works on libraries so I also have to import some libraries.

These Libraries i will Use

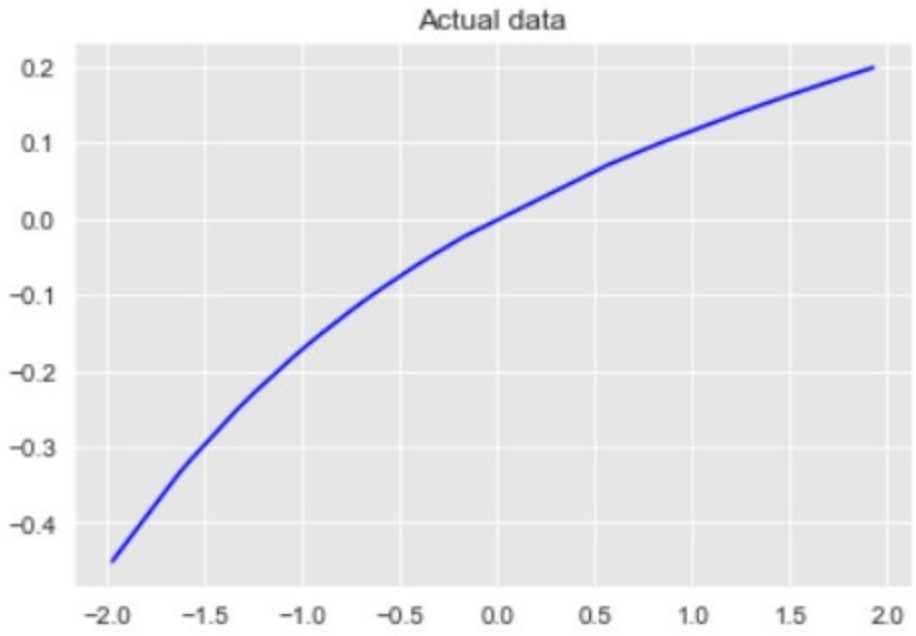
1) import NumPy as np : This library helps to perform mathematical operations with arrays and matrix.
2) import pandas as pd: This library helps to build a data frame.
3) from sklearn import datasets: This will help to import datasets like the iris dataset.
4) import matplotlib.pyplot as plt : This will help to plot visuals.
5) import seaborn as sns : This will help to plot advanced visuals.
6) from scipy.stats import norm : This will help to generate random points with normal distribution.
7) from numpy import dot : This will help to take the dot product of any matrix.
8) from sklearn.model_selection import train_test_split : This will help to split the dataset.

Task:1 Linear Regression with non-linear functions.

So for performing Linear Regression, I need some random points, SO first I will generate random 30 points between -2 to 2, and with the help of these points will make a target column and this function will help to get the target column.

$$f(x) = e^{-x/2}\sin(\pi x) + \sin\left(\frac{3\pi x}{2}\right).$$

This the Actual data it contins 30 points between -2 to 2,So I will use this dataset into Linear Regression Model



Gradien_Descent:

linear regression is a regression model that estimates the relationship between one independent variable and one dependent variable using a straight line. So to get the best regression line I have to find optimal weights and Baise and put them into the slope formula. in this task, I will also generate the Error of prediction with the help of the loss function.

Statistics: I will start my Regression with 0 weight and baise and then calculate the rate of change of Weight and baise and then predict the outcomes. I am going to build the Gradient Descent with the help of these equations.

