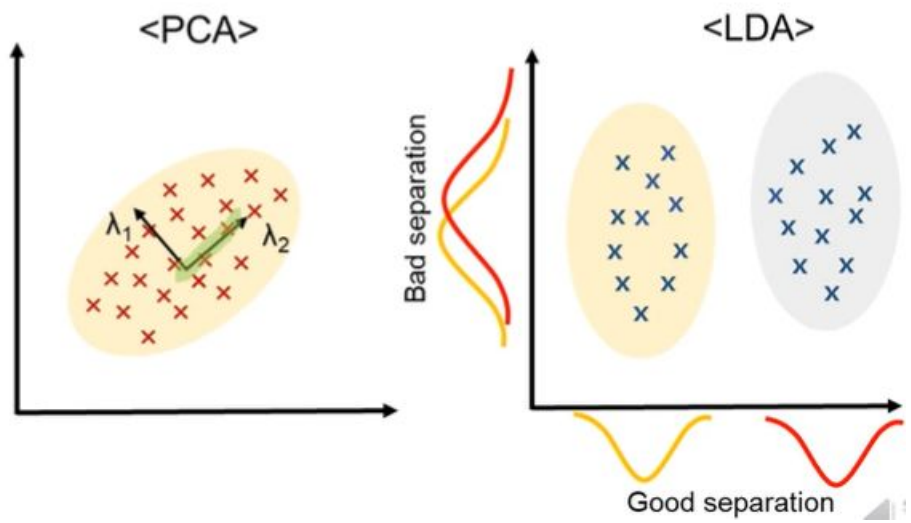
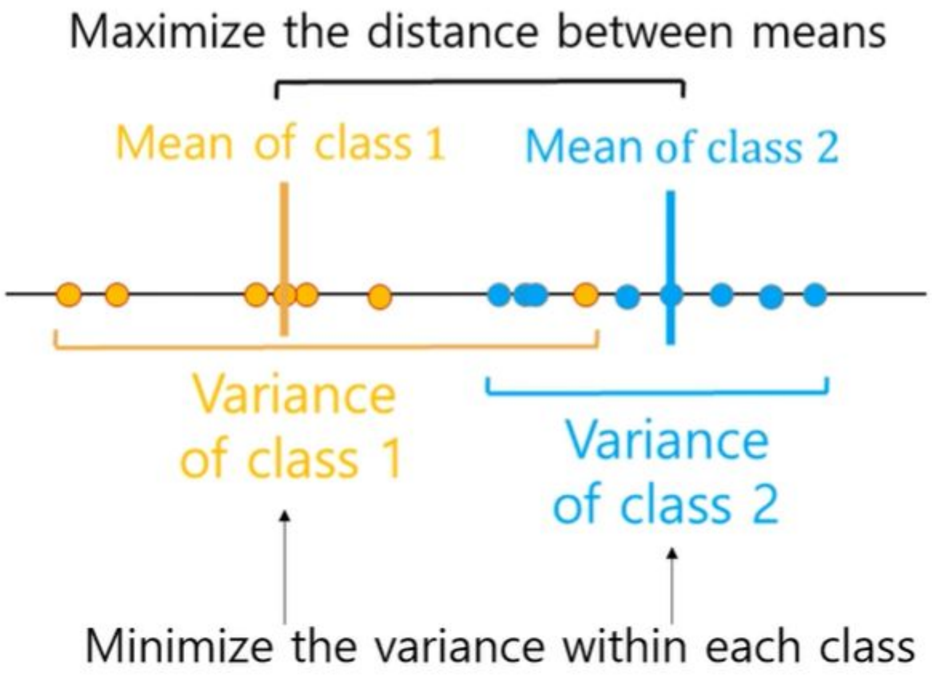
Linear Discriminant Analysis (LDA)

- Like PCA, LDA is a representative technique popularly used to reduce the dimensionality of large datasets.
- LDA requires class labels (supervised), while PCA does not require class labels (unsupervised).
- The purpose of LDA is to project the dataset onto a lower-dimensional space while maximizing the separation between multiple classes, while that of PCA is to find the principal components that maximize the variance in a dataset.

