Assessment Report

on

"Titanic Survival Prediction"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

GROUP 6

Palak Tiwari 202401100300166

Radhika 202401100300190

Roushan Kumar 202401100300204

Rudransh Mishra 202401100300205

Rudrika Singhal 202401100300206

Section: C

Under the supervision of

"Mr. Mayank Lakhotia"

KIET Group of Institutions, Ghaziabad

27 May, 2025

1. Introduction

The goal of this project is to develop a classification model that predicts whether a passenger survived the Titanic disaster based on features such as age, sex, passenger class, and other attributes. The Naive Bayes classifier, a probabilistic machine learning algorithm, is used due to its simplicity and effectiveness for classification tasks with independent features.

2. Problem Statement

Predicting which passengers survived based on available characteristics can provide insights into the factors that influenced survival outcomes. The challenge is to develop an accurate classification model that predicts passenger survival using demographic and travel information such as age, sex, passenger class, fare, and embarkation port.

This project aims to build a Naive Bayes classification model that leverages cleaned and preprocessed Titanic passenger data to predict survival status. Key tasks include handling missing data, encoding categorical variables, and evaluating model performance on unseen data. The ultimate goal is to understand the significant predictors of survival and create a reliable predictive tool for this historical dataset.

3. Objectives

- Clean and preprocess the Titanic dataset to handle missing and categorical data.
- Explore the data to identify key factors influencing survival.
- Build a Naive Bayes classification model to predict survival.
- Evaluate model performance using accuracy and classification metrics.
- Generate final survival predictions for unseen test data.

4. Methodology

Importing Libraries
The project begins by importing essential Python libraries:

- pandas and numpy for data manipulation and numerical operations.
- seaborn and matplotlib for visualizing the data.
- Scikit-learn modules for data preprocessing, model building, evaluation, and splitting.

Loading the Dataset
The Titanic training and test datasets are loaded from CSV files into pandas DataFrames for further analysis.

Exploratory Data Analysis (EDA)
Visualizations are created to understand the distribution of the target variable (Survived) and how survival relates to factors such as sex, passenger class (Pclass), and age. This helps in gaining insights into which features are important.

Data Cleaning and Preprocessing

- Irrelevant or redundant columns (PassengerId, Name, Ticket, Cabin) are dropped to simplify the dataset.
- Missing values in the Age column are filled with the median age to maintain data integrity.
- Missing Embarked values are replaced with the most common port of embarkation.
- Categorical features like Sex and Embarked are converted into numerical values using LabelEncoder, making them suitable for model training.

Feature and Label Separation
The dataset is divided into features (X) and the target variable (y), where Survived is the label to predict.

Train-Test Split

The training data is split into training and validation subsets using an 80:20 ratio to evaluate the model's performance on unseen data.

Model Training

A Gaussian Naive Bayes classifier is instantiated and trained on the training data (X_train, y_train). This probabilistic model assumes feature independence and models continuous features using Gaussian distributions.

Model Evaluation

Predictions are made on the validation set, and the model's accuracy is calculated. A detailed classification report is generated to evaluate precision, recall, and F1-score. The confusion matrix is plotted to visualize the model's performance in correctly and incorrectly classifying survivors and non-survivors.

Test Data Preparation and Prediction

The test dataset is cleaned similarly to the training data: missing values are filled, irrelevant columns dropped, and categorical variables encoded. The trained model then predicts survival outcomes for the test data.

Saving Final Predictions

The predictions for the test set are compiled along with passenger IDs into a DataFrame and saved to a CSV file (final_predictions.csv) for submission or further analysis.

Conclusion

The project concludes with a summary highlighting the model's accuracy (~77%) and insights derived from data visualizations (e.g., females and first-class passengers had higher survival rates).

5. Data Preprocessing

Data processing here involves:

- Cleaning missing data through median or mode imputation.
- Removing irrelevant or noisy columns.
- Encoding categorical variables into numerical form for the model.
- Preparing separate but consistent training and test datasets.