1. Prediction = $Weight * Points + Baise$

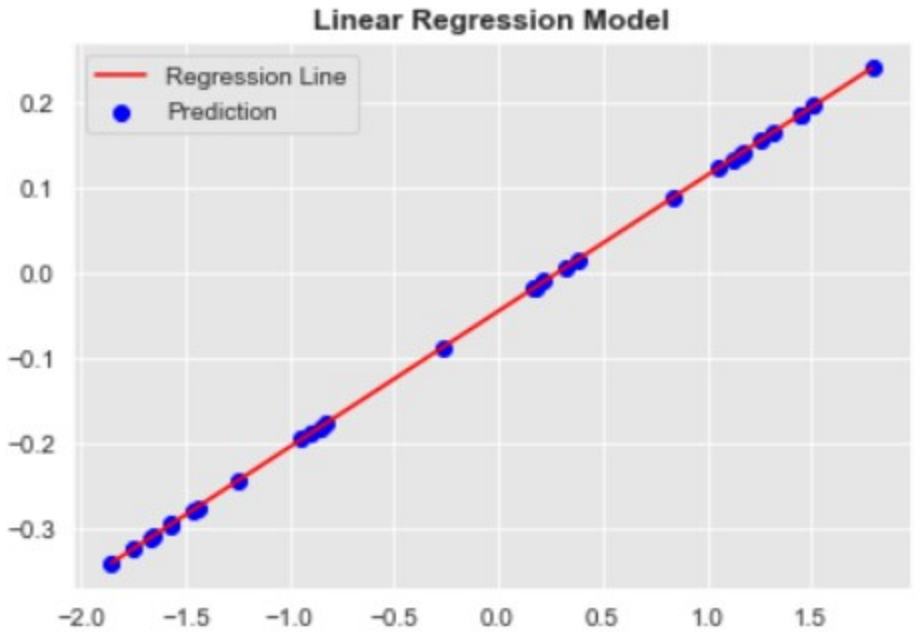
2. Loss = $(1/N) * \Sigma^{N} (y - predicted)^{2}$

3. Change in Weights =
   $(-2/N) * \Sigma^{N} Points * (y - predicted)^{2}$

4. Change in Baise =
   $(-2/N) * \Sigma^{N} (y - predicted)^{2}$

As you can see the Loss of Gradient descent is decreasing with the increase in number of iteration.We can also try different learning rates to get better results

As you can see the Regression line is fitted very well,but we can futher optimize the model by trying different parameters.



Task:2 gaussian radial basis function (RBF):

The radial distribution function (rdf) defines the probability of finding a particle at distance r from another tagged particle.So for Radial distribution I need a r distace from the center point.

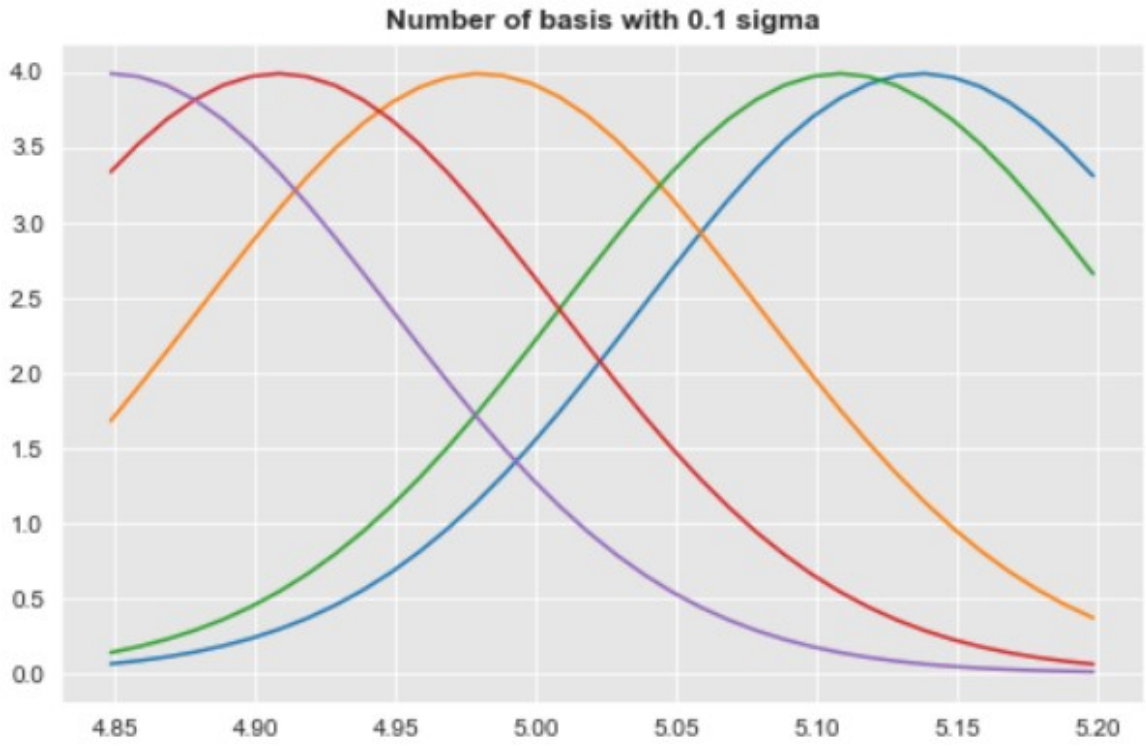
Statistics: First i will generate some random points for distribution and define the number of basis and sigmoid and then calculate the the r distance and then put them into the RBF equation which will distribute the data and i will calculate the weights by using these equation.

