Final Project: Diabetes Prediction from Medical Records

1. Project 1: Diabetes Prediction from Medical Records

The following findings and results are based on the machine learning analysis of the provided dataset i.e. Diabetes EHR data. The applied machine learning approach demonstrates the application of 5 different machine learning models based on classification methods which are *Random Forest, Naïve Bayes, Logistic Regression, Support Vector Machine, K-Nearest Neighbor.*

i. Data Exploration and Visualization

The provided EHR diabetic data from the National Institute of Diabetes and Digestive and Kidney Diseases consists information of female patients (aged ≥ 21) of Pima Indian Heritage. Upon initial inspection it was found that there are 550 entries (rows) and 10 variables (columns) in the dataset. The predictor variables are Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age, and Outcome, where Outcome has two classes 0 and 1, 0 for healthy and 1 for diabetic.

There are no missing data in this dataset as verified by following steps.

```
print(df.isnull().sum()) # Check missing values
```

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
Id                          0
dtype: int64
```

Some quick stats of these variables including mean, standard deviation, min and max is depicted below as a part of exploratory data analysis.

```
print(df.describe()) #Quick Stats Mean, Std, Min, Max
```

```
       Pregnancies     Glucose  BloodPressure  SkinThickness     Insulin  \
count   550.000000  550.000000     550.000000     550.000000  550.000000
mean      4.034545  121.560000      69.381818      20.014545   80.141818
std       3.447325   30.551206      19.036147      15.898006  115.429640
min       0.000000    0.000000       0.000000       0.000000    0.000000
25%       1.000000  100.000000      62.000000       0.000000    0.000000
50%       3.000000  119.000000      72.000000      22.000000   22.500000
75%       6.000000  141.000000      80.000000      32.000000  128.750000
max      17.000000  197.000000     122.000000      63.000000  846.000000

              BMI  DiabetesPedigreeFunction         Age     Outcome  \
count  550.000000                550.000000  550.000000  550.000000
mean    31.902000                  0.466582   33.590909    0.354545
std      7.822178                  0.320054   12.054140    0.478811
min      0.000000                  0.078000   21.000000    0.000000
25%     27.200000                  0.239250   24.000000    0.000000
50%     32.000000                  0.375000   29.000000    0.000000
75%     36.500000                  0.628250   41.000000    1.000000
max     59.400000                  2.420000   81.000000    1.000000

