MACHINE LEARNING MINI PROJECT

Title: Predicting Titanic Survival: A Machine Learning Approach

Group Members:
- Tanay Jadhav (29)
- Ritesh Shevare (30)
- Om Taskar (31)
- Pratik Kotkar (32)

Problem Statement:

Build a machine learning model that predicts the likelihood of a person surviving the Titanic shipwreck using passenger data such as name, age, gender, and socio-economic class. The goal is to classify passengers as survivors or non-survivors based on relevant features in the dataset.

Objectives:

- To predict the survival status of Titanic passengers based on historical data.
- To identify the key features influencing survival rates on the Titanic.
- To build machine learning models with high accuracy for classification.
- To evaluate model performance using metrics such as accuracy, precision, recall, and F1 score.

Introduction:

The sinking of the RMS Titanic in 1912 led to one of the deadliest maritime disasters, with over 1500 fatalities. Factors such as age, gender, and socio-economic class significantly influenced survival rates. Leveraging machine learning, this project aims to predict which passengers survived the disaster using a dataset that contains passenger details. Two models, Logistic Regression and

Naive Bayes, are used to classify passengers as survivors or non-survivors.

Dataset Description:

The dataset used for this project was obtained from the Kaggle Titanic Competition. The dataset contains the following features:

- PassengerId: Unique ID for each passenger.
- Survived: 0 = No, 1 = Yes (Target Variable).
- Pclass: Passenger class (1 = Upper, 2 = Middle, 3 = Lower).
- Name: Name of the passenger.
- Sex: Gender of the passenger.
- Age: Age of the passenger.
- SibSp: Number of siblings or spouses aboard.
- Parch: Number of parents or children aboard.
- Ticket: Ticket number.
- Fare: Passenger fare.
- Cabin: Cabin number.
- Embarked: Port of embarkation (C = Cherbourg, Q = Queenstown, S = Southampton).

Analysis Steps:

1. Data Exploration and Cleaning: Handle missing values, outliers, and irrelevant features.
2. Feature Engineering: Convert categorical variables into numerical form, and create additional features where necessary.
3. Model Selection: Implement classification models such as Logistic Regression and Naive Bayes.
4. Model Training: Train the models on the Titanic dataset.
5. Model Evaluation: Evaluate the performance of both models using metrics such as accuracy, precision, recall, and F1 score.
6. Hyperparameter Tuning: Adjust model parameters to improve accuracy and

performance.

Libraries Used:

- Pandas: For data manipulation and cleaning.
- NumPy: For numerical operations.
- Matplotlib/Seaborn: For data visualization.
- Scikit-learn: For machine learning models and evaluation metrics.

Algorithm / Program:

Importing Required Libraries:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report,
ConfusionMatrixDisplay
```

Data Cleaning and Feature Engineering:

```
df = pd.read_csv("train.csv")
df.drop(['PassengerId', 'Name', 'Ticket', 'Cabin', 'Embarked'], axis=1,
inplace=True)
# Fill missing values
def fill_age(data):
    age, sex = data[0], data[1]
    return 29 if sex == 'male' and pd.isnull(age) else (25 if sex == 'female' and
```

```
pd.isnull(age) else age)
df['Age'] = df[['Age', 'Sex']].apply(fill_age, axis=1)
df['Sex'] = pd.get_dummies(df['Sex'], drop_first=True)
X = df.drop('Survived', axis=1).values
y = df['Survived'].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
random_state=10)
```

Modeling and Evaluation:

Logistic Regression:

```
logistic_regression = LogisticRegression(random_state=0)
logistic_regression.fit(X_train, y_train)
y_pred_log = logistic_regression.predict(X_test)
cm_log = confusion_matrix(y_test, y_pred_log)
ConfusionMatrixDisplay(confusion_matrix=cm_log).plot()
plt.show()
print(classification_report(y_test, y_pred_log))
```

Naive Bayes:

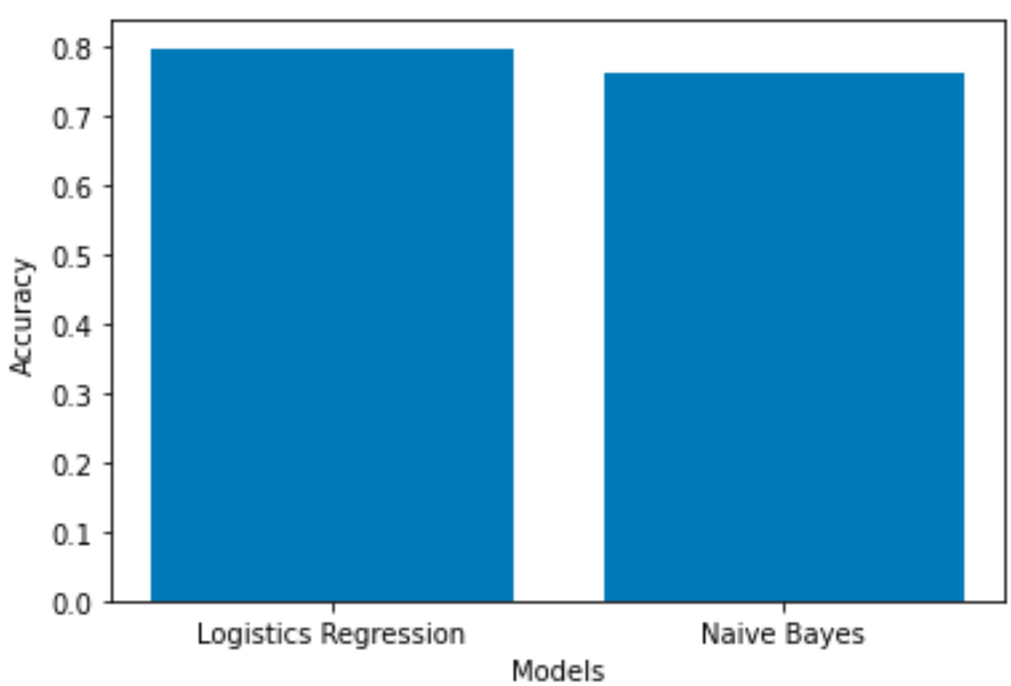
```
naive_bayes = GaussianNB()
naive_bayes.fit(X_train, y_train)
y_pred_nb = naive_bayes.predict(X_test)
cm_nb = confusion_matrix(y_test, y_pred_nb)
ConfusionMatrixDisplay(confusion_matrix=cm_nb).plot()
plt.show()
print(classification_report(y_test, y_pred_nb))
```

Comparative Study:

```
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score
models = ["Logistic Regression", "Naive Bayes"]
accuracy = [accuracy_score(y_test, y_pred_log), accuracy_score(y_test,
y_pred_nb)]
precision = [precision_score(y_test, y_pred_log), precision_score(y_test,
y_pred_nb)]
recall = [recall_score(y_test, y_pred_log), recall_score(y_test, y_pred_nb)]
f1 = [f1_score(y_test, y_pred_log), f1_score(y_test, y_pred_nb)]
error_rate = [1 - acc for acc in accuracy]
```

Results:

- Logistic Regression: 80% accuracy
- Naive Bayes: 76% accuracy



Conclusion:

In this project, we successfully built machine learning models to predict the survival of Titanic passengers. By implementing Logistic Regression and Naive Bayes, we achieved 80% and 76% accuracy, respectively. Logistic Regression outperformed Naive Bayes in terms of accuracy, precision, and F1 score. The comparative analysis highlights the importance of model selection in classification tasks.