1. RBF = $exp((-\ 1/2)\ *\ ((x\ -\ xj)/sigma)^2)$

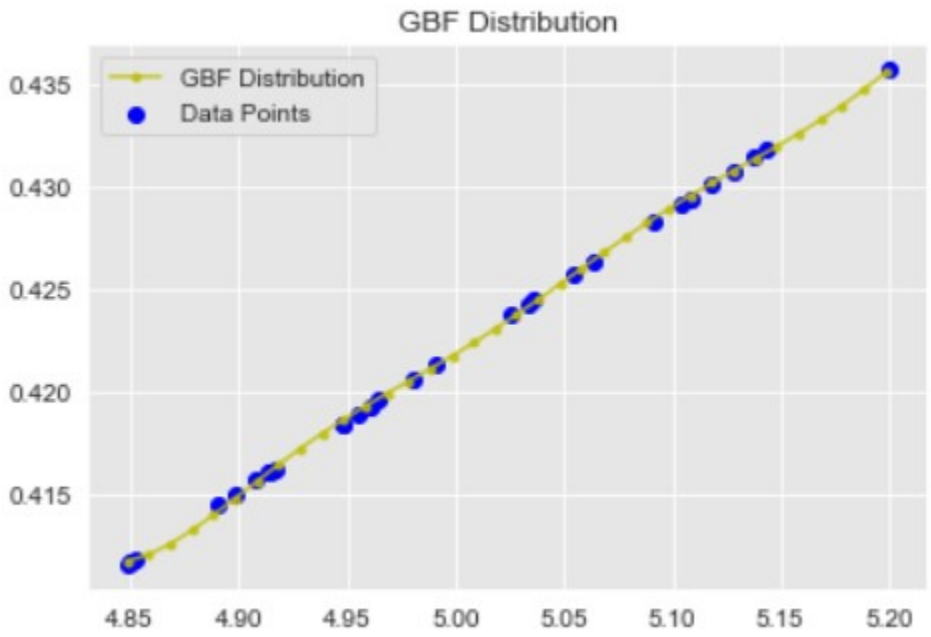
2. Weight = $(x^t\ *\ x)^{-1} sigma^2$

As you can see there are 5 basis.We can try different number of

basis and sigma value for better results



As you can see GBF is doing great. i have set the sigma = 0.1 and choose 5 number of basis.So can try different sigma and basis but i think this the optimal value



Task:3 Classification by Fisher LDA

In this code, I build a Fisher LDA model to create a boundary between classes and projected them on the same plane by maximizing their mean difference and minimizing their variances. In this code, I perform the following tasks:

1) I take the mean of their classes individually.

2) I take the overall mean of class.

3) I calculate the mean sum within classes by subtracting the individual mean from the overall mean and taking a dot product with their Transpose. And sum all values.

4) Then I calculate the mean sum of overall classes by taking a difference of each class mean and overall classes and then take a dot product by their transpose and after that, I multiply it with their number of classes and then I sum up all the values.

5) In this step I take the inverse of sum within classes and take a dot product with the sum of the overall mean.

6) In this step i create the eigenvectors to project the classes on pi plane. and then i transform the data for projection.

