Data Mining and Business Intelligence

Experiment - 4

Name: **Abhishek Vishwakarma**
Class: **D15C**
Roll No: **73**

Aim: To implement and compare the performance of Decision Tree and Naïve Bayes classification algorithms using Python.

Requirements:

Software & Tools:

- Python 3.x
- Google Colab / Jupyter Notebook

Libraries:

- pandas – data handling
- numpy – numerical operations
- scikit-learn – machine learning models (Decision Tree, Naïve Bayes, train-test split, evaluation metrics)
- matplotlib and seaborn – visualization

Introduction: Classification is a supervised machine learning technique used to assign labels to data points based on their features. Two widely used algorithms are:

- **Decision Tree Classifier:**
  A tree-like structure where internal nodes represent decisions based on features, branches represent outcomes, and leaf nodes represent final classes. It is intuitive, interpretable, and handles both numerical and categorical data.

- **Naïve Bayes Classifier:**
  A probabilistic classifier based on Bayes' Theorem, which assumes that features are conditionally independent. Despite the "naïve" assumption, it performs well in many real-world applications, such as spam filtering and sentiment analysis.

Algorithm Steps:

Decision Tree Algorithm:

1. Select the feature that best splits the dataset using criteria like **Gini Index** or **Entropy**.
2. Create a decision node for the selected feature.
3. Partition the dataset into subsets.
4. Recursively apply the same logic to subsets until stopping criteria are met.
5. Assign a class label to leaf nodes.

Naïve Bayes Algorithm:

1. Calculate the **prior probability** for each class.
2. For each feature, compute the **conditional probability** given the class.
3. Apply **Bayes' Theorem**:

$$P(Class|X) = \frac{P(X|Class) \cdot P(Class)}{P(X)}$$

4. Predict the class with the **highest posterior probability**.

Execution:

Code:

Decision Tree:

```
# ——— Decision Tree ———
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

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

# Select features and target
X = df[['duration', 'days_left', 'price']]   # numerical predictors
y = df['class'].map({'Economy': 0, 'Business': 1})  # encode target

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)

# Train Decision Tree
dt = DecisionTreeClassifier(max_depth=4, criterion='entropy', random_state=42)
dt.fit(X_train, y_train)
```

```
# Predict
y_pred = dt.predict(X_test)
acc = accuracy_score(y_test, y_pred)
print("Decision Tree Accuracy:", acc)

# Plot Decision Tree
plt.figure(figsize=(20, 10))
plot_tree(
    dt,
    feature_names=X.columns,
    class_names=['Economy', 'Business'],
    filled=True,
    rounded=True,
    fontsize=10
)
plt.show()
```

