**Practical Number 3**

**Aim:** K-Means Clustering using scikit-learn built-in dataset, Iris Flowers

**Theory:**

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

**Code:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import matplotlib.patches as patches

from sklearn import datasets

from sklearn.cluster import KMeans

from sklearn.metrics import accuracy\_score

# Load the iris dataset

iris = datasets.load\_iris()

# Use only two features for visualization purposes

X = iris["data"][:, (2, 3)]

y = iris["target"]

# Train the K-Means clustering model

kmeans = KMeans(n\_clusters=3)

kmeans.fit(X)

# Predict the cluster assignments

y\_pred = kmeans.predict(X)

# Calculate the accuracy

accuracy = accuracy\_score(y, y\_pred)

print("Accuracy: {:.2f}".format(accuracy))

# Plot the actual values, predicted values, and difference

fig, ax = plt.subplots(1, 3, figsize=(18, 4))

# Plot the actual values

sc = ax[0].scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Paired, edgecolor="black")

ax[0].set\_xlabel("Petal length")

ax[0].set\_ylabel("Petal width")

ax[0].set\_title("Actual values")

# Plot the predicted values

sc = ax[1].scatter(X[:, 0], X[:, 1], c=y\_pred, cmap=plt.cm.Paired, edgecolor="black")

ax[1].set\_xlabel("Petal length")

ax[1].set\_ylabel("Petal width")

ax[1].set\_title("Predicted values")

# Plot the difference

difference = y - y\_pred

sc = ax[2].scatter(X[:, 0], X[:, 1], c=difference, cmap=plt.cm.Paired, edgecolor="black")

ax[2].set\_xlabel("Petal length")

ax[2].set\_ylabel("Petal width")

ax[2].set\_title("Difference")

# Select five random difference points

n\_points = 5

random\_indices = np.random.choice(range(len(difference)), n\_points, replace=False)

random\_points = X[random\_indices]

# Add a box around each selected difference point

for point in random\_points:

ax[2].add\_patch(patches.Rectangle(point - 0.3, 0.6, 0.6, color="red", fill=None))

plt.show()

**Results:**