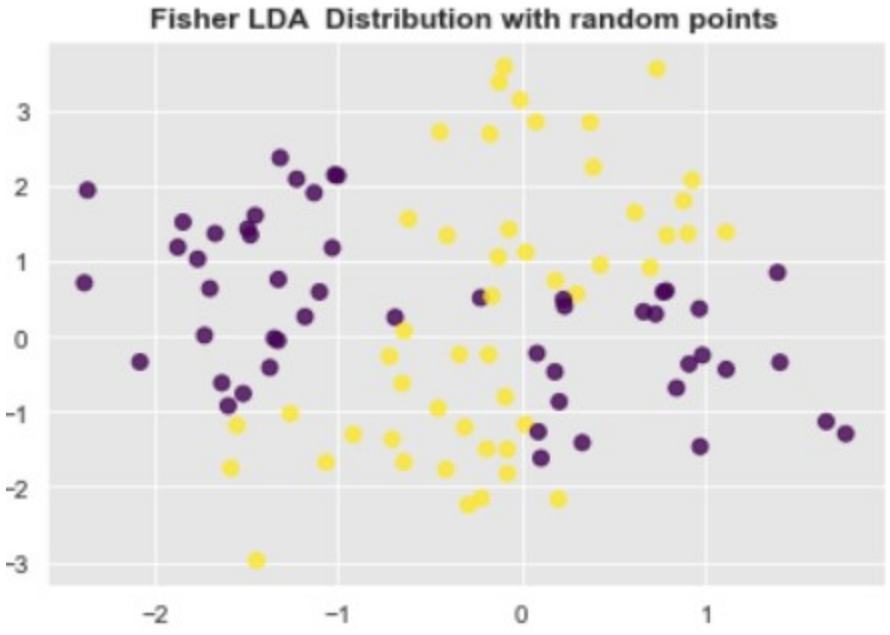
$$\tilde{s}_i^2 = \sum_{y \in \omega_i} (y - \tilde{\mu}_i)^2$$

$$J(w) = \frac{|\tilde{\mu}_1 - \tilde{\mu}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

Fisher LDA with Random Points:

Now i am going to generete some randomm points and also generate the target column with the help of sigmoid function. i want a target column in binary form and sigmoid converts all values in ones and zeros form. and then i will use the Fisher LDA function on the X and y feature

As you can see Fisher LDA separate the classes , the yellow points are belongs to 0 class and purple points belongs to 1 class

