**Name: Mahmood Rasheed**

**Roll no: 20SW047**

**Section: 1**

**Department: Software**

**Subject: Data Science & Analytics**

**Submitted To: Madam Rafia Sheikh**

**Supervised Algorithims:**

**1. Naive Bayes**

**2. KNN Classifier**

**3. Decision Tree (C4.5)**

**1. Naive Bayes:**

* **Define:** Naïve Bayes is a probabilistic classification algorithm based on Bayes' theorem. It assumes that features are conditionally independent given the class.
* **K-Fold Cross-Validation:** Split the dataset into three equal parts (3-fold).
* **Algorithm Steps:**

1. Calculate the prior probability of each class (e.g., P(Class=Yes), P(Class=No)).
2. For each feature (Refund, Status, Tax Income), calculate the likelihood of observing that feature given each class (e.g., P(Refund=Yes|Class=Yes), P(Refund=Yes|Class=No)).
3. Use Bayes' theorem to calculate the posterior probability of each class for each data point.
4. Assign the class with the highest posterior probability as the predicted class.

* **Evaluate Performance:** Calculate accuracy, precision, recall, F1-score, or other relevant metrics using cross-validation.

**2. KNN Classifier (K-Nearest Neighbors):**

**Definition:** KNN is a classification algorithm that assigns an object to the class most common among its k nearest neighbors in the feature space.  
  
**Algorithm Steps:**

1. Choose the number of neighbors (k).
2. Calculate the distance between the target point and all data points.
3. Select the k-nearest neighbors based on distance.
4. Assign the class that occurs most frequently among the neighbors as the prediction.

**Evaluate:** Apply 3-fold cross-validation and assess performance.

**3. Decision Tree (C4.5):**

**Definition:** C4.5 is an algorithm for generating decision trees that are used for classification. It selects the best attributes to split the dataset based on information gain or other criteria.

**Steps:**

1. Select the best attribute to split the dataset.
2. Divide the dataset into subsets based on the chosen attribute's values.
3. Recursively repeat steps 1 and 2 for each subset until a stopping criterion is met.
4. Assign class labels to the leaf nodes.

**Evaluate:** Use 3-fold cross-validation to evaluate its performance.

**Unsupervised Algorithm:**

1. **K-Means Clustering**
2. **Hierarchical Clustering**

**1. K-Means Clustering:**

**Initialization:** Start with the initial seeds A1, A4, and A7 as cluster centers.

**Iteration 1:**

1. Calculate the Euclidean distance between each point and the cluster centers (A1, A4, A7).
2. Assign each point to the nearest cluster based on the smallest distance.
3. Calculate the new cluster centers by taking the mean of the points in each cluster.

**Result After 1st Iteration:**

* + New Clusters:
    - Cluster 1: A2, A3, A6
    - Cluster 2: A5, A8
    - Cluster 3: A1, A4, A7
  + New Centers:
    - Cluster 1 Center: (5.33, 5.67)
    - Cluster 2 Center: (5.5, 8.5)
    - Cluster 3 Center: (2.33, 7.67)
  + Plot:
    - Draw a 10x10 space with all 8 points.
    - Color points in each cluster differently.
    - Plot the new centroids for each cluster.

**Guess for Convergence:**

* Without running the algorithm again, it's challenging to precisely guess the number of iterations required to converge.
* K-Means typically converges when the cluster assignments and centroids no longer change significantly.
* In this case, since the points have moved closer to their respective centroids, it's possible that just a few more iterations are needed for convergence.

**2. Hierarchical Clustering:**

* **Single Linkage:**
  + Calculate the pairwise Euclidean distances between all points.
  + Start with each point as its own cluster.
  + Merge the two closest clusters based on the minimum distance between any pair of points in the two clusters.
  + Repeat the merging until you have 3 clusters.
* **Complete Linkage:**
  + Similar to single linkage but merge clusters based on the maximum distance between any pair of points in the two clusters.
* **Group Average:**
  + Merge clusters based on the average distance between all pairs of points from the two clusters.
* **Distance Between Centroids:**
  + Calculate the Euclidean distance between the centroids of each pair of clusters.
  + Merge the two clusters with the closest centroids.
* **Result:**
  + For each of the four hierarchical clustering methods, you'll get a dendrogram that shows how the points are grouped into clusters at different levels of granularity. You can then choose the number of clusters you want based on the dendrogram.

**THE END**