               Id
count  550.000000
mean   379.630909
std    222.127731
min      1.000000
25%    187.250000
50%    377.500000
75%    571.500000
max    766.000000
```

Some Visualization of these variables are also depicted below.

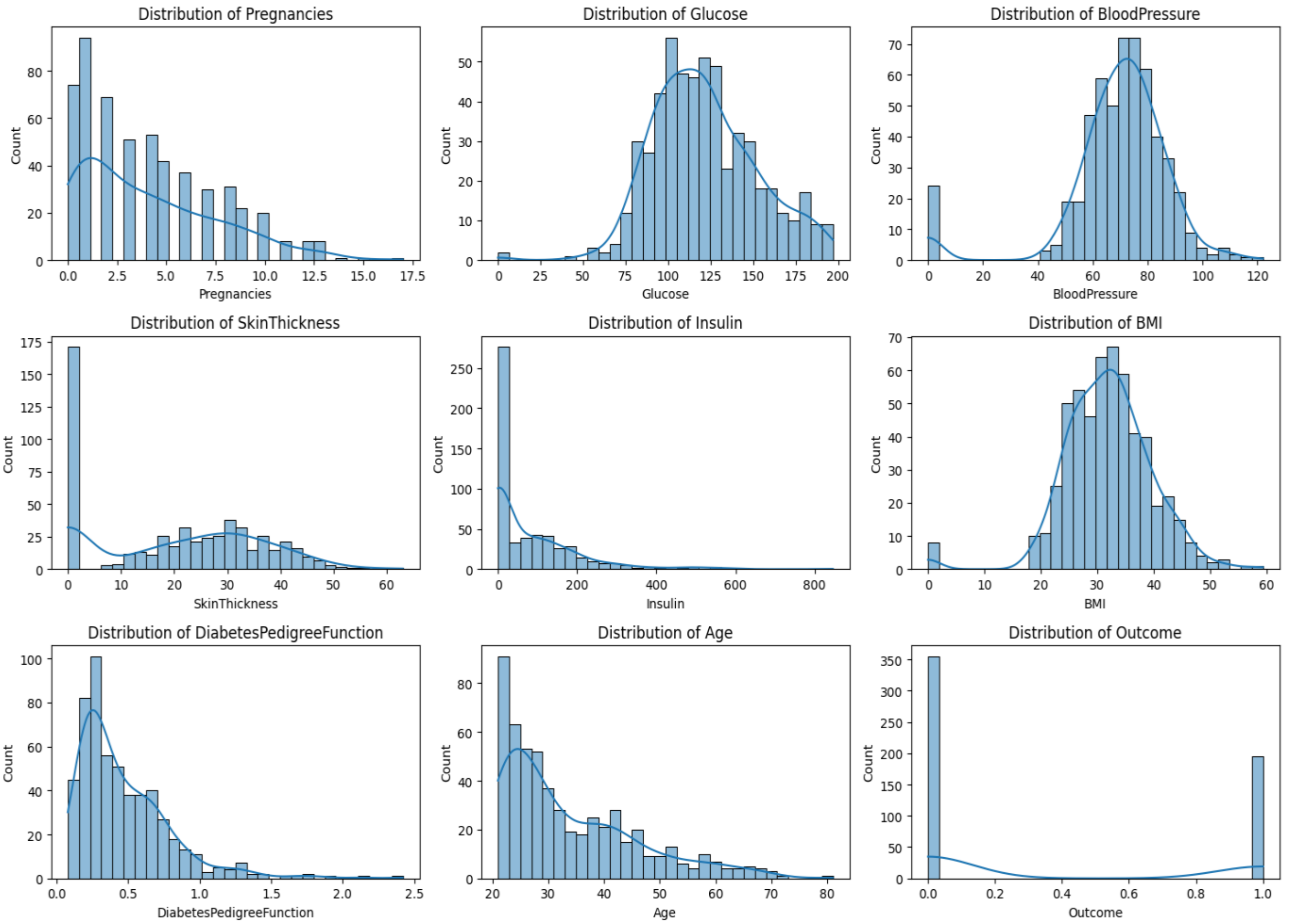

Fig 1: Distribution of Variables

The figures illustrated above provides the overview of variables distribution. Variables like *Glucose, BloodPressure, SkinThickness,* and *Outcome* have a bell-shaped formation indicating normal distribution. Rest of others follow slight left or right skewed but that is something to worry since it is expected with the medical nature of dataset.

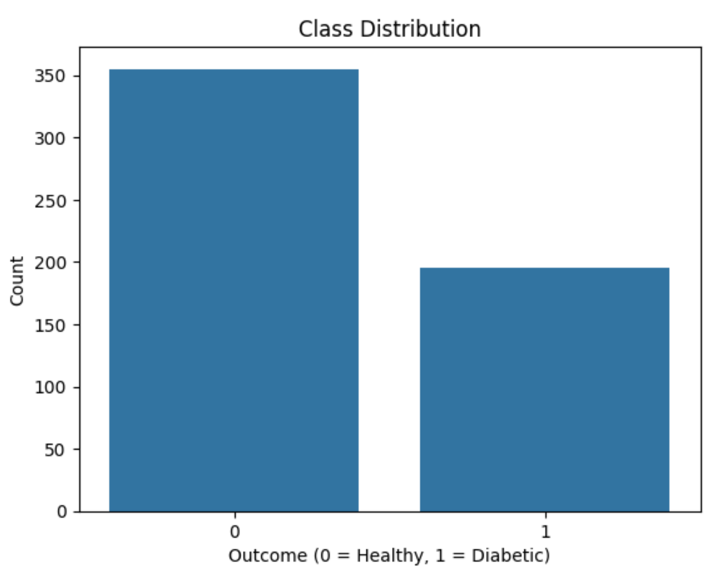