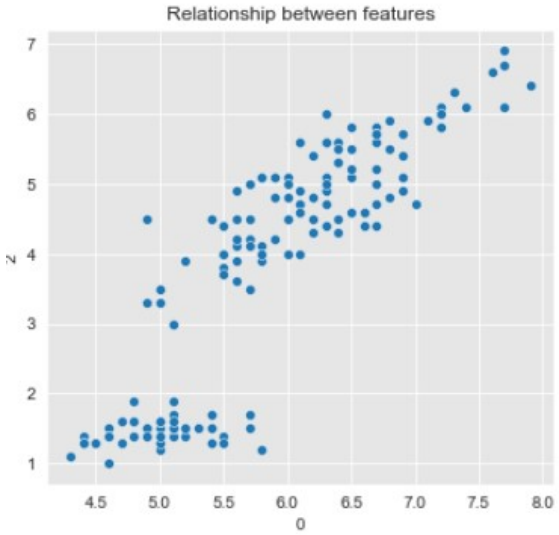


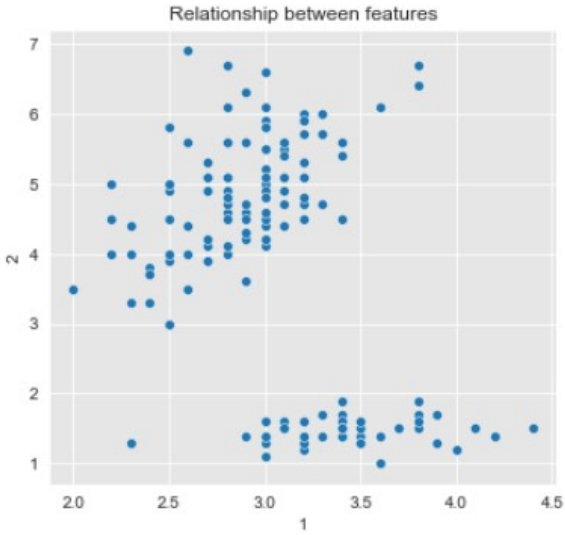
Fisher LDA with Iris dataset

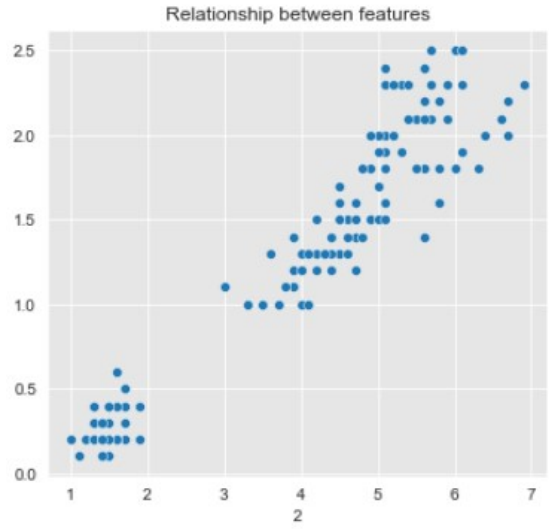
I have a dataset of Iris flowers. Irises are either bulbous or rhizomatous (with thick creeping underground stems). In species with a rhizome, the stem is usually horizontal, robust, and ringed with leaf scars. It often grows partially exposed but is firmly rooted in the soil. The dataset got 5 features and 4 features contain information about petal length and width and sepal length and width. The 5th column is a target column (class) which describes to which class the specific iris flower belongs by the length and width of its petal and sepal.