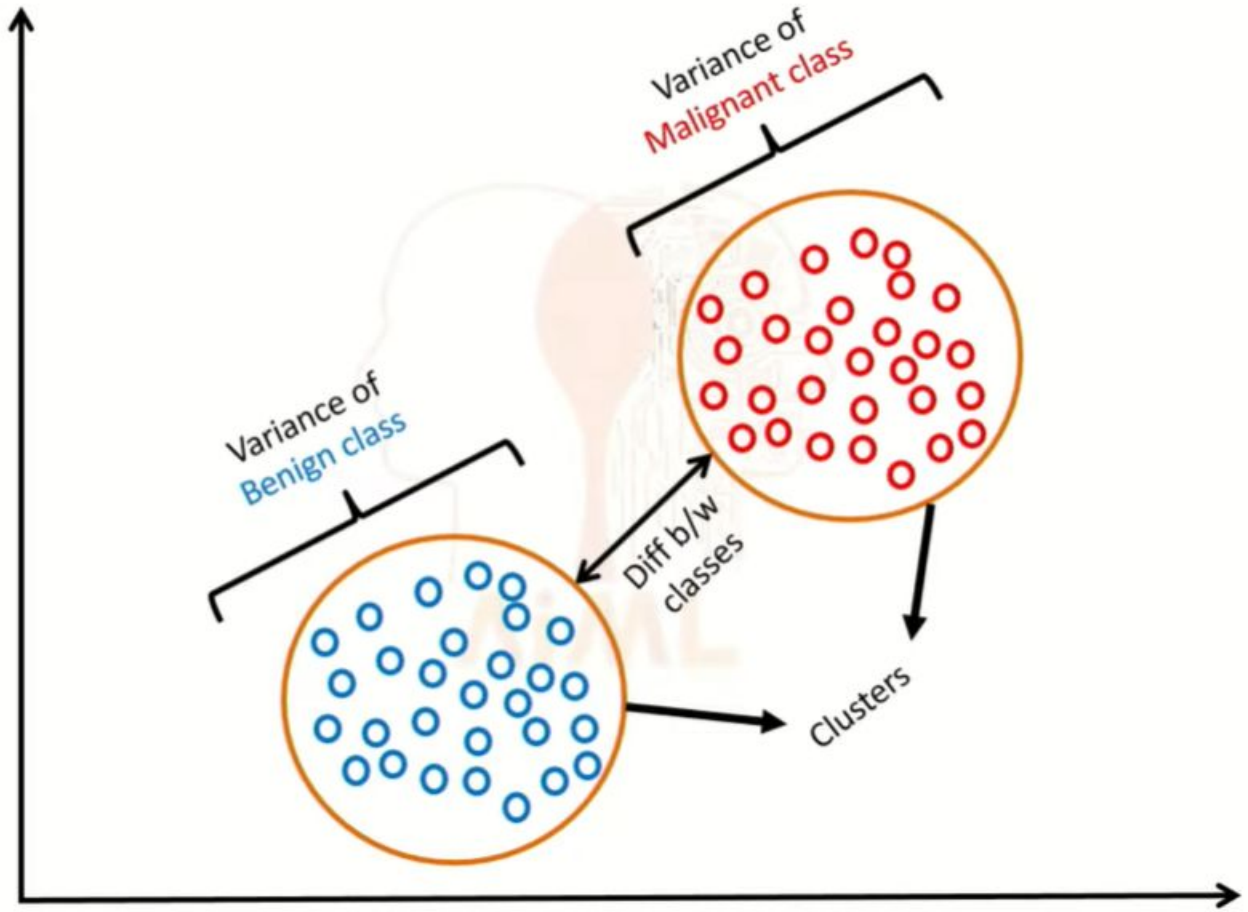


Linear Discriminant Analysis (LDA)

- LDA maximizes the between-class variance (between-class scatter, SB), while minimizing the within-class variance (within-class scatter, SW),

