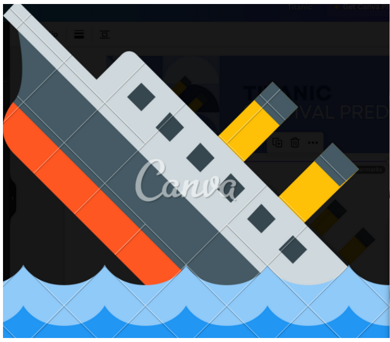

TITANIC
SURVIVAL PREDICTION

Introduction

The Titanic Survival Prediction project aims to create a machine learning model that predicts whether a passenger on the Titanic survived or not based on various features. This documentation outlines the steps taken to develop the model, the techniques used, and the evaluation of the model's performance.

Dataset overview

Data Source

The dataset used for this project is the Titanic training dataset, obtained from Kaggle. It contains information about passengers such as age, gender, class, and whether they survived or not.

Features

The features used in the model include:

- Passenger Class (Pclass)
- Age
- Sibling/Spouse Count (SibSp)
- Parent/Child Count (Parch)
- Fare
- Gender (encoded as binary)
- Embarkation Point (encoded as binary)

Data Preprocessing

Loading Data

The training data is loaded using the Pandas library.

Handling Missing Values

Missing values in the 'Age', 'Embarked', and 'Fare' columns are imputed with median and mode values.

Feature Selection

Irrelevant columns ('Survived', 'PassengerId', 'Name', 'Ticket', 'Cabin') are dropped.

Categorical Variable Encoding

Gender and Embarked columns are converted to numerical values using one-hot encoding.

Train-Test Split

The data is split into training and testing sets using the train_test_split function.

Feature Scaling

Standardization is applied to the features using the StandardScaler.

Model Training

Random Forest Classifier

A Random Forest Classifier is chosen as the predictive model due to its ability to handle complex relationships in the data.

Model Training Process

The classifier is trained on the training set after preprocessing.

Model Evaluation

Classification Report

The classification report displays precision, recall, and F1-score for each class (Survived and Not Survived).

Accuracy Score

The accuracy of the model on the test set is calculated.

Code

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score, classification_report


# Load the training data

train_data = pd.read_csv("titanictrain.csv")


# Drop columns that may not be useful for prediction

X = train_data.drop(['Survived', 'PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1)

y = train_data['Survived']


# Handle missing values (you may need to customize this based on your specific dataset)

X['Age'].fillna(X['Age'].median(), inplace=True)

X['Embarked'].fillna(X['Embarked'].mode()[0], inplace=True)

X['Fare'].fillna(X['Fare'].median(), inplace=True)


# Convert categorical variables to numerical

X = pd.get_dummies(X, columns=['Sex', 'Embarked'], drop_first=True)


# 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)


# Standardize the features

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)
```

```python
# Train a RandomForestClassifier

clf = RandomForestClassifier(random_state=42)

clf.fit(X_train, y_train)


# Make predictions on the test set

y_pred = clf.predict(X_test)


# Print the classification results

print("Classification Report:")

print(classification_report(y_test, y_pred))


# Evaluate the accuracy

accuracy = accuracy_score(y_test, y_pred)

print(f"Accuracy: {accuracy}")


# Print the first few rows of X_test, y_test, and y_pred

print("X_test:")

print(pd.DataFrame(X_test, columns=X.columns).head())

print("y_test:")

print(y_test.head())

print("y_pred:")

print(y_pred)
```

Results and Discussion

Performance Metrics

The model's performance is assessed using accuracy and classification metrics.

Model Limitations

Discuss any limitations of the model, such as potential biases and areas for improvement.

Future Improvements

Highlight potential enhancements to the model or dataset for future work.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.87      0.85       105
           1       0.80      0.76      0.78        74

    accuracy                           0.82       179
   macro avg       0.82      0.81      0.81       179
weighted avg       0.82      0.82      0.82       179

Accuracy: 0.8212290502793296
X_test:
     Pclass       Age     SibSp     Parch      Fare  Sex_male  Embarked_Q  \
0  0.813034 -0.092634  0.379923  0.784700 -0.333901  0.724310   -0.303355
1 -0.400551  0.138156 -0.470722 -0.479342 -0.425284  0.724310   -0.303355
2  0.813034 -0.708074 -0.470722 -0.479342 -0.474867  0.724310   -0.303355
3 -0.400551 -1.785093 -0.470722  0.784700  0.007966 -1.380624   -0.303355
4  0.813034 -1.169653  0.379923 -0.479342 -0.411002 -1.380624   -0.303355

   Embarked_S
0   -1.687794
1    0.592489
2    0.592489
3    0.592489
4   -1.687794
y_test:
709    1
439    0
840    0
720    1
39     1
Name: Survived, dtype: int64
```

```
y_pred:
[0 0 0 1 0 1 1 0 1 1 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1
 1 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 1 1 0 0 1 1 0 0 1 0 0 0 1 1 1 1 1
 0 0 1 1 1 1 0 1 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 1 1 0 1 0 1
 0 1 1 0 0 0 0 1 0 0 1 1 1 0 0 1 1 0 0 1 0 0 1 1 0 1 0 0 0 0 1 1 0 0 1 0 1
 1 0 0 0 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 1 0 0 0 1 1 1 0 0 0 1 1]
```

Conclusion

In conclusion, the Titanic Survival Prediction project successfully employed a Random Forest Classifier to predict passenger survival. The model demonstrated good accuracy and classification metrics. Further enhancements could focus on addressing model limitations and exploring additional features for improved predictive capabilities.