Name : Vaishnavi Eknath Avhad
Class : D15C
Roll No. : 41

Practical 5

Aim : To implement Classification algorithms (Decision Tree and Naïve Bayes algorithms) using Python.

Theory :

1. Classification Algorithms

Classification is a supervised machine learning technique used to categorize data into predefined labels. In this practical, we apply two commonly used classification algorithms: Decision Tree and Naïve Bayes.

---

2. Decision Tree Classifier

A decision tree splits the dataset into branches based on feature values, leading to a decision node (leaf) with a class label.

- Working : Uses criteria like Gini Index or Entropy to determine best splits.
- Advantages : Easy to interpret, supports both numerical and categorical data.
- Limitation : Prone to overfitting if the tree is too deep.

---

3. Naïve Bayes Classifier

Naïve Bayes is a probabilistic model based on Bayes' Theorem with the assumption that features are conditionally independent given the class.

$P(C|X)=P(X|C)\cdot P(C)/P(X)$

Types : GaussianNB, MultinomialNB, BernoulliNB

- Advantages : Fast, performs well on high-dimensional data.
- Limitation : Assumes feature independence, which might not always be true.

---

4. Evaluation Metrics

- Accuracy : Fraction of correctly predicted instances.
- Confusion Matrix : Shows TP, FP, FN, TN values.
- Classification Report : Includes precision, recall, F1-score.

---

5. Dataset Used

- Name : luxury_cosmetics_fraud_analysis_2025.csv
- Objective : Predict fraudulent transactions based on customer and transaction features.
- Target Variable : Fraud_Flag (0 = Non-Fraud, 1 = Fraud)

Code with output :

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier, plot_tree

from sklearn.naive_bayes import GaussianNB

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

import seaborn as sns

import matplotlib.pyplot as plt
```

```
# Load dataset

df = pd.read_csv("/content/airlines_flights_data.csv")

# Drop rows with missing values

df.dropna(inplace=True)

# Select features and target

X = df[['duration', 'days_left', 'price']]   # numerical predictors

y = df['class'].map({'Economy': 0, 'Business': 1})  # target encoding

# Train-test split

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

# ---- Decision Tree ----

dt = DecisionTreeClassifier(random_state=42)

dt.fit(X_train, y_train)

y_pred_dt = dt.predict(X_test)

# ---- Naive Bayes ----

nb = GaussianNB()

nb.fit(X_train, y_train)

y_pred_nb = nb.predict(X_test)

# ---- Evaluation ----

print("Decision Tree Accuracy:", accuracy_score(y_test, y_pred_dt))

print("Naive Bayes Accuracy:", accuracy_score(y_test, y_pred_nb))

# Define labels for classification report and confusion matrix
```

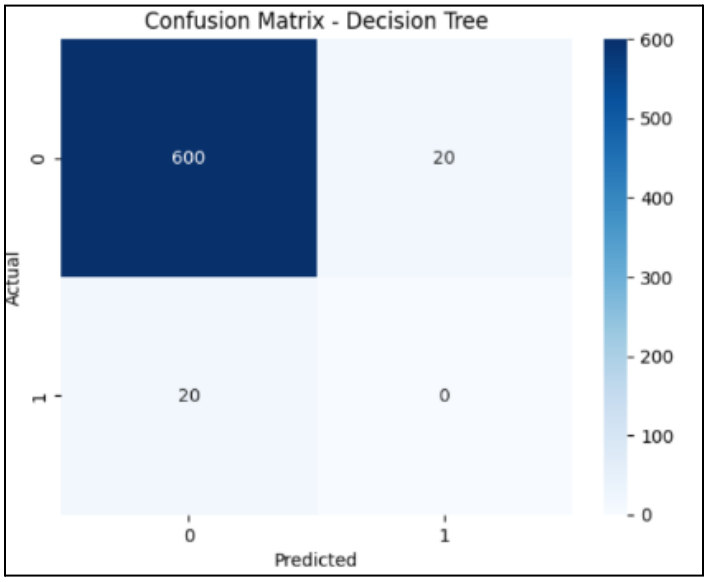
```
labels = [0] # Only 'Economy' class (encoded as 0) is present

print("\nClassification Report - Decision Tree\n", classification_report(y_test, y_pred_dt, labels=labels, zero_division=0))

print("\nClassification Report - Naive Bayes\n", classification_report(y_test, y_pred_nb, labels=labels, zero_division=0))

# Confusion Matrix for Decision Tree

cm_dt = confusion_matrix(y_test, y_pred_dt, labels=labels)

sns.heatmap(cm_dt, annot=True, fmt="d", cmap="Blues")

plt.title("Confusion Matrix - Decision Tree")

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.show()

# Confusion Matrix for Naive Bayes

cm_nb = confusion_matrix(y_test, y_pred_nb, labels=labels)

sns.heatmap(cm_nb, annot=True, fmt="d", cmap="Greens")

plt.title("Confusion Matrix - Naive Bayes")

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.show()