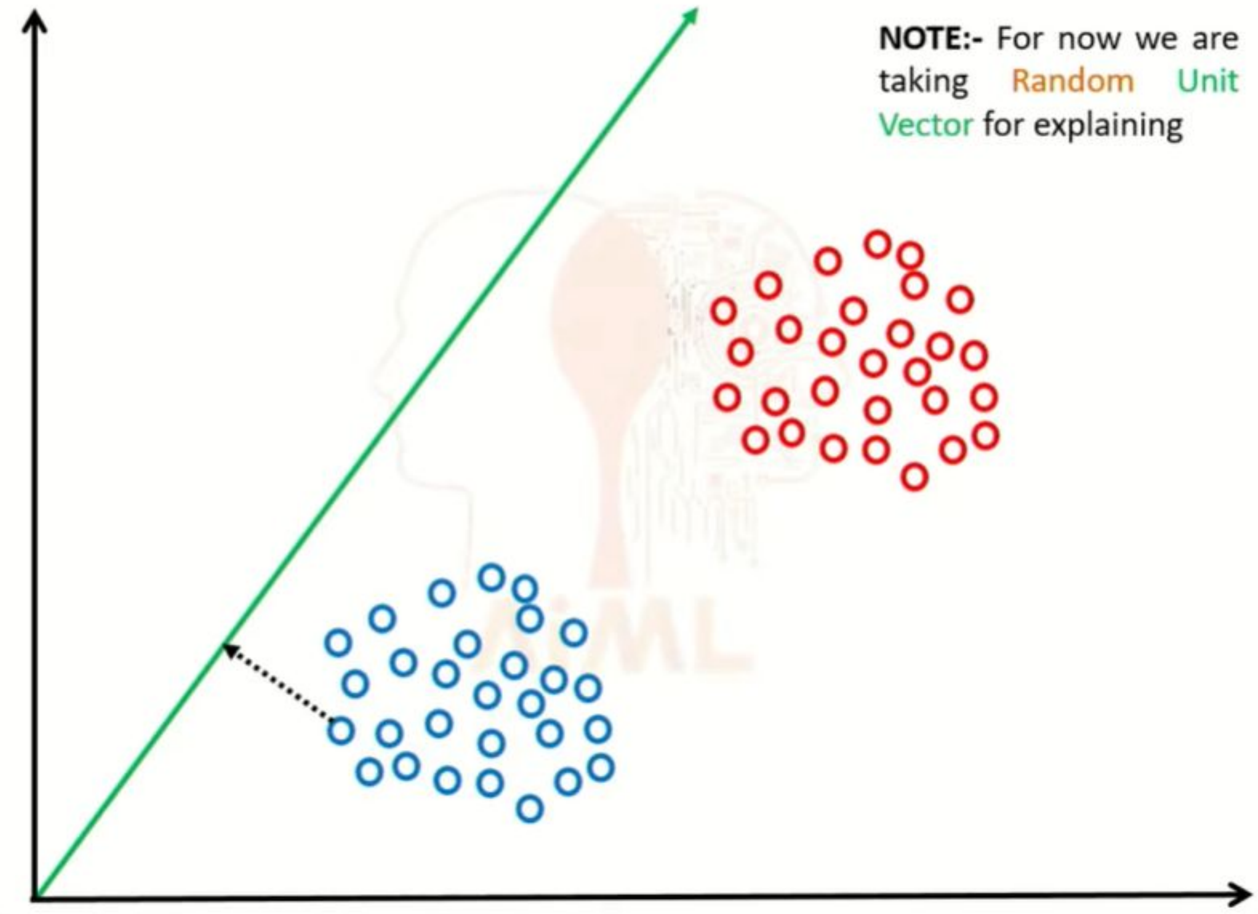


LDA as Dimensionality Reduction

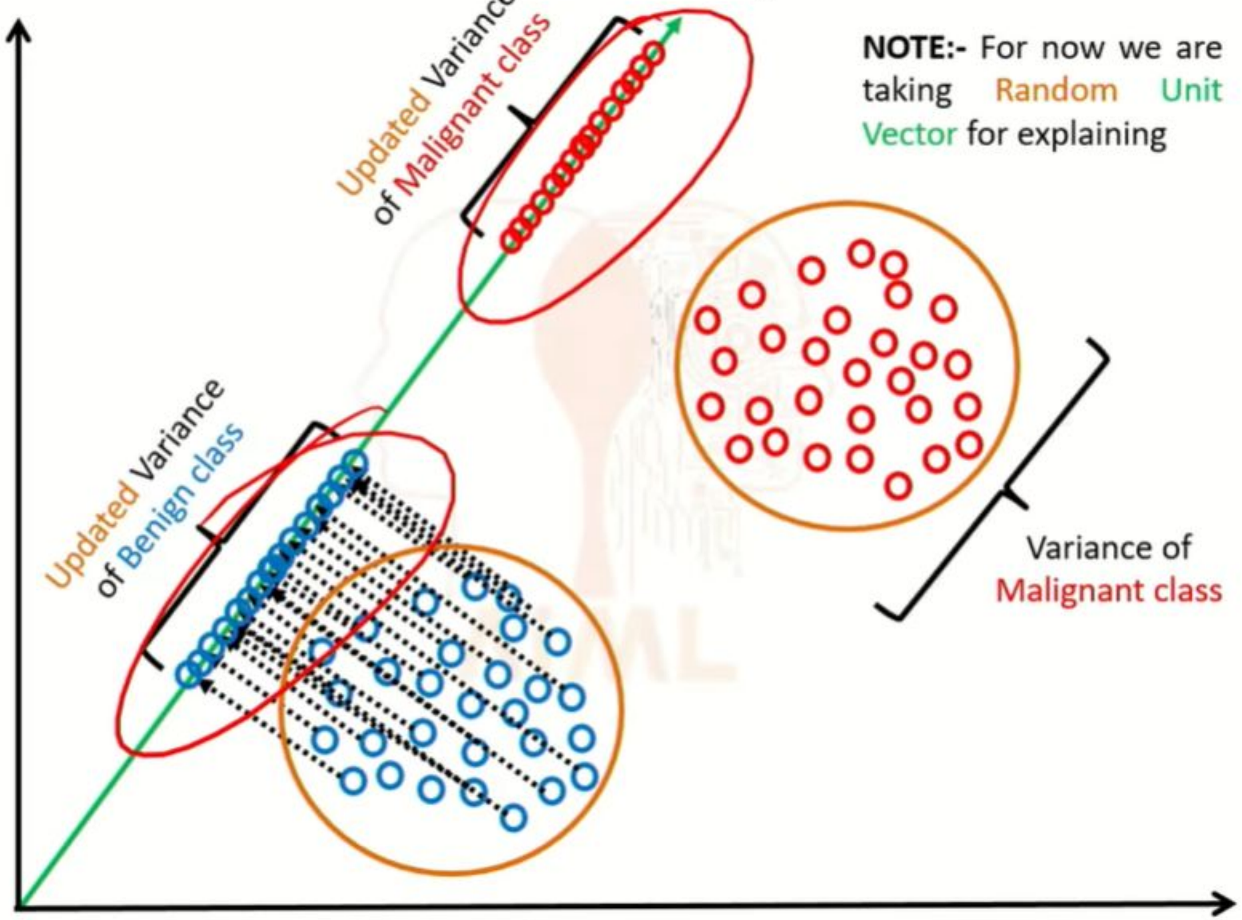