Fig 2: Plot of Class Imbalance

The Class imbalance plot above illustrates the count of 0 (healthy) and 1 (diabetic) cases in the dataset. It can be seen that there are 350 cases of 0 but only 200 cases of 1 which represents a slight imbalance but we can address this during the evaluation process.

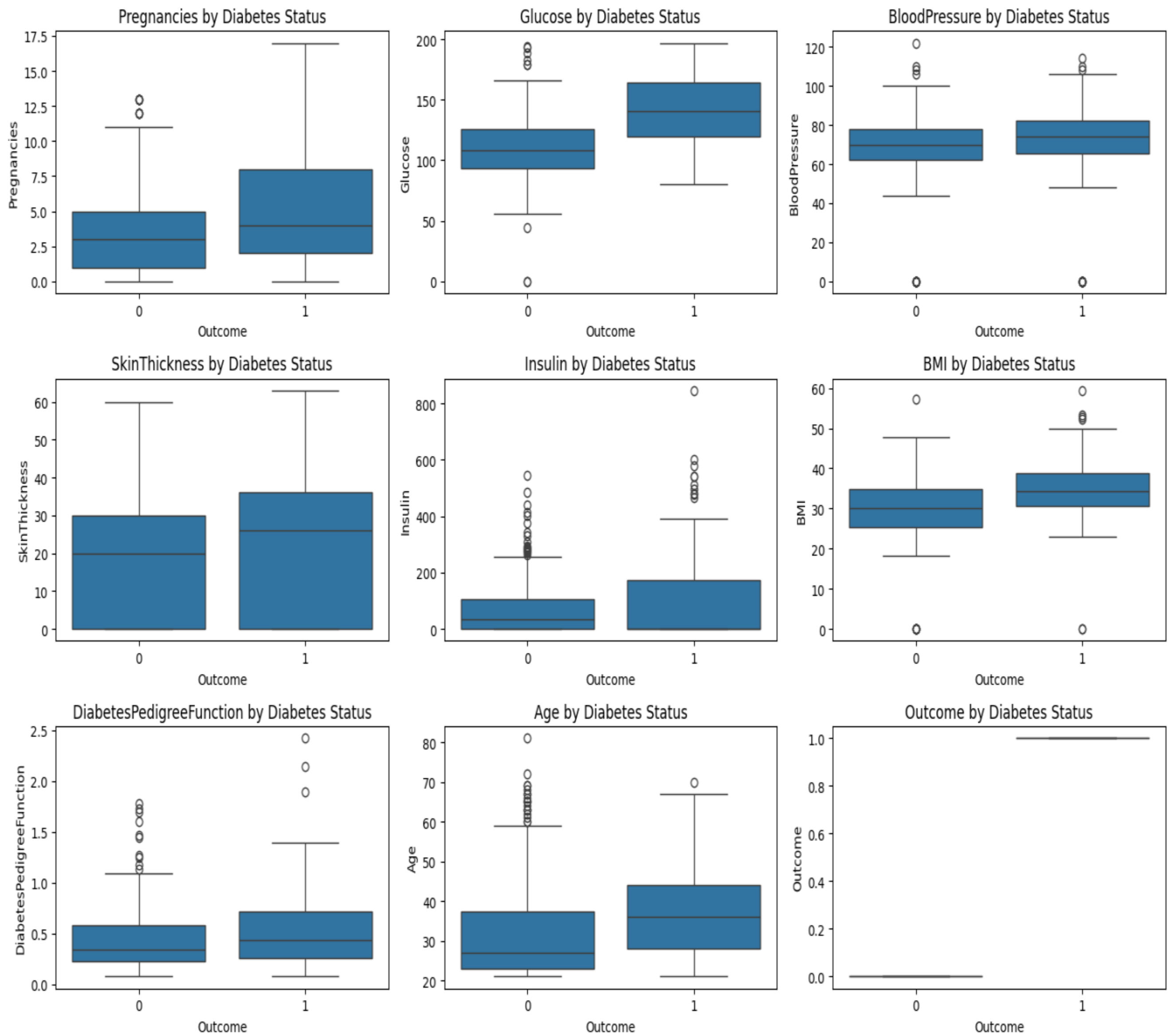

Fig 3: Box Plot of Variables

The box plot of variables above provides the snapshot of presence of any outliers within the dataset but so far, our data looks normal there is no extreme cases of outlier which can alter our findings.

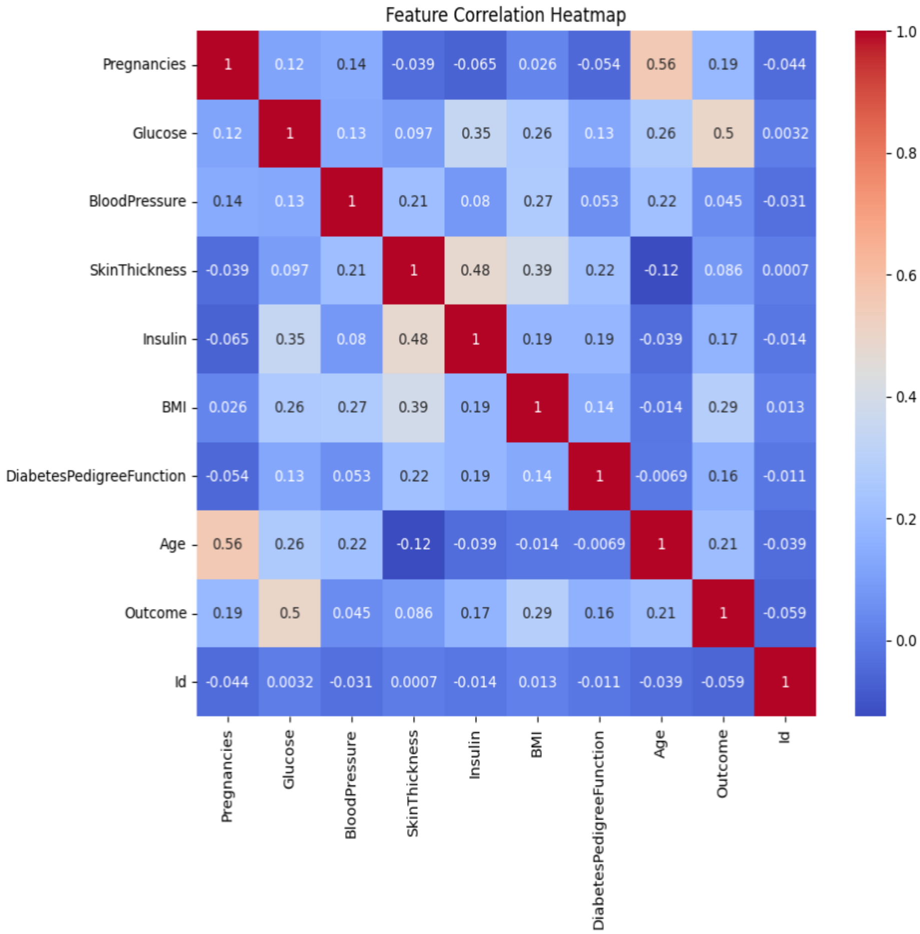

Fig 4: Correlation Heatmap of the Variables

The Correlation heatmap above suggest that Glucose (0.5) have the highest or strongest positive correlation with the target variable outcome meaning higher glucose level are moderately associated with having diabetes. BMI and Age also suggest positive but small correlation. Since we are using a comprehensive approach for this analysis, we will be using all relevant variables as these medical biomarkers are indeed relevant for our study.

ii. 5 Machine Learning Models

To carry out with analysis, 5 machine learning classification methods were used; they are.

1. **Support Vector Machine (SVM)**
2. **K-Nearest Neighbor (KNN)**
3. **Random Forest**
4. **Naïve Bayes**
5. **Logistic Regression**