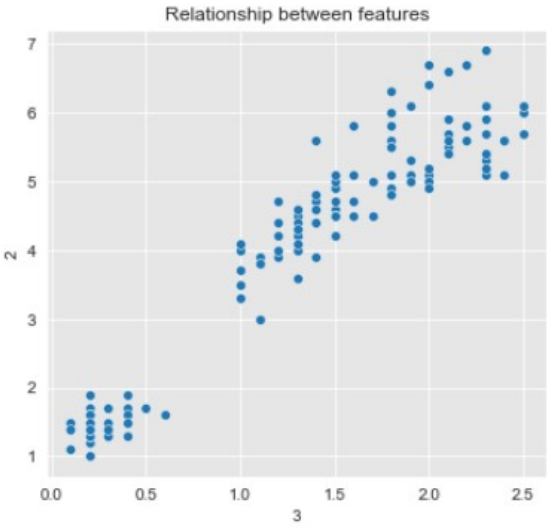
Many columns are strongly correlated with each other, Which is a good sign for model training. petal width petal length is highly correlated with the target column So these are the most important features for better results.

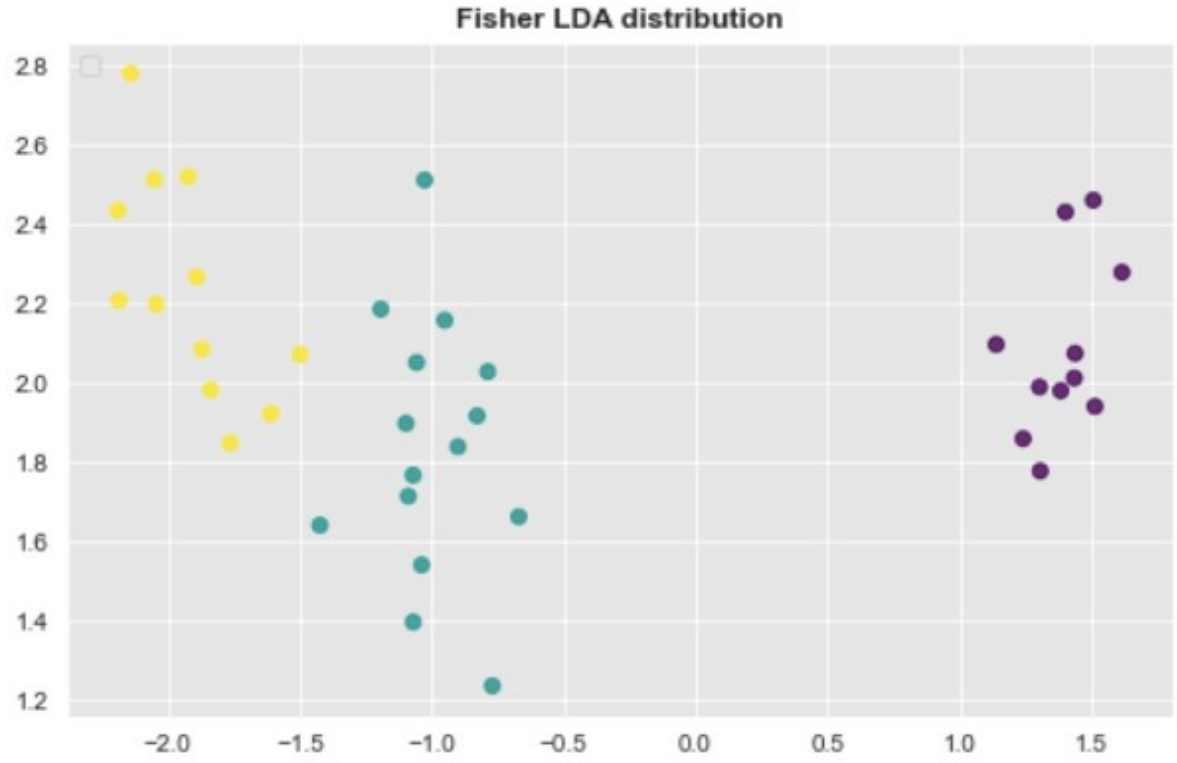
As you can see that every feature have a linear Relation with another Feature, So this means that every feature depends upon other feature there is no independent feature. Which is a good sign for model training.