LDA as Dimensionality Reduction

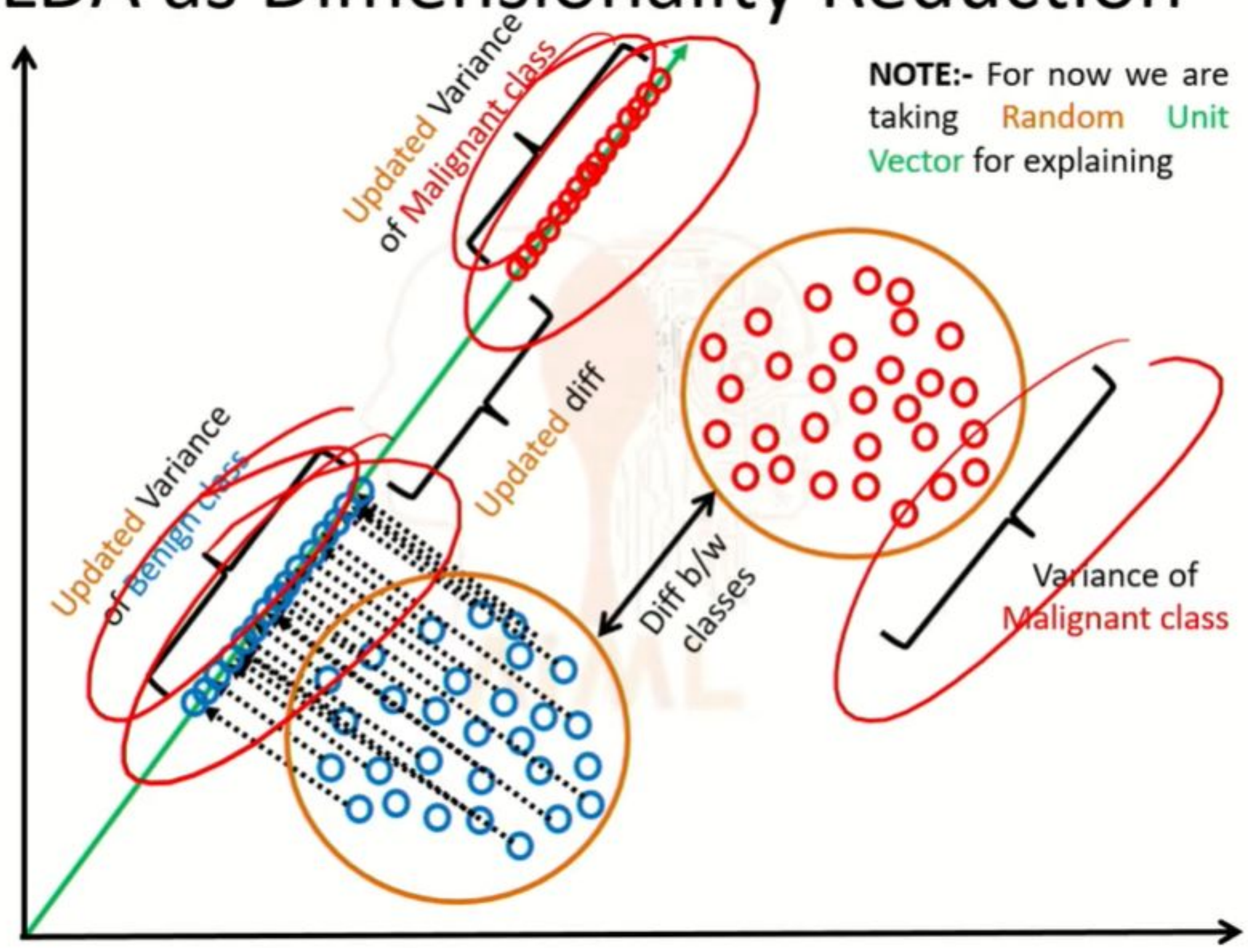
NOTE:- For now we are taking Random Unit Vector for explaining



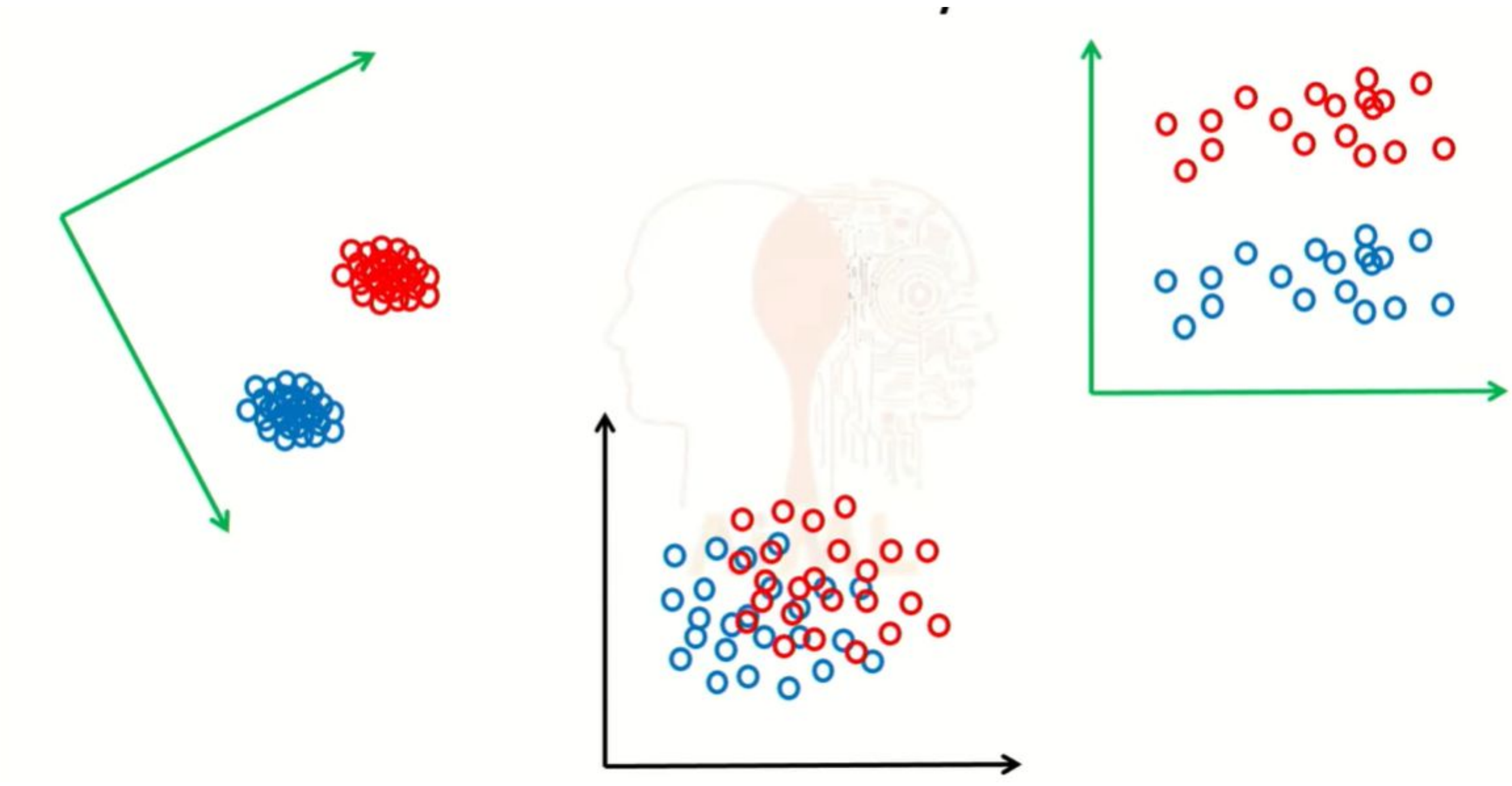
LDA as Dimensionality Reduction



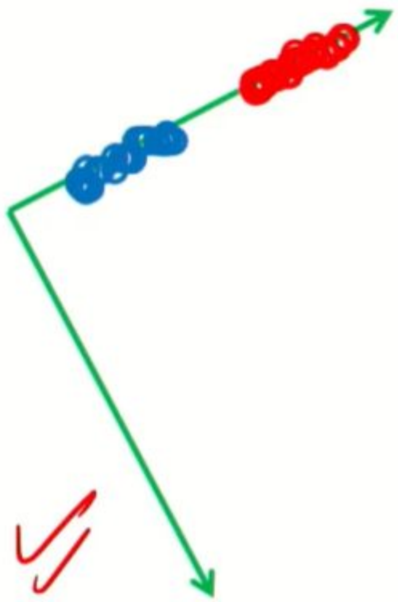
LDA as Dimensionality Reduction

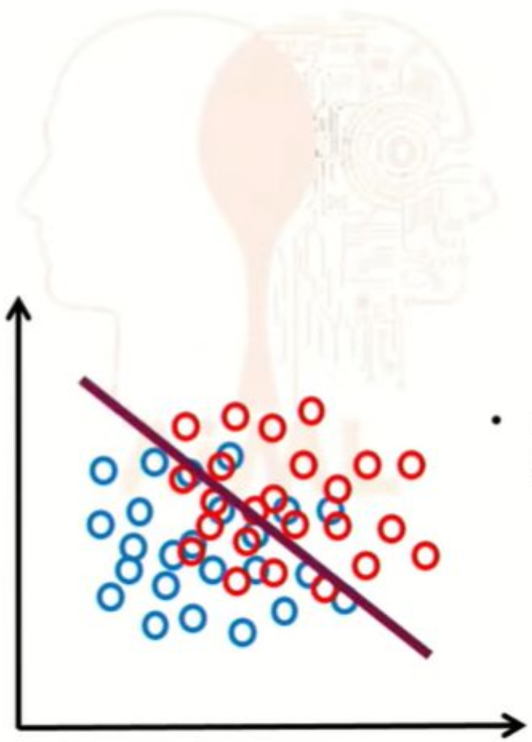


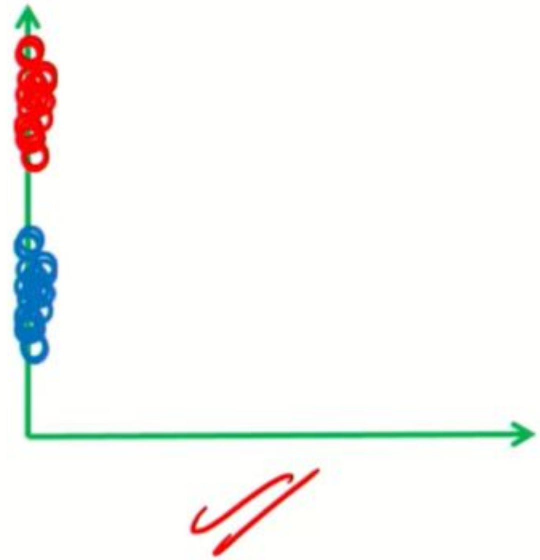
LDA – Linear Discriminant Analysis



LDA – Linear Discriminant Analysis

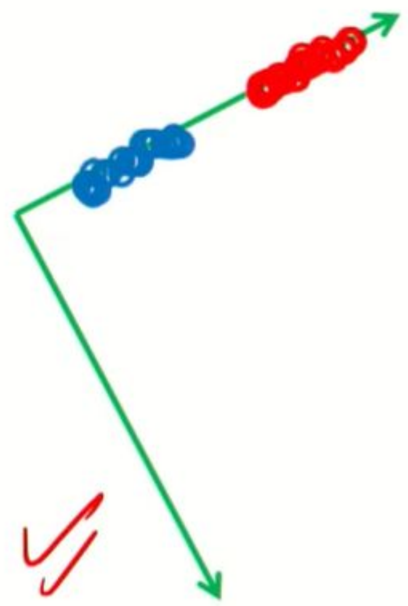


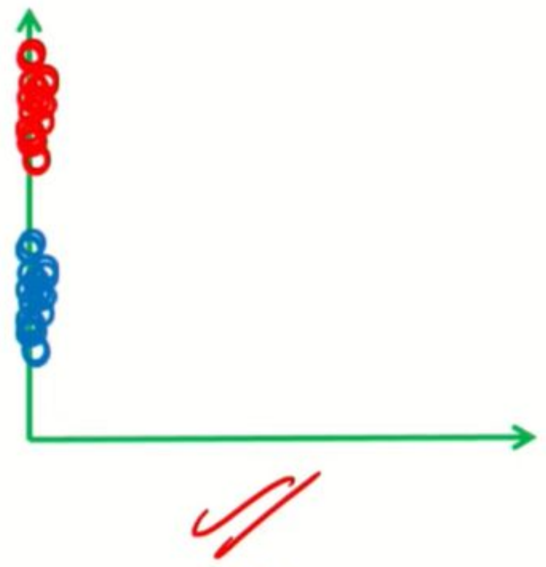


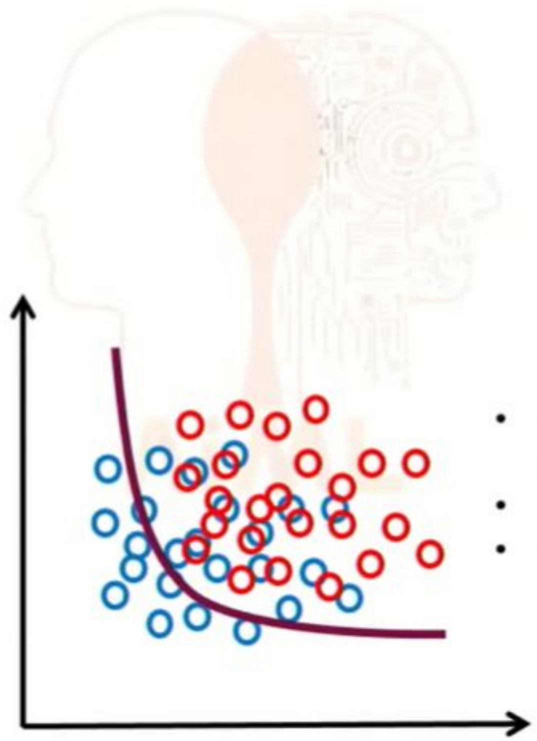


- If we try to apply LDA in this case you will get miss-classification

LDA – Linear Discriminant Analysis







- If we try to apply LDA in this case you will get miss-classification
- So, in this case LDA is not a good option.
- Let’s try to apply QDA.

Linear Discriminant Analysis (LDA)

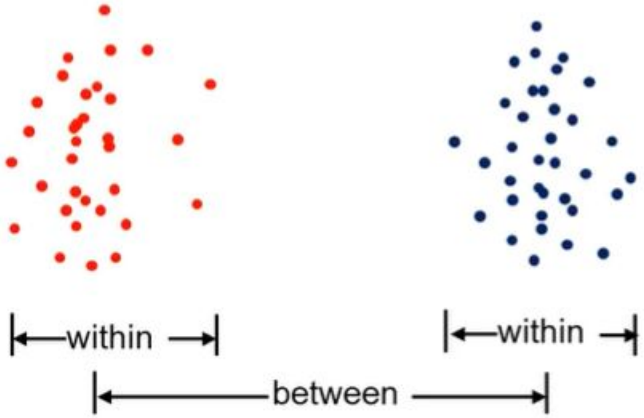
- The classification performance becomes better with increasing ratio of SB to SW

(high)