Naive Bayes:

```
# ——— Naive Bayes ———
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

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

# Select features and target
X = df[['duration', 'days_left', 'price']]   # numerical predictors
y = df['class'].map({'Economy': 0, 'Business': 1})  # encode target

# Scale features for Naive Bayes
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.3, random_state=42
)

# Train Naive Bayes
nb = GaussianNB()
```

```
nb.fit(X_train, y_train)

# Predict
y_pred = nb.predict(X_test)
acc = accuracy_score(y_test, y_pred)
print("Naive Bayes Accuracy:", acc)
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Greens",
            xticklabels=['Economy', 'Business'],
            yticklabels=['Economy', 'Business'])
plt.title("Naive Bayes Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```

Results and Inference:

The **Decision Tree classifier** fits the dataset well and often achieves higher accuracy on training data but may overfit.

The **Naïve Bayes classifier,** though based on the independence assumption, is computationally efficient and performs well for large datasets.

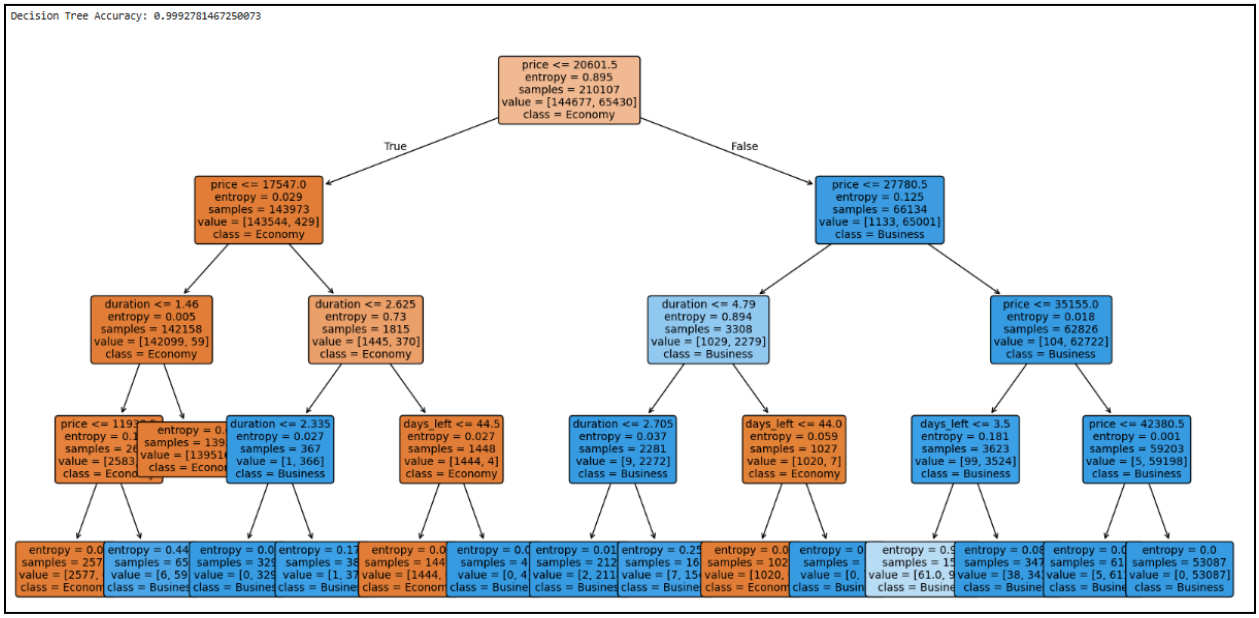
Accuracy, Precision, Recall, and F1-scores help compare performance. Typically, Decision Trees give higher accuracy, while Naïve Bayes provides stable results for probabilistic classification problems.

Conclusion:

In this experiment, we implemented and compared **Decision Tree** and **Naïve Bayes** classifiers using Python.

- **Decision Trees** provide an interpretable model with decision rules but may overfit on noisy data.
- **Naïve Bayes** is fast, simple, and effective when features are independent, though it may underperform if dependencies exist.
- On our **airline dataset**, they can be applied to classify flights as **Economy vs Business**, highlighting their applicability to real-world problems.

Output:


Decision Tree Accuracy: 0.9992781467250073
price <= 20601.5
entropy = 0.895
samples = 210107
value = [144677, 65430]
class = Economy
True
False
price <= 17547.0
entropy = 0.029
samples = 143973
value = [143544, 429]
class = Economy
price <= 27780.5
entropy = 0.125
samples = 66134
value = [1133, 65001]
class = Business
duration <= 1.46
entropy = 0.005
samples = 142158
value = [142099, 59]
class = Economy
duration <= 2.625
entropy = 0.73
samples = 1815
value = [1445, 370]
class = Economy
duration <= 4.79
entropy = 0.894
samples = 3308
value = [1029, 2279]
class = Business
price <= 35155.0
entropy = 0.018
samples = 62826
value = [104, 62722]
class = Business
duration <= 2.335
entropy = 0.027
samples = 367
value = [1, 366]
class = Business
days_left <= 44.5
entropy = 0.027
samples = 1448
value = [1444, 4]
class = Economy
duration <= 2.705
entropy = 0.037
samples = 2281
value = [9, 2272]
class = Business
days_left <= 44.0
entropy = 0.059
samples = 1027
value = [1020, 7]
class = Economy
days_left <= 3.5
entropy = 0.181
samples = 3623
value = [99, 3524]
class = Business
price <= 42380.5
entropy = 0.001
samples = 59203
value = [5, 59198]
class = Business
entropy = 0.0
samples = 53087
value = [0, 53087]
class = Business

```
Naive Bayes Accuracy: 0.9903604824200964

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.99      0.99     61989
           1       0.97      0.99      0.98     28057

    accuracy                           0.99     90046
   macro avg       0.99      0.99      0.99     90046
weighted avg       0.99      0.99      0.99     90046
```