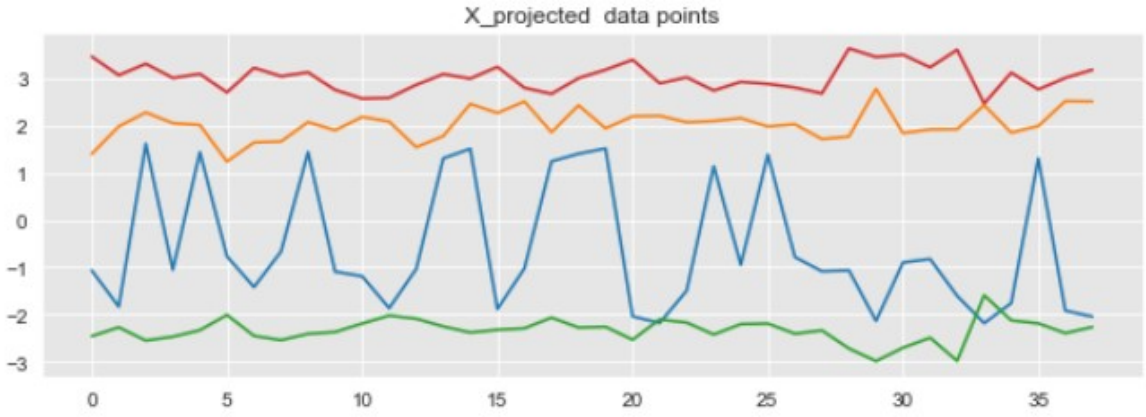




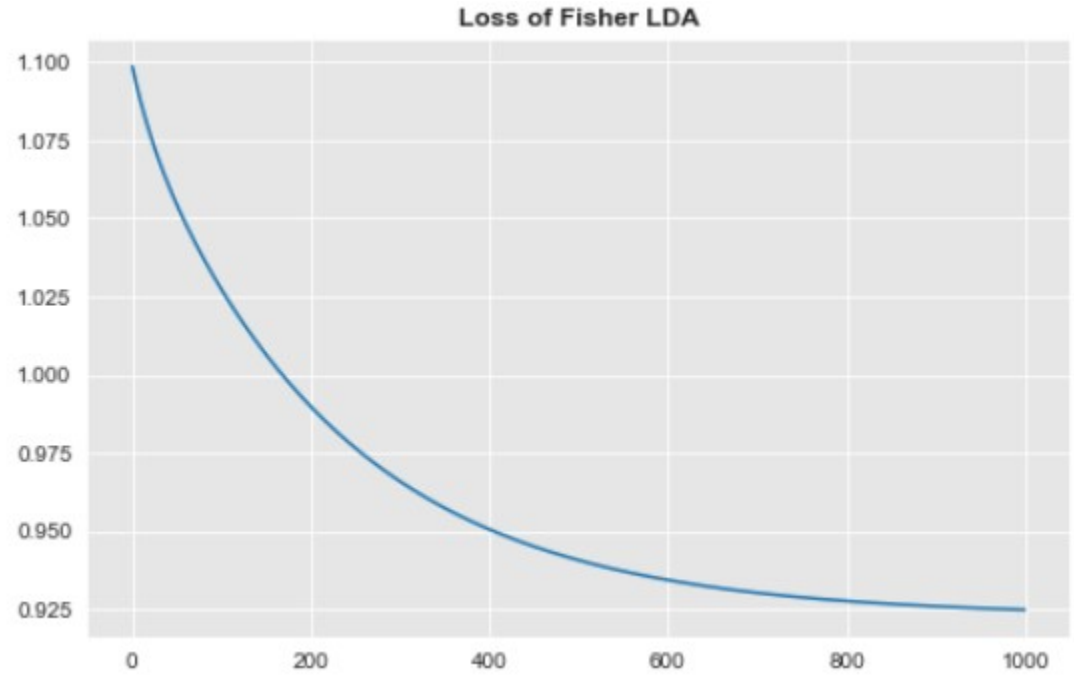


Comparison Analysis of X Projected VS Test Points.

Fisher LDA separate the classes very well , the first plot shows the fisher LDA distribution and The second plot shows the orignal ponts



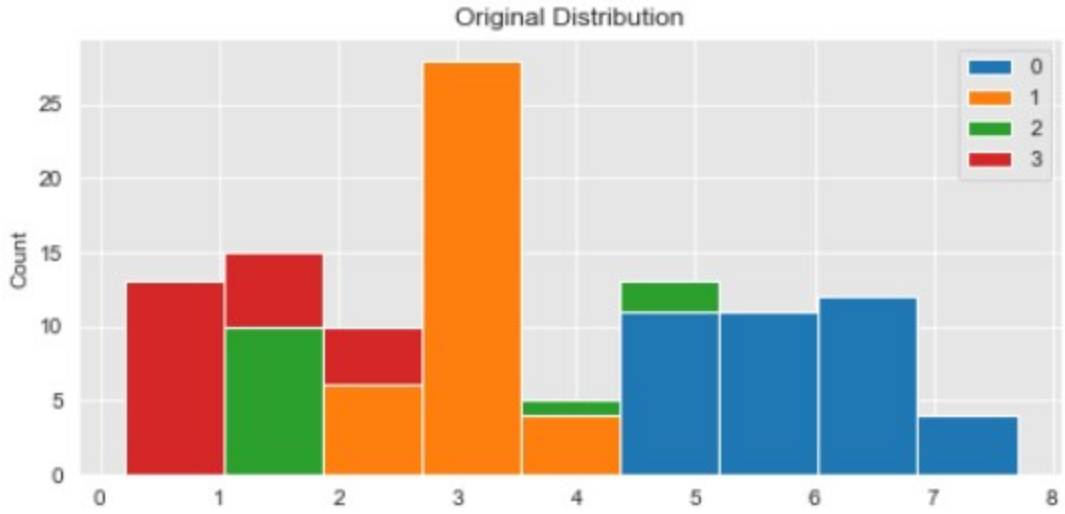
The Loss function is Decreasing with the increase in number of iterations.



Comparison Analsis of LDA with Histogram:

The Original points was very close together and it was not easy to distinguish between them but after Distribution now you can see

all the features are properly distributed now we can use this dataset for prediction purposes



Conclusion:

- LinearRegression Line is fitted very well and the loss is almost equal to zero.
- I have Created 5 Number of basis for RBF and it distribute the values very well.

- Fisher LDA worked much better on Irish dataset rather than on random points.
- The model Accuracy is 68%.
  Eigenvector are:
  1) Eigenvector 0: 0.9918956340892251
  2) Eigenvector 1: 0.008104365910775194
  3) Eigenvector 2: 2.5023944092295735e-16
  4) Eigenvector 3: 2.8656636567445944e-17

In [ ]: