Ankush Kumar

AP21110011026

CSE-P

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics

# Load the Diabetes dataset
url =
"https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-
indians-diabetes.data.csv"
names = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome']
diabetes_data = pd.read_csv(url, names=names)
print(diabetes_data)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI
0              6      148             72             35        0  33.6
\
1              1       85             66             29        0  26.6

2              8      183             64              0        0  23.3

3              1       89             66             23       94  28.1

4              0      137             40             35      168  43.1

..           ...      ...            ...            ...      ...   ...
763           10      101             76             48      180  32.9

764            2      122             70             27        0  36.8

765            5      121             72             23      112  26.2

766            1      126             60              0        0  30.1

767            1       93             70             31        0  30.4


     DiabetesPedigreeFunction  Age  Outcome
0                       0.627   50        1
```

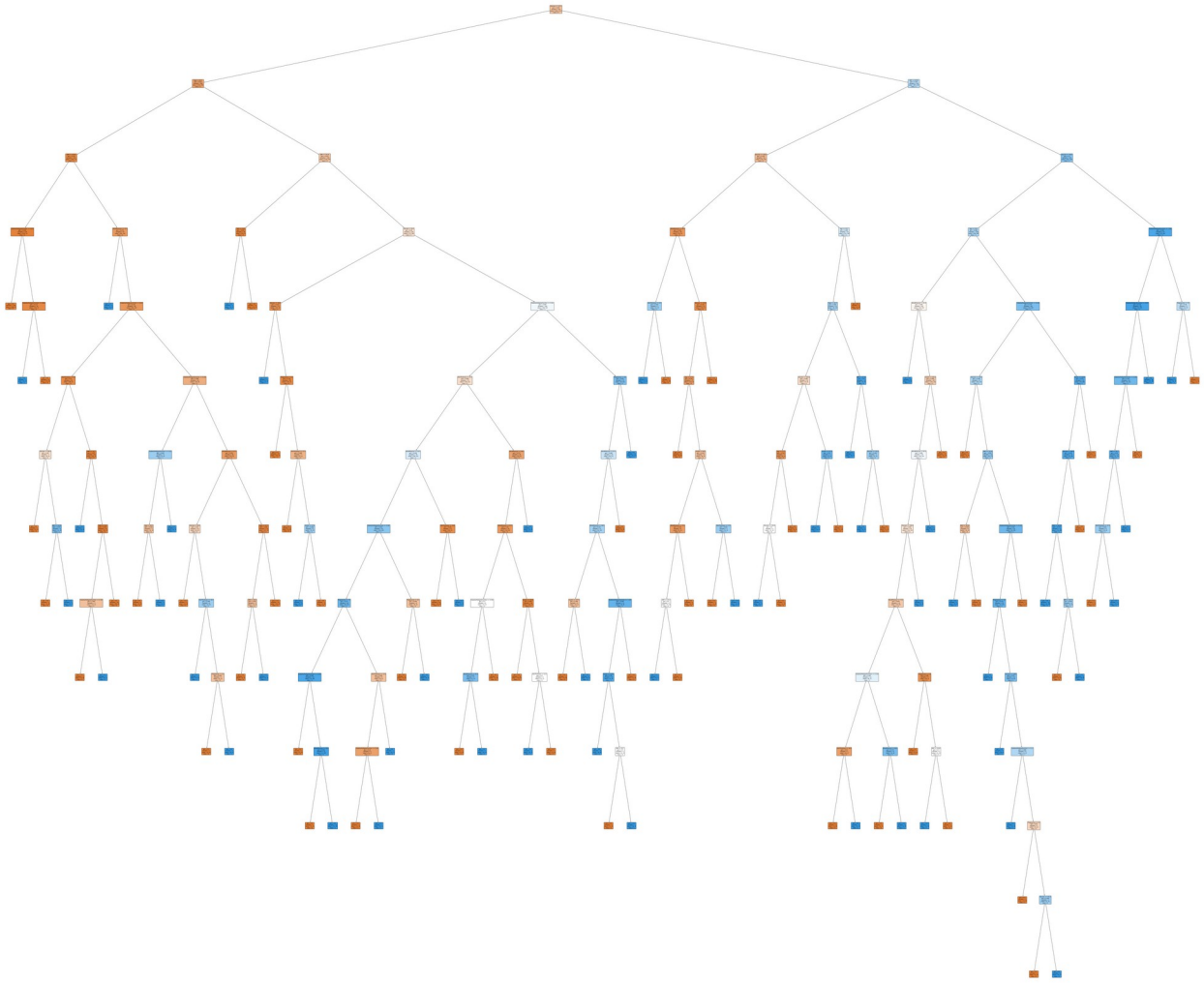
```
1                         0.351   31        0
2                         0.672   32        1
3                         0.167   21        0
4                         2.288   33        1
..                          ...  ...      ...
763                       0.171   63        0
764                       0.340   27        0
765                       0.245   30        0
766                       0.349   47        1
767                       0.315   23        0

[768 rows x 9 columns]
```

Decision Tree

```
# Split the dataset into features and target variable
X = diabetes_data.drop('Outcome', axis=1)
y = diabetes_data['Outcome']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Create a decision tree classifier
clf = DecisionTreeClassifier()