# Display Decision Tree

plt.figure(figsize=(20,10))
```

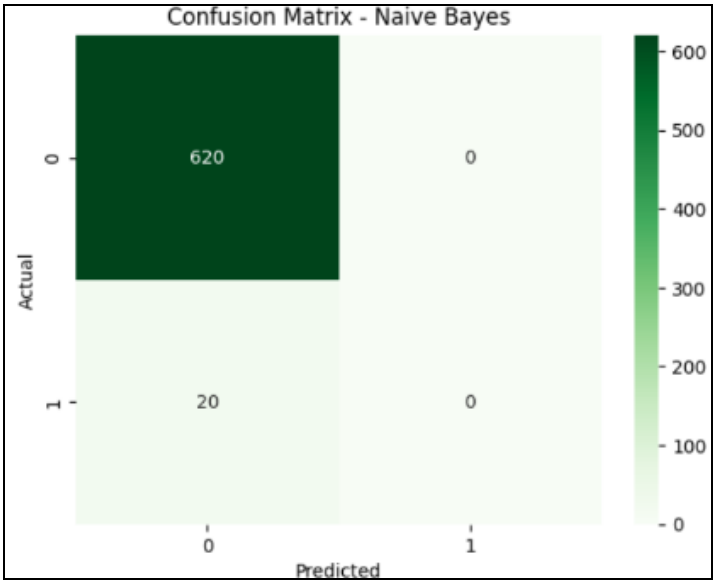
plot_tree(dt, feature_names=X.columns, class_names=['Economy', 'Business'], filled=True, rounded=True)

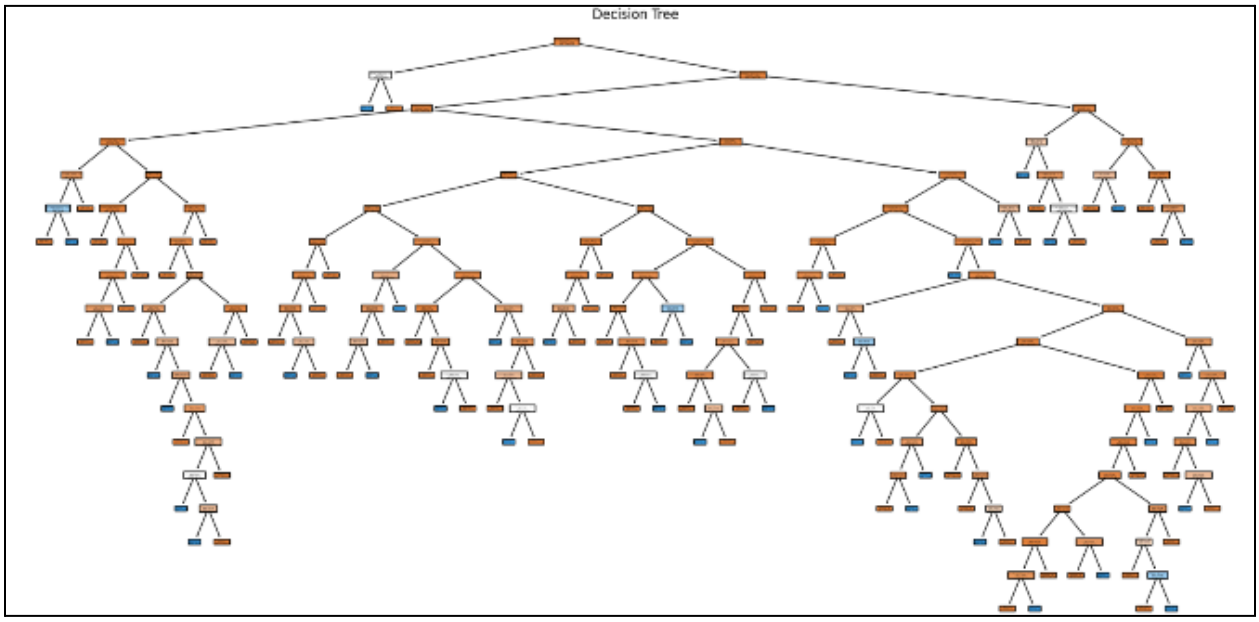
plt.title("Decision Tree")

plt.show()

```
Decision Tree Accuracy: 0.9375
Naive Bayes Accuracy: 0.96875

Classification Report - Decision Tree
              precision    recall  f1-score   support

           0       0.97      0.97      0.97       620
           1       0.00      0.00      0.00        20

    accuracy                           0.94       640
   macro avg       0.48      0.48      0.48       640
weighted avg       0.94      0.94      0.94       640


Classification Report - Naive Bayes
              precision    recall  f1-score   support

           0       0.97      1.00      0.98       620
           1       0.00      0.00      0.00        20

    accuracy                           0.97       640
   macro avg       0.48      0.50      0.49       640
weighted avg       0.94      0.97      0.95       640

/tmp/ipython-input-889793509.py:14: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplac
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead,


  df['Customer_Age'].fillna(df['Customer_Age'].mean(), inplace=True)
/tmp/ipython-input-889793509.py:15: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplac
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as
```







Conclusion :

In this practical, we implemented two classification algorithms — Decision Tree and Naïve Bayes — using Python. Both models were trained on a luxury fraud dataset and evaluated using standard metrics. Visualization of confusion matrices and decision tree structure provided insights into model performance and decision logic.