📘 Machine Learning

✨ Assessment 2

MBA IT - Division A (Data Analytics)

👨‍🎓 **Students:**

- Avash Sahu *(ID: 24030141012)*
- Shlok Tilokani *(ID: 24030141072)*

📂 **Dataset Used:** *Iris Flower Dataset*

1. **Import Libraries**
   - Use `pandas` for data handling
   - Use `scikit-learn` for machine learning

In [268…
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
```

2. **Load Dataset**
   - Load the `IRIS.csv` file into a Pandas DataFrame
   - Define the **target column**

In [269…
```
df = pd.read_csv("IRIS.csv")
df.head()
target_column = "species"
```

3. **Data Preparation**
   - Separate dataset into:
     - **Features (X)**
     - **Target variable (y)**

In [270…
```
X = df.drop(target_column, axis=1)
y = df[target_column]
```

4. **Train-Test Split**
   - Split data into:

- **40% Training set**
- **60% Testing set**

In [271…
```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.6, random_state=20
)
```

5. **Model Initialization & Training**
   - **K-Nearest Neighbors (KNN)** classifier
   - **Decision Tree** classifier

In [272…
```
# Model 1: K-Nearest Neighbors
knn_model = KNeighborsClassifier(n_neighbors=25)
knn_model.fit(X_train, y_train)
```

Out[272…
```
KNeighborsClassifier
▶ Parameters
```

In [273…
```
# Model 2: Decision Tree
tree_model = DecisionTreeClassifier(max_depth=3, random_state=42)
tree_model.fit(X_train, y_train)
```

Out[273…
```
DecisionTreeClassifier
▶ Parameters
```

6. **Model Prediction**
   - Use both trained models to predict on the **test dataset**

In [274…
```
knn_predictions = knn_model.predict(X_test)
tree_predictions = tree_model.predict(X_test)
```

7. **Model Evaluation**
   - Calculate accuracy scores for both models:
     - ✅ **KNN Accuracy:** `88.89%`
     - ✅ **Decision Tree Accuracy:** `87.78%`

In [275…
```
knn_accuracy = accuracy_score(y_test, knn_predictions)
tree_accuracy = accuracy_score(y_test, tree_predictions)
```

In [276…
```
print(f"K-Nearest Neighbors Model Accuracy: {knn_accuracy:.2%}")
print(f"Decision Tree Model Accuracy:       {tree_accuracy:.2%}")
```

```
K-Nearest Neighbors Model Accuracy: 88.89%
Decision Tree Model Accuracy:       87.78%
```

8. **Custom Data Testing**
   - Define a list of **custom test points**:
     - Easy cases
     - Ambiguous cases
     - Outlier cases
   - Run predictions for each case using both models
   - Print and compare the results

In [280…
```
test_cases = [
    # --- Case 1: Easy Predictions ---
    {"description": "Test 1.1 (Easy)", "data": [[5.0, 3.5, 1.5, 0.2]]},
    {"description": "Test 1.2 (Easy)", "data": [[4.8, 3.4, 1.6, 0.2]]},
    {"description": "Test 1.3 (Easy)", "data": [[5.2, 3.8, 1.4, 0.3]]},
    # --- Case 2: Ambiguous Predictions ---
    {"description": "Test 2.1 (Ambiguous)", "data": [[6.5, 3.0, 5.0, 1.6]]},
    {"description": "Test 2.2 (Ambiguous)", "data": [[6.7, 3.1, 5.2, 1.8]]},
    {"description": "Test 2.3 (Ambiguous)", "data": [[6.0, 2.8, 4.8, 1.7]]},
    # --- Case 3: Outlier Predictions ---
    {"description": "Test 3.1 (Outlier)", "data": [[7.5, 2.5, 7.0, 0.5]]},
    {"description": "Test 3.2 (Outlier)", "data": [[5.0, 2.0, 2.5, 2.0]]},
    {"description": "Test 3.3 (Outlier)", "data": [[8.0, 5.0, 1.5, 0.5]]},
]

feature_names = X.columns
```

In [279…
```
for case in test_cases:
    print(f"\n--------------------------------------")
    print(f"{case['description']}")
    print("--------------------------------------")

    # Create a DataFrame for the single test case with correct column names
    test_df = pd.DataFrame(case["data"], columns=feature_names)
    print("Input Features:")
    print(test_df)

    # Use the already-trained models to make predictions
    knn_pred = knn_model.predict(test_df)
    tree_pred = tree_model.predict(test_df)

    print(f"\n\t -> KNN Model Prediction:     {knn_pred[0]}")
    print(f"\t -> Decision Tree Prediction: {tree_pred[0]}")
```

```
--------------------------------------
Test 1.1 (Easy)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           5.0          3.5           1.5          0.2

         -> KNN Model Prediction:     Iris-setosa
         -> Decision Tree Prediction: Iris-setosa

--------------------------------------
Test 1.2 (Easy)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           4.8          3.4           1.6          0.2

         -> KNN Model Prediction:     Iris-setosa
         -> Decision Tree Prediction: Iris-setosa

--------------------------------------
Test 1.3 (Easy)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           5.2          3.8           1.4          0.3

         -> KNN Model Prediction:     Iris-setosa
         -> Decision Tree Prediction: Iris-setosa

--------------------------------------
Test 2.1 (Ambiguous)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           6.5          3.0           5.0          1.6

         -> KNN Model Prediction:     Iris-virginica
         -> Decision Tree Prediction: Iris-versicolor

--------------------------------------
Test 2.2 (Ambiguous)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           6.7          3.1           5.2          1.8

         -> KNN Model Prediction:     Iris-virginica
         -> Decision Tree Prediction: Iris-virginica

--------------------------------------
Test 2.3 (Ambiguous)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           6.0          2.8           4.8          1.7
```

```
         -> KNN Model Prediction:     Iris-versicolor
         -> Decision Tree Prediction: Iris-versicolor

--------------------------------------
Test 3.1 (Outlier)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           7.5          2.5           7.0          0.5

         -> KNN Model Prediction:     Iris-virginica
         -> Decision Tree Prediction: Iris-versicolor

--------------------------------------
Test 3.2 (Outlier)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           5.0          2.0           2.5          2.0

         -> KNN Model Prediction:     Iris-setosa
         -> Decision Tree Prediction: Iris-setosa

--------------------------------------
Test 3.3 (Outlier)
--------------------------------------
Input Features:
   sepal_length  sepal_width  petal_length  petal_width
0           8.0          5.0           1.5          0.5

         -> KNN Model Prediction:     Iris-setosa
         -> Decision Tree Prediction: Iris-setosa
```

🎯 Key Outcome

- Both models perform with **high accuracy** on the Iris dataset.
- **KNN slightly outperforms Decision Tree** in this setup.