# Train the classifier using the training data
clf = clf.fit(X_train, y_train)

# Make predictions on the test data
y_pred = clf.predict(X_test)

# Evaluate the model
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.7402597402597403
```

```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

# Increase the size of the plot
plt.figure(figsize=(150, 130))
plot_tree(clf, feature_names=X.columns, class_names=['0', '1'],
filled=True, rounded=True)
plt.savefig('diabetes_tree.png')  # Save the decision tree as a PNG
file
```

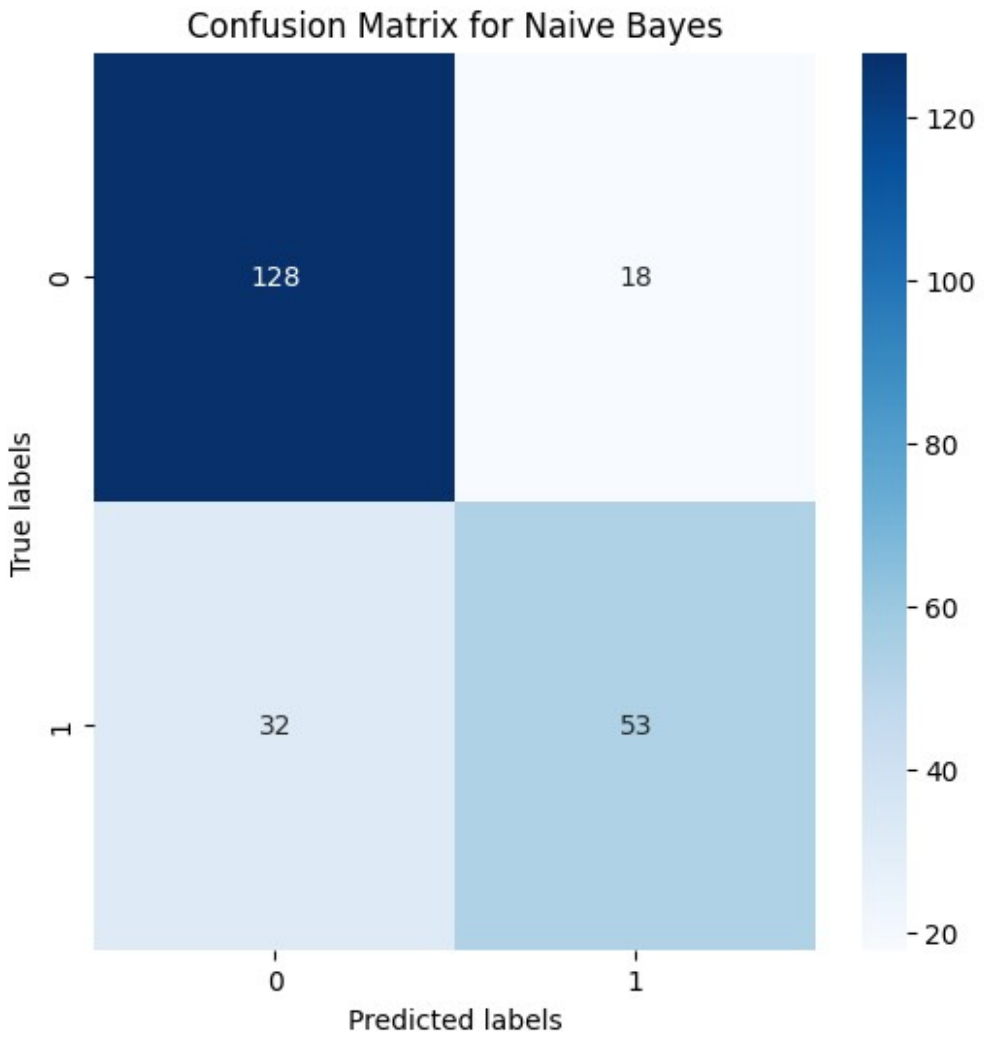


Naive Bayes

```
# Import necessary libraries
from sklearn.naive_bayes import GaussianNB

# Create a Gaussian Naive Bayes classifier
gnb = GaussianNB()

# Train the classifier using the training data
gnb = gnb.fit(X_train, y_train)

# Make predictions on the test data
y_pred_nb = gnb.predict(X_test)

# Evaluate the model
print("Accuracy for Naive Bayes:", metrics.accuracy_score(y_test,
y_pred_nb))
```

```
Accuracy for Naive Bayes: 0.7835497835497836
```

```
from sklearn.metrics import confusion_matrix, classification_report
import matplotlib.pyplot as plt
import seaborn as sns

# Generate confusion matrix
cm = confusion_matrix(y_test, y_pred_nb)

# Plot confusion matrix
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Predicted labels")
plt.ylabel("True labels")
plt.title("Confusion Matrix for Naive Bayes")
plt.show()

# Print classification report
print("Classification Report for Naive Bayes:")
print(classification_report(y_test, y_pred_nb))
```




```
Classification Report for Naive Bayes:
              precision    recall  f1-score   support

           0       0.80      0.88      0.84       146
           1       0.75      0.62      0.68        85

    accuracy                           0.78       231
   macro avg       0.77      0.75      0.76       231
weighted avg       0.78      0.78      0.78       231
```