21/11/25

PAGE :
DATE : / /

1. Random forest Ensembling algorithm
2. Boosting
3. K-mean clustering unsupervised.
4. PCA Principal Component Analysis

2.

```
from sklearn.datasets import load_iris
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
data = load_iris()
x = data.data
y = data.target
target_name = data.target_names
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size
=0.4, random_state = 42)
rf_model = RandomForestClassifier(n_estimators=100, random_state=
42)
rf_model.fit(x_train, y_train)
y_pred = rf_model.predict(x_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy of Random Forest : {accuracy * 100 : .2f}%
")
print("Classification Report :")
print(classification_report(y_test, ypred, target_name = target_names))
print("Confusion Report :")
print(classification_report(y_test, y_pred, target_names = target_name))
print("confusion matrix:")
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

o/p: Accuracy of Random forest : 98.33 %

Classification Report:

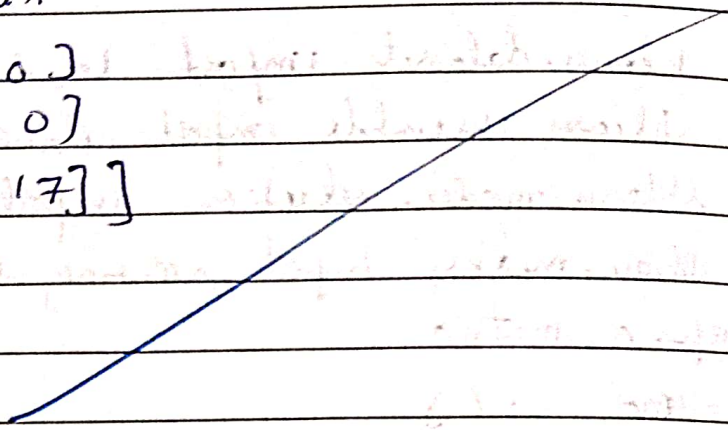
	Precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	23
Versicolor	0.95	1.00	0.97	19

PAGE :
DATE : / /

Virginica	1.00	0.94	0.97	18
accuracy			0.98	60
macro avg	0.99	0.98	0.98	60
weighted avg	0.98	0.98	0.98	60

Confusion Matrix:-

[[23 0 0]
[0 19 0]
[0 1 17]]



Q.

PAGE :
DATE : / /

2 Boosting

```
from sklearn.datasets import load_iris
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

data = load_iris()
x = data.data
y = data.target
target_names = data.target_names

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=
    0.3, random_state=42)
base_estimator = DecisionTreeClassifier(max_depth=1)
boost_model = AdaBoostClassifier(estimator=base_estimator,
    n_estimators=50, random_state=42)
boost_model.fit(x_train, y_train)
y_pred = boost_model.predict(x_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy of adaBoost: {accuracy*100:.2f}%")
print("Classification Report:")
print(classification_report(y_test, y_pred, target_names=
                                target_names))
print("Confusion Matrix:")
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

PAGE :
DATE : / /

Accuracy of AdaBoost : 100.00%

Classification Report :

	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	19
Versicolor	1.00	1.00	1.00	13
Virginica	1.00	1.00	1.00	13
			1.00	45
Accuracy	1.00	1.00	1.00	45
macro avg	1.00	1.00	1.00	45
weighted avg				

Confusion Matrix :

[[19 0 0]
 [0 13 0]
 [0 0 13]]