```
# Initialize models
models = {
    'Support Vector Machine': SVC(kernel='rbf', random_state=42),
    'Random Forest': RandomForestClassifier(n_estimators=100, random_state=42),
    'Logistic Regression': LogisticRegression(max_iter=1000, random_state=42),
    'K-Nearest Neighbors': KNeighborsClassifier(),
    'Naive Bayes': GaussianNB()
}
```

iii. Averaged Results using 10-fold Cross Validation

A 10-fold Stratified Cross Validation was used in order to address the slight imbalance that was encountered. The results of the 10-fold cross validation are illustrated in the table below.

```python
# 10-fold cross-validation setup
cv = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)

# Metrics to evaluate
metrics = ['accuracy', 'f1', 'precision', 'recall']

# Evaluate models
results = []
for name, model in models.items():
    scores = {m: cross_val_score(model, X_scaled, y, cv=cv, scoring=m).mean() for m in metrics}
    results.append([name, scores['accuracy'], scores['f1'], scores['precision'], scores['recall']])

# Create and display results DataFrame
results_df = pd.DataFrame(results, columns=['Model', 'Accuracy', 'F1-Score', 'Precision', 'Recall'])
results_df = results_df.sort_values('F1-Score', ascending=False).reset_index(drop=True)
```

iv. Tables and Figures of Classification F1-Scores and Accuracies

Model	Accuracy	F1-Score	Precision	Recall
Naive Bayes	0.7636	0.6455	0.6968	0.6108
Random Forest	0.7618	0.6417	0.7005	0.6100
Logistic Regression	0.7600	0.6204	0.7098	0.5592
Support Vector Machine (SVM)	0.7509	0.6057	0.6876	0.5487
K-Nearest Neighbors (KNN)	0.7327	0.5952	0.6412	0.5629

Table 1: Performance Comparison of Models

From the table above, we have accuracy, F1-score, precision and recall for all the models. These scores are further demonstrated graphically below to provide a visual overview.

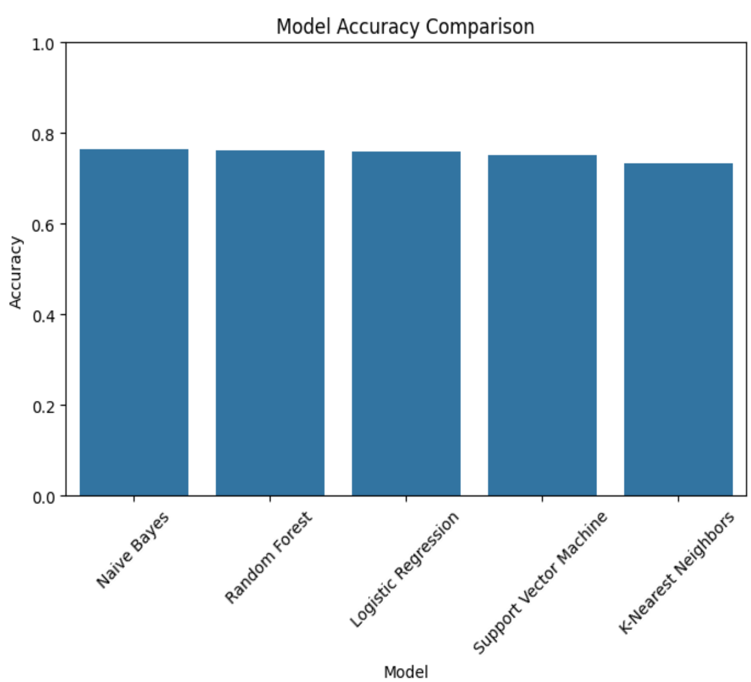

Fig 5: Bar Plot of Accuracy Scores of all Models

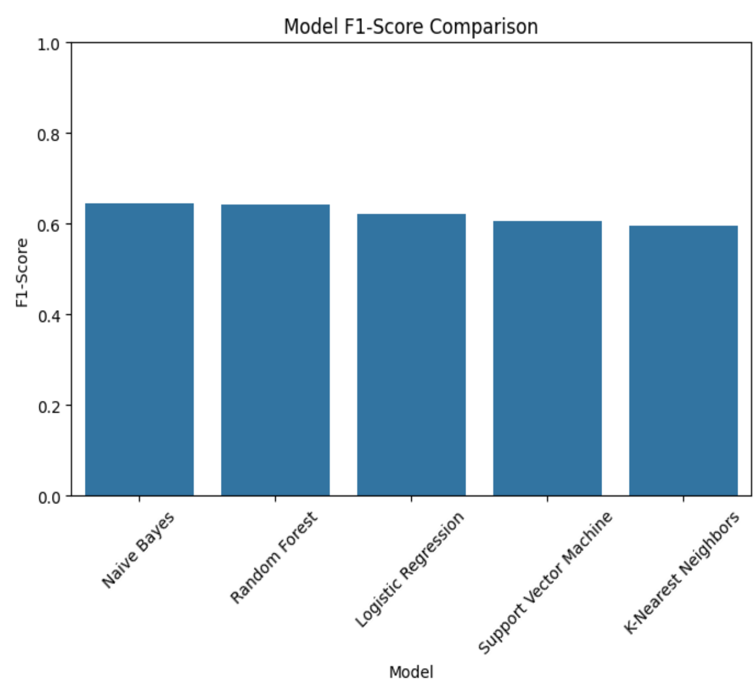

Fig 6: Bar Plot of F1-Scores of all Models

v. Conclusion and Discussion on the best Classification method and Classification Report

