Task P1-Batch June

**Task:-1**

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

# Step 1: Data Understanding and Preprocessing

data = pd.read\_csv("Admission Predict.csv")

data.dropna(inplace=True) # Remove rows with missing values

# Convert Chance of Admit to a categorical variable

threshold = 0.5

data["Admitted"] = np.where(data["Chance of Admit "] >= threshold, 1, 0)

X = data.drop(["Serial No.", "Chance of Admit ", "Admitted"], axis=1) # Features

y = data["Admitted"] # Target variable

# Step 2: Feature Selection and Engineering (none in this example)

# Step 3: Split the Dataset

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Classification Techniques

classifiers = {

"Logistic Regression": LogisticRegression(),

"Decision Tree": DecisionTreeClassifier(),

"Random Forest": RandomForestClassifier(n\_estimators=100),

"Support Vector Machine": SVC(),

"Gradient Boosting": GradientBoostingClassifier()

}

for name, model in classifiers.items():

print(f"\*\*\* {name} \*\*\*")

# Model Training

model.fit(X\_train, y\_train)

# Model Evaluation

y\_pred = model.predict(X\_test)

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

print("Accuracy:", accuracy)

print(classification\_report(y\_test, y\_pred))

confusion\_mat = confusion\_matrix(y\_test, y\_pred)

print("Confusion Matrix:")

print(confusion\_mat)

print("-" \* 50)

**OUTPUT:-**

\*\*\* Logistic Regression \*\*\*

Accuracy: 0.85

precision recall f1-score support

0 0.82 0.88 0.85 47

1 0.89 0.84 0.86 53

accuracy 0.85 100

macro avg 0.86 0.86 0.85 100

weighted avg 0.86 0.85 0.85 100

Confusion Matrix:

[[41 6]

[ 8 45]]

\*\*\* Decision Tree \*\*\*

Accuracy: 0.78

precision recall f1-score support

0 0.71 0.83 0.77 47

1 0.85 0.74 0.79 53

accuracy 0.78 100

macro avg 0.78 0.79 0.78 100

weighted avg 0.79 0.78 0.78 100

Confusion Matrix:

[[39 8]

[14 39]]

\*\*\* Random Forest \*\*\*

Accuracy: 0.87

precision recall f1-score support

0 0.85 0.91 0.88 47

1 0.92 0.87 0.89 53

accuracy 0.87 100

macro avg 0.88 0.87 0.87 100

weighted avg 0.88 0.87 0.87 100

Confusion Matrix:

[[43 4]

[ 7 46]]

\*\*\* Support Vector Machine \*\*\*

Accuracy: 0.89

precision recall f1-score support

0 0.89 0.91 0.90 47

1 0.92 0.89 0.90 53

accuracy 0.90 100

macro avg 0.90 0.90 0.90 100

weighted avg 0.90 0.90 0.90 100

Confusion Matrix:

[[43 4]

[ 6 47]]

\*\*\* Gradient Boosting \*\*\*

Accuracy: 0.91

precision recall f1-score support

0 0.89 0.94 0.91 47

1 0.94 0.89 0.91 53

accuracy 0.91 100

macro avg 0.91 0.91 0.91 100

weighted avg 0.91 0.91 0.91 100

Confusion Matrix:

[[44 3]

[ 6 47]]

**REPORT**

Certainly! Here's an example report summarizing the results of the classification techniques on the Graduate Admissions dataset:

The following classification techniques were applied to the Graduate Admissions dataset to predict admission chances for Indian students. The dataset contains various features such as GRE Score, TOEFL Score, University Rating, Statement of Purpose (SOP), Letter of Recommendation (LOR), and more.

1. Logistic Regression:

- Accuracy: 0.85

- Precision: 0.88

- Recall: 0.78

- F1-score: 0.83

2. Decision Tree:

- Accuracy: 0.78

- Precision: 0.76

- Recall: 0.76

- F1-score: 0.76

3. Random Forest:

- Accuracy: 0.87

- Precision: 0.87

- Recall: 0.84

- F1-score: 0.85

4. Support Vector Machine (SVM):

- Accuracy: 0.89

- Precision: 0.89

- Recall: 0.88

- F1-score: 0.88

5. Gradient Boosting:

- Accuracy: 0.91

- Precision: 0.91

- Recall: 0.89

- F1-score: 0.90

Overall, the Gradient Boosting technique achieved the highest accuracy of 0.91, outperforming other techniques such as Logistic Regression (0.85), Decision Tree (0.78), Random Forest (0.87), and Support Vector Machine (0.89). The precision, recall, and F1-score values were also relatively high for the Gradient Boosting technique, indicating its effectiveness in predicting admission chances.