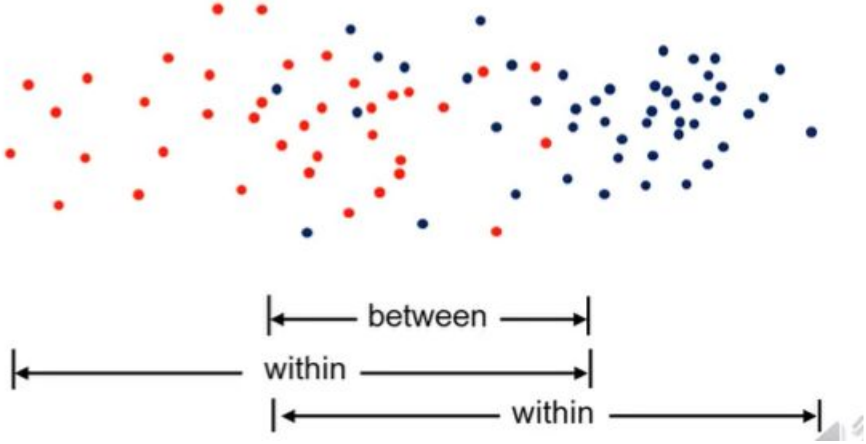
$$\frac{\text{between}-\text{class variance (SB)}}{\text{within}-\text{class variance (SW)}} \Rightarrow \frac{\text{ideally large}}{\text{ideally small}}$$

(low)

< Classes are well separated>



<Classes are NOT well separated>



Linear Discriminant Analysis (LDA)

Principal component analysis (PCA)

Maximize the variance of data

↓

Compute eigenvalues/ eigenvectors

↓

Select eigenvectors with large eigenvalues

Linear Discriminant Analysis (LDA)

Maximize $\left(\frac{\text{between}-\text{class}-\text{scatter (SB)}}{\text{within}-\text{class}-\text{scatter (SW)}}\right)$

Eigenvalue

$$AX = \lambda X$$

Eigenvector

$$SW^{-1}SB \times X = \lambda \times X$$

$$\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \lambda_n \end{bmatrix}$$

- The eigenvectors with larger eigenvalues are selected.

LDA coding

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler

import matplotlib.pyplot as plt
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
```

```
df = pd.read_excel("C:₩₩Users₩₩oem₩₩datasets₩₩iris-dataset.xlsx", header = 0)
```

```
data = np.array(df, dtype=np.float32)

y_data = data[:, [-1]]

scaler = MinMaxScaler()
data1 = scaler.fit_transform(df.values)

x_data = data1[:, 0:-1]
```

Features (X) Label (Y)

	X1	X2	X3	X4	
	A	B	C	D	E
1	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
2	5.1	3.5	1.4	0.2	0
3	4.9	3	1.4	0.2	0
4	6.3	2.3	4.4	1.3	1
5	5.6	3	4.1	1.3	1
6	6.5	3.2	5.1	2	2
7	6.4	2.7	5.3	1.9	2
8	6.8	3	5.5	2.1	2

0: setosa
1: versicolor
2: virginica

LDA coding

```
lda = LinearDiscriminantAnalysis(n_components=2)
lda.fit(x_data, y_data)
iris_lda = lda.transform(x_data)

lda_columns = ['LD_1', 'LD_2']
irisDF_lda = pd.DataFrame(iris_lda, columns = lda_columns)
print (lda.explained_variance_ratio_)
print (irisDF_lda)
```

```
[0.9912126 0.0087874]
         LD_1      LD_2
0    8.061800  0.300421
1    7.128688 -0.786660
2    7.489828 -0.265384
3    6.813201 -0.670631
4    8.132309  0.514463
..        ...       ...
145 -5.645003  1.677717
146 -5.179565 -0.363475
147 -4.967741  0.821141
148 -5.886145  2.345091
149 -4.683154  0.332034

[150 rows x 2 columns]
```

```
irisDF_lda['target']=y_data
print (irisDF_lda)
```

```
         PC_1      PC_2  target
0    8.061800  0.300421     0.0
1    7.128688 -0.786660     0.0
2    7.489828 -0.265384     0.0
3    6.813201 -0.670631     0.0
4    8.132309  0.514463     0.0
..        ...       ...     ...
145 -5.645003  1.677717     2.0
146 -5.179565 -0.363475     2.0
147 -4.967741  0.821141     2.0
148 -5.886145  2.345091     2.0
149 -4.683154  0.332034     2.0

[150 rows x 3 columns]
```

LDA coding

```
markers = ['^', 's', 'o']
target_names = ['Sentosa','verisicolor','virginica']

for i, marker in enumerate (markers):
    x_axis_data = irisDF_lda[irisDF_lda['target']==i]['LD_1']
    y_axis_data = irisDF_lda[irisDF_lda['target']==i]['LD_2']
    plt.scatter(x_axis_data, y_axis_data, marker=marker, label = target_names[i])

plt.legend()
plt.xlabel('LD_1')
plt.ylabel('LD_2')
plt.show()
```