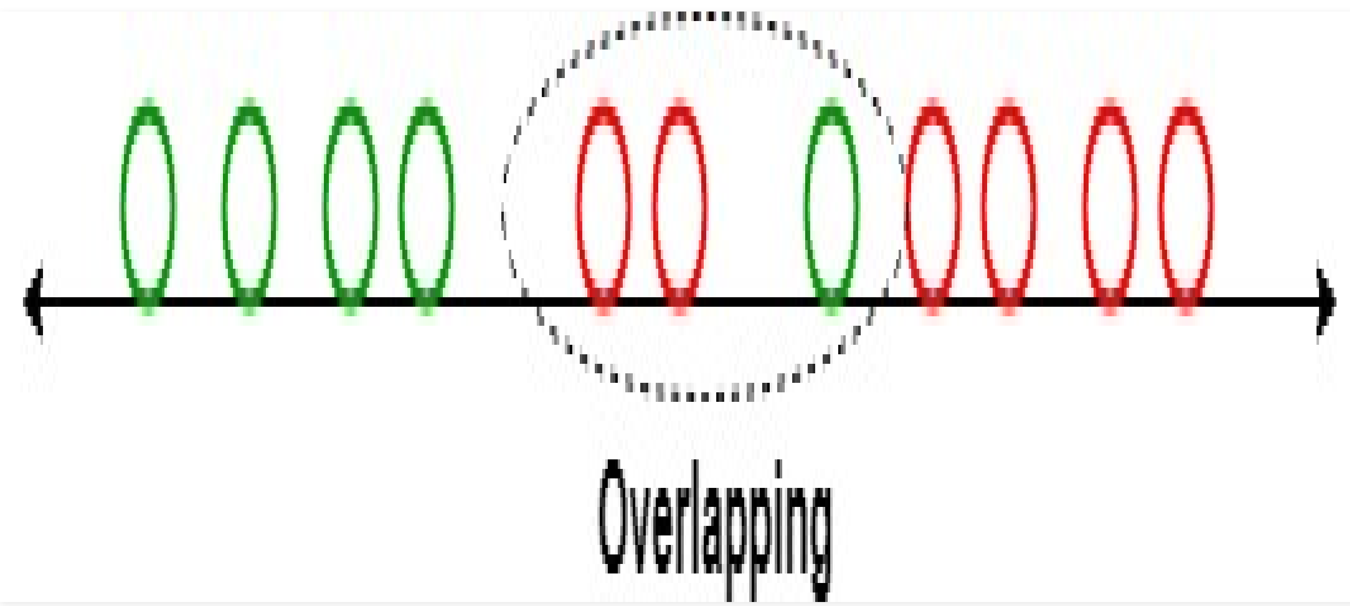
ML | Linear Discriminant Analysis

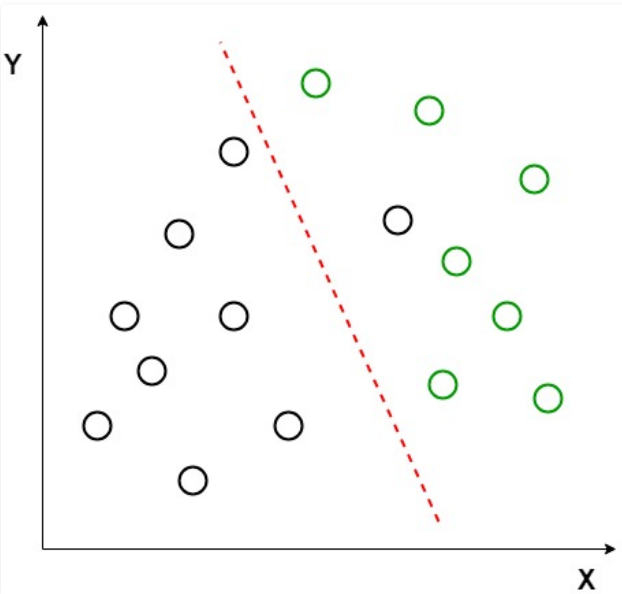
Difficulty Level : Easy • Last Updated : 19 Aug, 2021

Linear Discriminant Analysis or **Normal Discriminant Analysis** or **Discriminant Function Analysis** is a dimensionality reduction technique which is commonly used for the supervised classification problems. It is used for modeling differences in groups i.e. separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space.
For example, we have two classes and we need to separate them efficiently. Classes can have multiple features. Using only a single feature to classify them may result in some overlapping as shown in the below figure. So, we will keep on increasing the number of features for proper classification.