6. Model Implementation

- Imported necessary libraries for data handling, visualization, preprocessing, modelling, and evaluation.
- Loaded training and test datasets.
- Performed exploratory data analysis (EDA) to understand feature distributions and their relation to survival.
- Cleaned data by dropping irrelevant columns and imputing missing values in Age and Embarked.
- Encoded categorical features (Sex, Embarked) into numerical values using Label Encoding.
- Separated features (X) and target (y), then split data into training (80%) and validation (20%) sets.
- Trained a Gaussian Naive Bayes classifier on the training data.
- Evaluated the model on the validation set using accuracy, classification report, and confusion matrix.
- Applied the same preprocessing steps to the test dataset.
- Predicted survival on the test set using the trained model.
- Saved final predictions along with PassengerId to a CSV file.
- Achieved approximately 77% accuracy; findings showed females and higher-class passengers had better survival chances.

7. Evaluation Metrics

- **Accuracy:** Approximately 77% — the model correctly predicts survival status for around 77% of passengers in the validation set.
- **Classification Report:** Provides precision, recall, and F1-scores that help analyze model balance between false positives and false negatives.
- **Confusion Matrix:** Visual representation to diagnose classification errors.

8. Results and Analysis

- ✓ **The Naive Bayes model achieved approximately 77% accuracy on the validation set, indicating a reasonably good fit given the dataset and model simplicity.**
- ✓ **Visual analysis confirmed well-known trends: females and passengers in higher classes had better survival chances.**
- ✓ **The confusion matrix revealed the model performed well in identifying survivors and non-survivors, though some misclassifications occurred (as expected with this dataset).**
- ✓ **Final predictions were generated for the test data and saved successfully**

9. Conclusion

- ✓ The project successfully built a Naive Bayes classifier to predict Titanic passenger survival.
- ✓ Data cleaning and feature encoding were crucial steps to prepare the dataset for modeling.
- ✓ Exploratory visualizations helped validate assumptions about survival factors.
- ✓ While the model's performance is decent, further improvements could include:

- ✓ More advanced feature engineering (e.g., titles from names, family size).

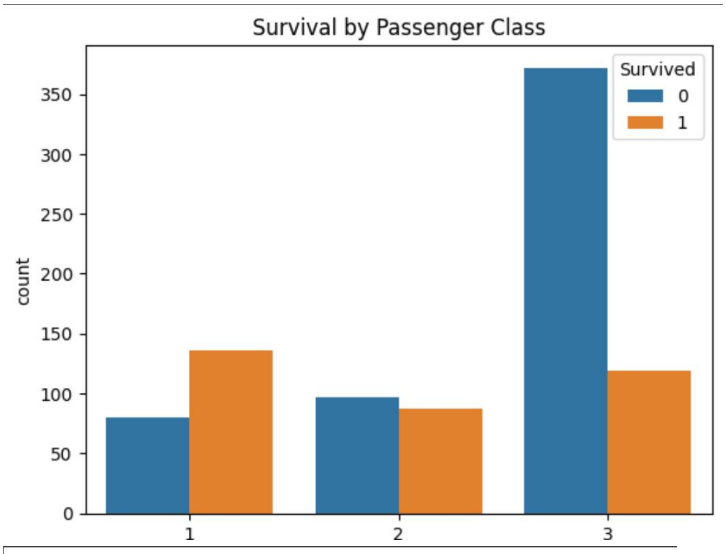
- ✓ Trying other classification models (Random Forest, Logistic Regression, SVM).