K-mean

PAGE:
DATE: / /

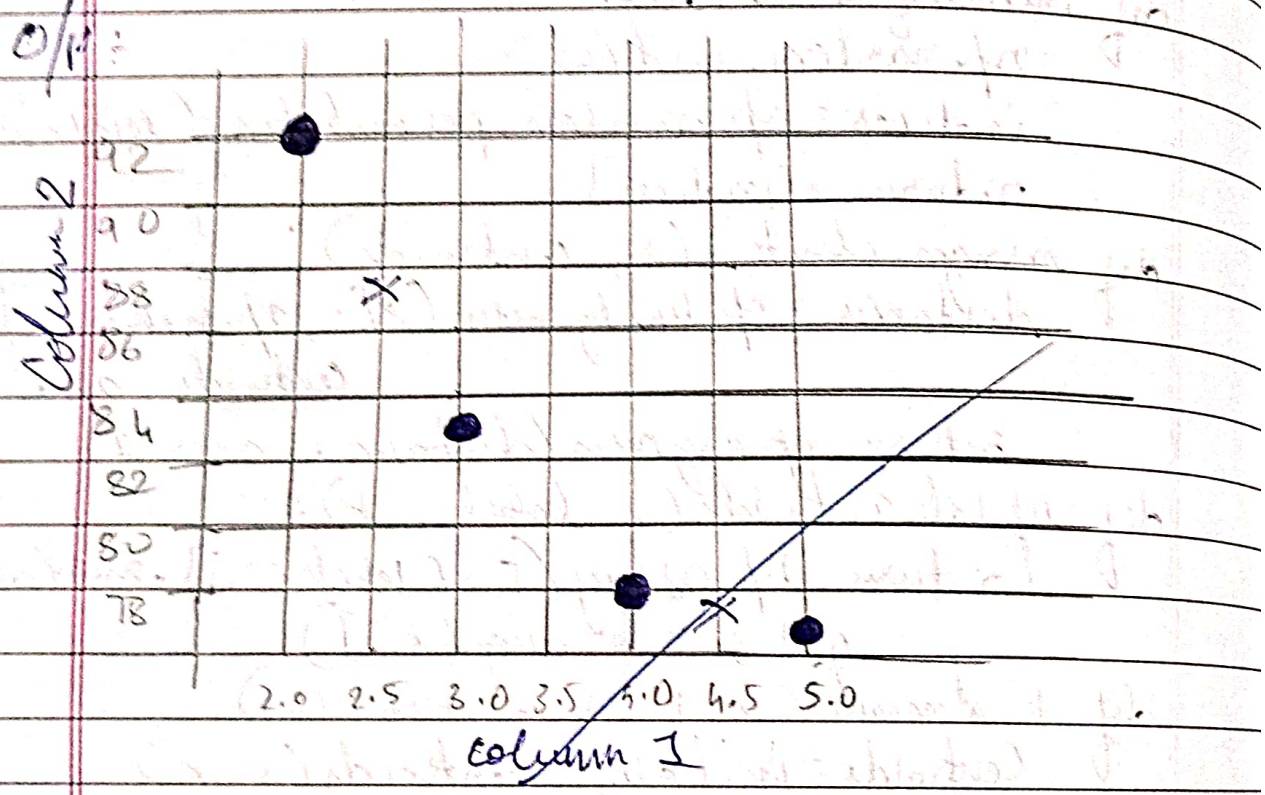
3
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

def load_data():
    df = pd.read_csv("/mj_lab.csv")
    return df.values
def initialize_centroids(x, K):
    np.random.seed(42)
    indices = np.random.permutation(len(x))[:K]
    return x[indices]
def assign_clusters(x, centroids):
    distances = np.linalg.norm(x[:, np.newaxis] -
                               centroids, axis=2)
    return np.argmin(distances, axis=1)
def update_centroids(x, label, K):
    return np.array([x[labels == i].mean(axis=0)
                     for i in range(K)])
def K_dmeans(x, K, max_iters=100):
    Centroids = initialize_centroids(x, K)
    for _ in range(max_iters):
        labels = assign_clusters(x, centroids)
        new_centroids = update_centroids(x, labels, K)
        if np.allclose(centroids, new_centroids):
            break
        centroids = new_centroids
    return centroids, labels
def plot_clusters(x, labels, centroids):
    plt.scatter(x[:, 0], x[:, 1], c=labels, cmap=
                'viridis', s=100)
    plt.scatter(centroids[:, 0], centroids[:, 1], c='red',
                s=200, marker='x')
    plt.xlabel("Column 1")
    plt.ylabel("Column 2")
```

PAGE :
DATE : / /

```
plt.title("K-Means Clustering")
plt.grid(True)
plt.show()
X = load_data()
K = 2
centroids, labels = K_means(X, K)
plot_clusters(X, labels, centroids)
```

O/P :



PCA

PAGE :
DATE : / /

4.
```
import numpy as np
from sklearn.decomposition import PCA
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

data = load_iris()
X = data.data
y = data.target
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X_scaled)
print("Explained variance ratio of each principal component:")
print(pca.explained_variance_ratio_)

X_train, X_test, y_train, y_test = train_test_split(X_reduced, y, test_size=0.3, random_state=42)

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
print(f"Accuracy on PCA-reduced data: {accuracy_score(y_test, y_pred)*100:.2f}%")
```

o/p:- Explained variance ratio of each principal component:
[0.72962445 0.22850762]
Accuracy on PCA-reduced data: 95.56%