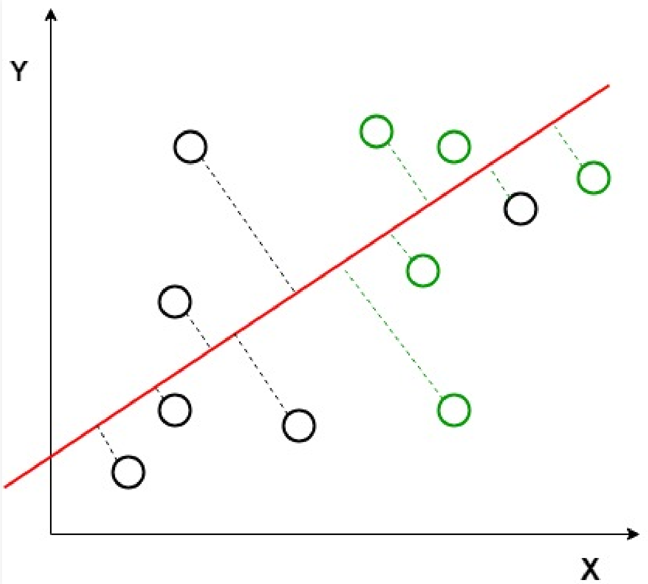
ample:

to classify. As shown in the given 2D graph, when the data points are plotted on the 2D plane, there's no straight line that can separate the two classes of the data points completely. Hence, in this case, LDA (Linear Discriminant Analysis) is used which reduces the 2D graph into a 1D graph in order to maximize the separability between the two classes.

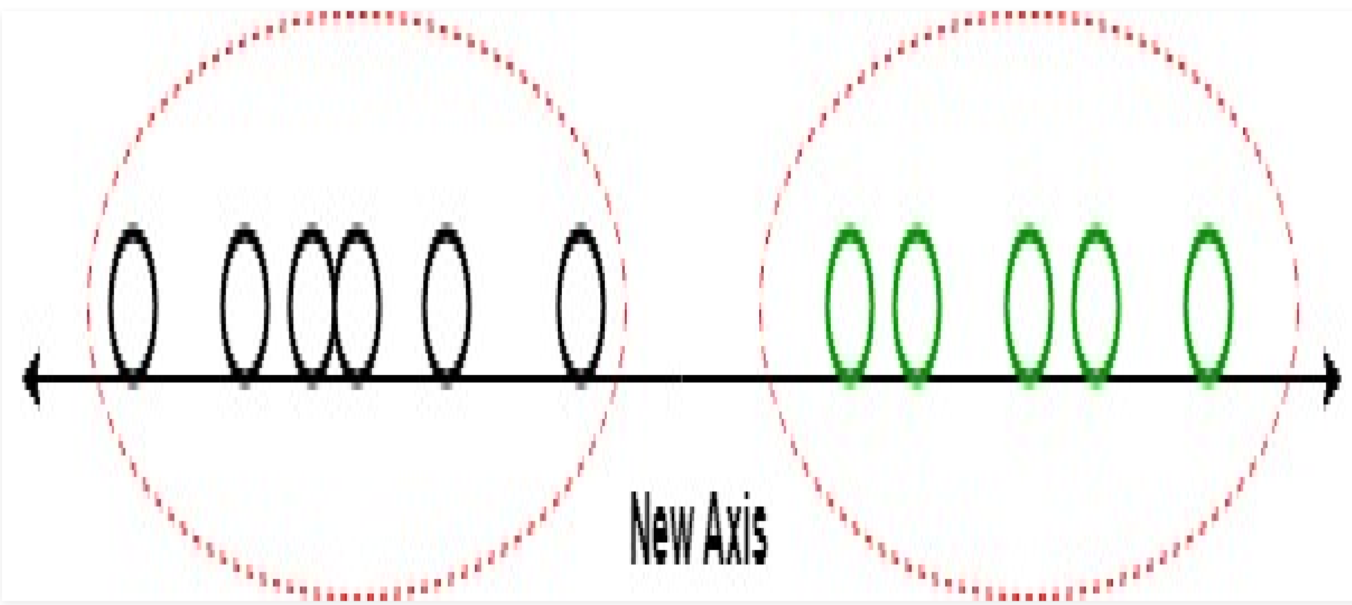


Here, Linear Discriminant Analysis uses both the axes (X and Y) to create a new axis and projects data onto a new axis in a way to maximize the separation of the two categories and hence, reducing the 2D graph into a 1D graph.
Two criteria are used by LDA to create a new axis:

1. Maximize the distance between means of the two classes.
2. Minimize the variation within each class.



In the above graph, it can be seen that a new axis (in red) is generated and plotted in the 2D graph such that it maximizes the distance between the means of the two classes and minimizes the variation within each class. In simple terms, this newly generated axis increases the separation between the data points of the two classes. After generating this new axis using the above-mentioned criteria, all the data points of the classes are plotted on this new axis and are shown in the figure given below.



But Linear Discriminant Analysis fails when the mean of the distributions are shared, as it becomes impossible for LDA to find a new axis that makes both the classes linearly

Let's suppose we have two classes and a d- dimensional samples such as x1, x2 … xn, where:

- n1 samples coming from the class (c1) and n2 coming from the class (c2).

If xi is the data point, then its projection on the line represented by unit vector v can be written as vTxi

Let's consider u1 and u2 be the means of samples class c1 and c2 respectively before projection and u1hat denotes the mean of the samples of class after projection and it can be calculated by:

$\widetilde{\mu_1} = \frac{1}{n_1}\sum_{x_i \in c_1}^{n_1} v^T x_i = v^T \mu_1$

Similarly,

$\widetilde{\mu_2} = v^T \mu_2$

Now, In LDA we need to normalize |\widetilde{\mu_1} -\widetilde{\mu_2} |. Let y_i = v^{T}x_i be the projected samples, then scatter for the samples of c1 is:

$\widetilde{s_1^2} = \sum_{y_i \in c_1} (y_i - \mu_1)^2$

Similarly:

$\widetilde{s_2^2} = \sum_{y_i \in c_1} (y_i - \mu_2)^2$

Now, we need to project our data on the line having direction v which maximizes

$J(v) = \frac{\widetilde{\mu_1} - \widetilde{\mu_2}}{\widetilde{s_1^2} + \widetilde{s_2^2}}$

For maximizing the above equation we need to find a projection vector that maximizes the difference of means of reduces the scatters of both classes. Now, scatter matrix of s1 and s2 of classes c1 and c2 are:

$= \sum_{x_i \in c_1} (x_i - \mu_1)(x_i - \mu_1)^T$

After simplifying the above equation, we get:

Now, we define, scatter within the classes(s_w) and scatter b/w the classes(s_b):

$$s_w = s_1 + s_2$$

$$s_b = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$$

Now, we try to simplify the numerator part of J(v)

$$J(v) = \frac{|\widetilde{\mu_1} - \widetilde{\mu_2}|}{\widetilde{s_1^2} + \widetilde{s_2^2}} = \frac{v^T s_b v}{v^T s_w v}$$

Now, To maximize the above equation we need to calculate differentiation with respect to v

$$\frac{dJ(v)}{dv} = s_b v - \frac{v^t s_b v (s_w v)}{v^T s_w v}$$

$$= s_b v - \lambda s_w v = 0$$

$$s_b v = \lambda s_w v$$

$$s_w^{-1} s_b v = \lambda v$$

$$M v = \lambda v$$

$$where,$$

$$\lambda = \frac{v^T s_b v}{v^T s_w v} and$$

$$M = s_w^{-1} s_b$$

Here, for the maximum value of J(v) we will use the value corresponding to the highest eigenvalue. This will provide us the best solution for LDA.