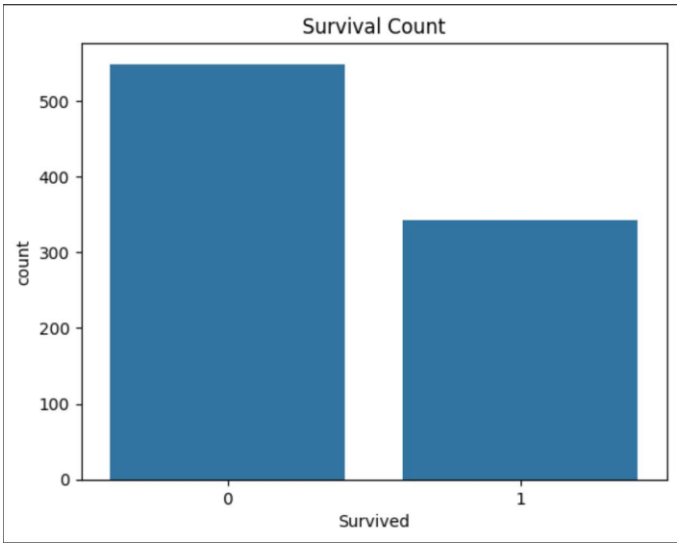
- ✓ Hyperparameter tuning or ensemble methods to boost accuracy.

- ✓ The approach demonstrates the power of simple probabilistic models in binary classification problems with structured data.

10. References

- Kaggle Titanic Dataset
- Scikit-learn Gaussian Naive Bayes
- Pandas Library
- Seaborn Visualization
- Scikit-learn Label Encoding
- Scikit-learn Train-Test
- Scikit-learn Evaluation Metrics



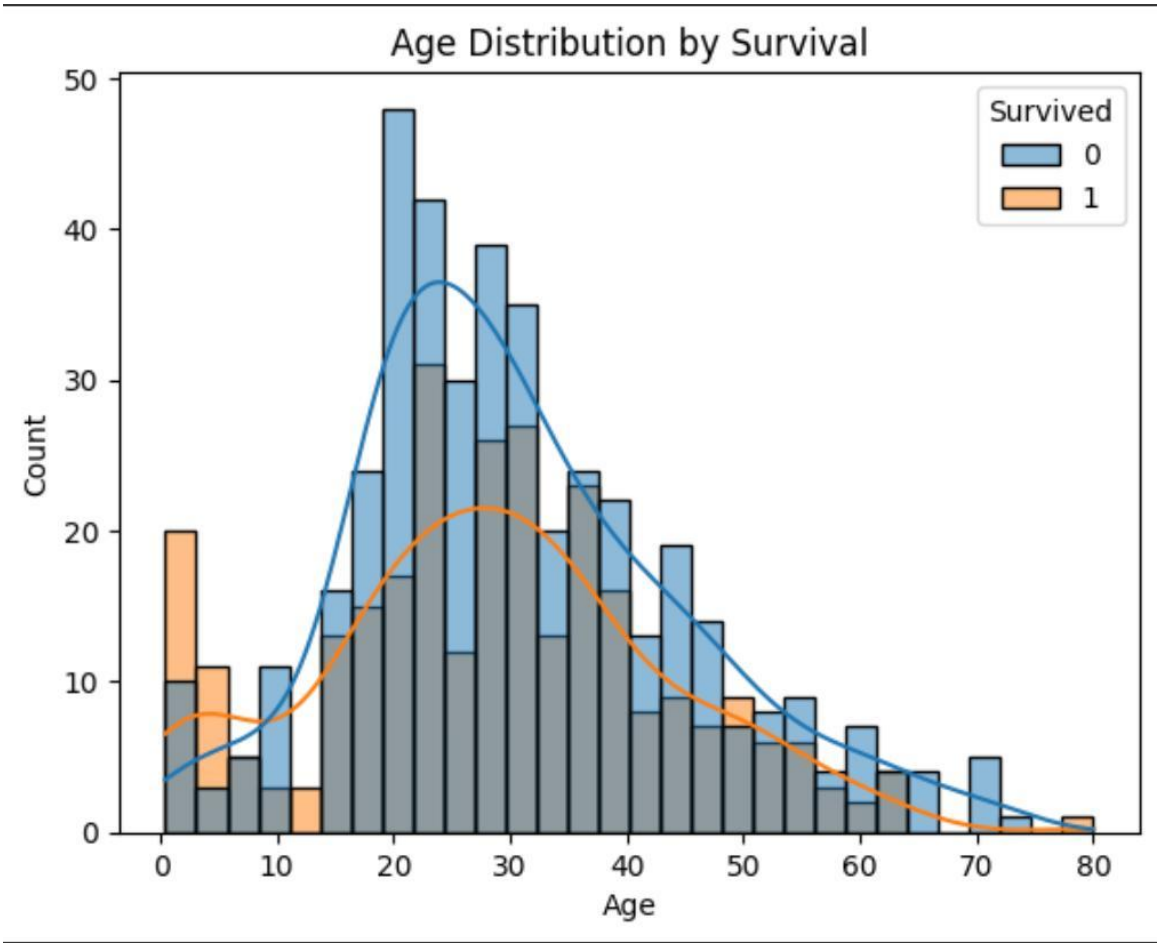



```
✅ Final predictions saved to final_predictions.csv
   PassengerId  Survived
0          892         0
1          893         1
2          894         0
3          895         0
4          896         1
```

```
✅ Accuracy: 0.776536312849162
✅ Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.78      0.80       105
           1       0.71      0.77      0.74        74

    accuracy                           0.78       179
   macro avg       0.77      0.78      0.77       179
weighted avg       0.78      0.78      0.78       179
```



```python
# Titanic Survival Prediction using Naive Bayes

#  1. Importing Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report, ConfusionMatrixDisp

#  2. Load the Datasets
train_df = pd.read_csv('/content/train.csv')
test_df = pd.read_csv('/content/test.csv')

#  3. Exploratory Data Analysis (EDA)
sns.countplot(x='Survived', data=train_df)
plt.title('Survival Count')
plt.show()

print("\n")
print("\n")

sns.countplot(x='Sex', hue='Survived', data=train_df)
plt.title('Survival by Sex')
plt.show()
```

```python
sns.countplot(x='Pclass', hue='Survived', data=
plt.title('Survival by Passenger Class')
plt.show()

print("\n")
print("\n")

sns.histplot(data=train_df, x='Age', kde=True, hue='Survived', bins=30)
plt.title('Age Distribution by Survival')
plt.show()

print("\n")
print("\n")

#  4. Data Cleaning on train.csv
train_df.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1, inplace=True)
train_df['Age'] = train_df['Age'].fillna(train_df['Age'].median())
train_df['Embarked'] = train_df['Embarked'].fillna(train_df['Embarked'].mode()[0])

le_sex = LabelEncoder()
train_df['Sex'] = le_sex.fit_transform(train_df['Sex'])

le_embarked = LabelEncoder()
train_df['Embarked'] = le_embarked.fit_transform(train_df['Embarked'])

#  5. Feature and Label Separation
X = train_df.drop('Survived', axis=1)
y = train_df['Survived']
```

```
# 6. Train-Test Split
X_train, X_valid, y_train, y_valid = train_test_split(X, y, test_size=0.2, rando

# 7. Model Training
model = GaussianNB()
model.fit(X_train, y_train)

# 8. Evaluation on Validation Set
y_pred = model.predict(X_valid)

print("✅ Accuracy:", accuracy_score(y_valid, y_pred))
print("✅ Classification Report:\n", classification_report(y_valid, y_pred))

print("\n")

ConfusionMatrixDisplay.from_estimator(model, X_valid, y_valid, cmap='Blues')
plt.title("Confusion Matrix (Validation Set)")
plt.show()

print("\n")
# 9. Prepare test.csv for final prediction
test_passenger_ids = test_df['PassengerId']
test_df.drop(['Name', 'Ticket', 'Cabin'], axis=1, inplace=True)

test_df['Age'] = test_df['Age'].fillna(train_df['Age'].median())
test_df['Fare'] = test_df['Fare'].fillna(train_df['Fare'].median())
test_df['Embarked'] = test_df['Embarked'].fillna('S')
```