```
print("Classification Reports (trained on full data):\n")

for name, model in models.items():
    # Fit model on full data
    model.fit(X, y)

    # Predict on full data
    y_pred = model.predict(X)

    # Print classification report
    print(f"Model: {name}")
    print(classification_report(y, y_pred, digits=4))
    print("-" * 60)
```

```
Model: Support Vector Machine
              precision    recall  f1-score   support

           0     0.7183    0.9408    0.8146       355
           1     0.7529    0.3282    0.4571       195

    accuracy                         0.7236       550
   macro avg     0.7356    0.6345    0.6359       550
weighted avg     0.7306    0.7236    0.6879       550

------------------------------------------------------------
Model: Random Forest
              precision    recall  f1-score   support

           0     1.0000    1.0000    1.0000       355
           1     1.0000    1.0000    1.0000       195

    accuracy                         1.0000       550
   macro avg     1.0000    1.0000    1.0000       550
weighted avg     1.0000    1.0000    1.0000       550

------------------------------------------------------------
Model: Logistic Regression
              precision    recall  f1-score   support

           0     0.7908    0.8732    0.8300       355
           1     0.7152    0.5795    0.6402       195

    accuracy                         0.7691       550
   macro avg     0.7530    0.7264    0.7351       550
weighted avg     0.7640    0.7691    0.7627       550

------------------------------------------------------------
Model: K-Nearest Neighbors
              precision    recall  f1-score   support

           0     0.8056    0.8873    0.8445       355
           1     0.7484    0.6103    0.6723       195

    accuracy                         0.7891       550
   macro avg     0.7770    0.7488    0.7584       550
weighted avg     0.7853    0.7891    0.7835       550

------------------------------------------------------------
Model: Naive Bayes
              precision    recall  f1-score   support

           0     0.8042    0.8563    0.8295       355
           1     0.7035    0.6205    0.6594       195

    accuracy                         0.7727       550
   macro avg     0.7539    0.7384    0.7444       550
weighted avg     0.7685    0.7727    0.7692       550
```

Model	Accuracy	F1-Score (Weighted Avg)	Notes
Random Forest	**1.0000**	**1.0000**	**Perfect scores but highly suspicious of overfitting!**
KNN	**0.7891**	**0.7835**	**Good performance**
Logistic Regression	**0.7691**	**0.7627**	**Decent, balanced**
Naive Bayes	**0.7727**	**0.7692**	**Decent, balanced**
SM	**0.7236**	**0.6879**	**Lowest performance**

Table 2: Classification Reports

Considering all the results from accuracy, F1 scores, classification report, etc. the best model for our data is the Naïve Bayes. It has an accuracy of 0.7636 and a F1 score with 0.6455 which in comparison is a better number given other models and has a balanced fit.