Extensions to LDA:

1. **Quadratic Discriminant Analysis (QDA):** Each class uses its own estimate of variance (or covariance when there are multiple input variables).
2. **Flexible Discriminant Analysis (FDA):** Where non-linear combinations of inputs is used such as splines.
3. **Regularized Discriminant Analysis (RDA):** Introduces regularization into the estimate of the variance (actually covariance), moderating the influence of different variables

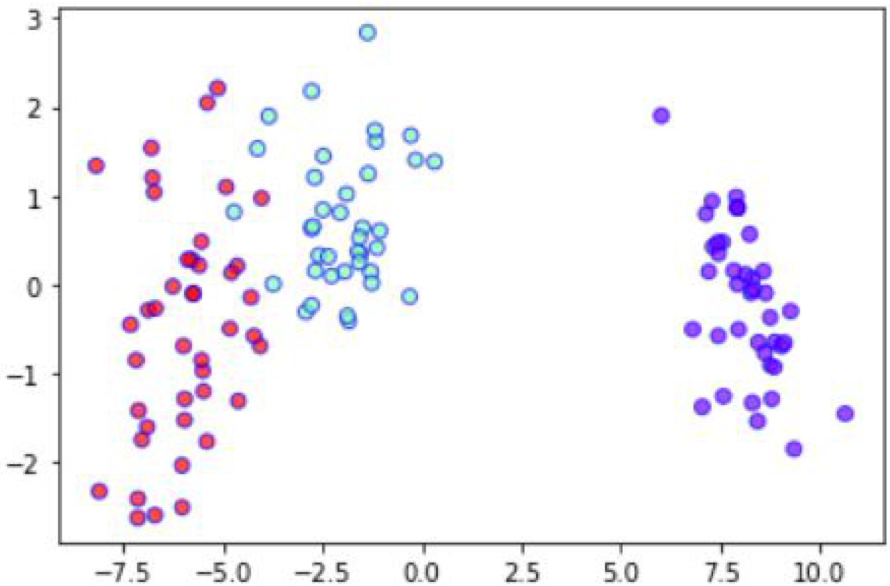
Implementation

- In this implementation, we will perform linear discriminant analysis using the Scikit-learn library on the Iris dataset.

Python3

```
# necessary import
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix

# read dataset from URL
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
cls = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'Class']
dataset = pd.read_csv(url, names=cls)

# divide the dataset into class and target variable
X = dataset.iloc[:, 0:4].values
y = dataset.iloc[:, 4].values

# Preprocess the dataset and divide into train and test
sc = StandardScaler()
X = sc.fit_transform(X)
le = LabelEncoder()
y = le.fit_transform(y)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

# apply Linear Discriminant Analysis
lda = LinearDiscriminantAnalysis(n_components=2)
X_train = lda.fit_transform(X_train, y_train)
X_test = lda.transform(X_test)

# plot the scatterplot
plt.scatter(
    X_train[:,0],X_train[:,1],c=y_train,cmap='rainbow',
  alpha=0.7,edgecolors='b'
)

# classify using random forest classifier
classifier = RandomForestClassifier(max_depth=2, random_state=0)
```

```
# print the accuracy and confusion matrix
print('Accuracy : ' + str(accuracy_score(y_test, y_pred)))
conf_m = confusion_matrix(y_test, y_pred)
print(conf_m)
```



LDA 2 -variable plot

```
Accuracy : 0.9

[[10  0  0]
 [ 0  9  3]
 [ 0  0  8]]
```

Applications:

1. **Face Recognition:** In the field of Computer Vision, face recognition is a very popular application in which each face is represented by a very large number of pixel values. Linear discriminant analysis (LDA) is used here to reduce the number of features to a more manageable number before the process of classification. Each of the new dimensions generated is a linear combination of pixel values, which form a template. The linear combinations obtained using Fisher's linear discriminant are called Fisher faces.

and the medical treatment he is going through. This helps the doctors to intensify or reduce the pace of their treatment.

3. **Customer Identification:** Suppose we want to identify the type of customers which are most likely to buy a particular product in a shopping mall. By doing a simple question and answers survey, we can gather all the features of the customers. Here, Linear discriminant analysis will help us to identify and select the features which can describe the characteristics of the group of customers that are most likely to buy that particular product in the shopping mall.

Previous Next

18, Jan 19

03 **Regularized Discriminant Analysis**
27, May 21

07 **Exploratory Data Analysis in Python | Set 2**
18, Jan 19

04 **Analysis of test data using K-Means Clustering in Python**
07, Jan 18

08 **ML | R-squared in Regression Analysis**
07, May 19

Article Contributed By :

raman_257
@raman_257

Vote for difficulty

Current difficulty : Easy

Easy Normal Medium Hard Expert

Improved By : pawangfg

Article Tags : Machine Learning

Practice Tags : Machine Learning

Improve Article Report Issue

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments