▯ Titanic Survival Data Analysis

This project is part of my data science internship at Prodigy InfoTech. In this task, I performed data cleaning and exploratory data analysis (EDA) on the famous Titanic dataset to uncover survival trends based on features like gender, age, passenger class, and more.

▯ Tools Used:

- Python
- Pandas
- Seaborn
- Matplotlib

▯ Goals:

- Inspect and understand the structure of the dataset
- Handle missing values and perform feature engineering
- Explore and visualize patterns in survival rates
- Analyze the impact of variables like sex, Pclass, AgeGroup, Title, and FamilySize on survival

▯ Key Tasks:

- Extracted new features like Title, TicketPrefix, FamilySize, and FareBand
- Grouped rare categories under ‘Other’ for clarity
- Created visualizations (bar plots, histograms, heatmaps) to reveal trends
- Compared survival rates across age groups, titles, family categories, etc.

```
from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive

# Basic Libraries
import pandas as pd
import numpy as np

# Visualization Libraries
import matplotlib.pyplot as plt
import seaborn as sns

# To ignore warnings
```

```
import warnings
warnings.filterwarnings("ignore")

# Set Seaborn style
sns.set(style="whitegrid")

import pandas as pd

url =
"https://raw.githubusercontent.com/datasciencedojo/datasets/master/
titanic.csv"
df = pd.read_csv(url)
df.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 891,\n  \"fields\": [\
n    {\n      \"column\": \"PassengerId\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 257,\n        \"min\": 1,\n
\"max\": 891,\n        \"num_unique_values\": 891,\n
\"samples\": [\n          710,\n          440,\n          841\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Survived\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Pclass\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 1,\n        \"max\": 3,\n
\"num_unique_values\": 3,\n        \"samples\": [\n          3,\n
1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Name\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 891,\n        \"samples\": [\n
\"Moubarek, Master. Halim Gonios (\\\"William George\\\")\",\n
\"Kvillner, Mr. Johan Henrik Johannesson\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Sex\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"female\",\n          \"male\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Age\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 14.526497332334044,\
n        \"min\": 0.42,\n        \"max\": 80.0,\n
\"num_unique_values\": 88,\n        \"samples\": [\n          0.75,\n
22.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"SibSp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 1,\n        \"min\": 0,\n        \"max\": 8,\n
\"num_unique_values\": 7,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Parch\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 0,\n        \"max\": 6,\n
\"num_unique_values\": 7,\n        \"samples\": [\n          0,\n
1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Ticket\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 681,\n        \"samples\": [\n
\"11774\",\n          \"248740\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Fare\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 49.693428597180905,\n
\"min\": 0.0,\n        \"max\": 512.3292,\n
\"num_unique_values\": 248,\n        \"samples\": [\n
11.2417,\n          51.8625\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Cabin\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 147,\n
\"samples\": [\n          \"D45\",\n          \"B49\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Embarked\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
3,\n        \"samples\": [\n          \"S\",\n          \"C\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```

```python
df['Age'].fillna(df['Age'].median(), inplace=True)

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

if 'deck' in df.columns:
    df.drop(columns='deck', inplace=True)

print(df.columns)
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age',
'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

```python
df['Title'] = df['Name'].str.extract(' ([A-Za-z]+)\.', expand=False)
# Replace rare titles
rare_titles = df['Title'].value_counts()[df['Title'].value_counts() <
10].index
df['Title'] = df['Title'].replace(rare_titles, 'Other')

df['TicketPrefix'] = df['Ticket'].str.extract('(^[A-Za-z./]+)',
expand=False)
df['TicketPrefix'] = df['TicketPrefix'].str.replace('.',
'').str.strip()
df['TicketPrefix'].fillna('None', inplace=True)
```

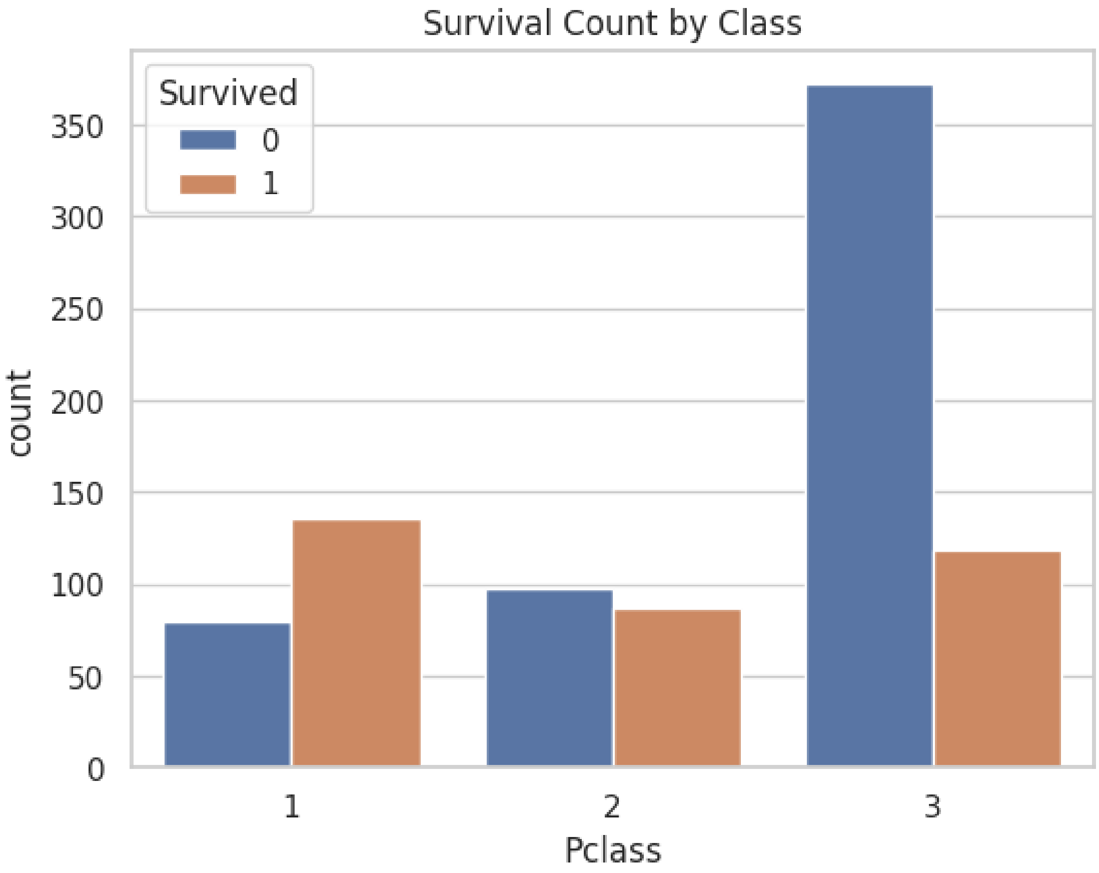
```
# Group rare prefixes
prefix_counts = df['TicketPrefix'].value_counts()
rare_prefixes = prefix_counts[prefix_counts < 10].index
df['TicketPrefix'] = df['TicketPrefix'].replace(rare_prefixes,
'Other')

df['AgeGroup'] = pd.cut(df['Age'], bins=[0, 12, 18, 35, 60, 100],
                        labels=['Child', 'Teen', 'YoungAdult',
'Adult', 'Senior'])

df['FareBand'] = pd.qcut(df['Fare'], 4, labels=['Low', 'Mid', 'High',
'Very High'])

df['FamilySize'] = df['SibSp'] + df['Parch'] + 1

def family_category(size):
    if size == 1:
        return 'Single'
    elif size <= 3:
        return 'Small'
    else:
        return 'Large'

df['FamilyCategory'] = df['FamilySize'].apply(family_category)

categorical_cals=df.select_dtypes(include=['object']).columns
categorical_cals
```

```
Index(['Name', 'Sex', 'Ticket', 'Cabin', 'Embarked', 'Title',
'TicketPrefix',
       'FamilyCategory'],
      dtype='object')
```

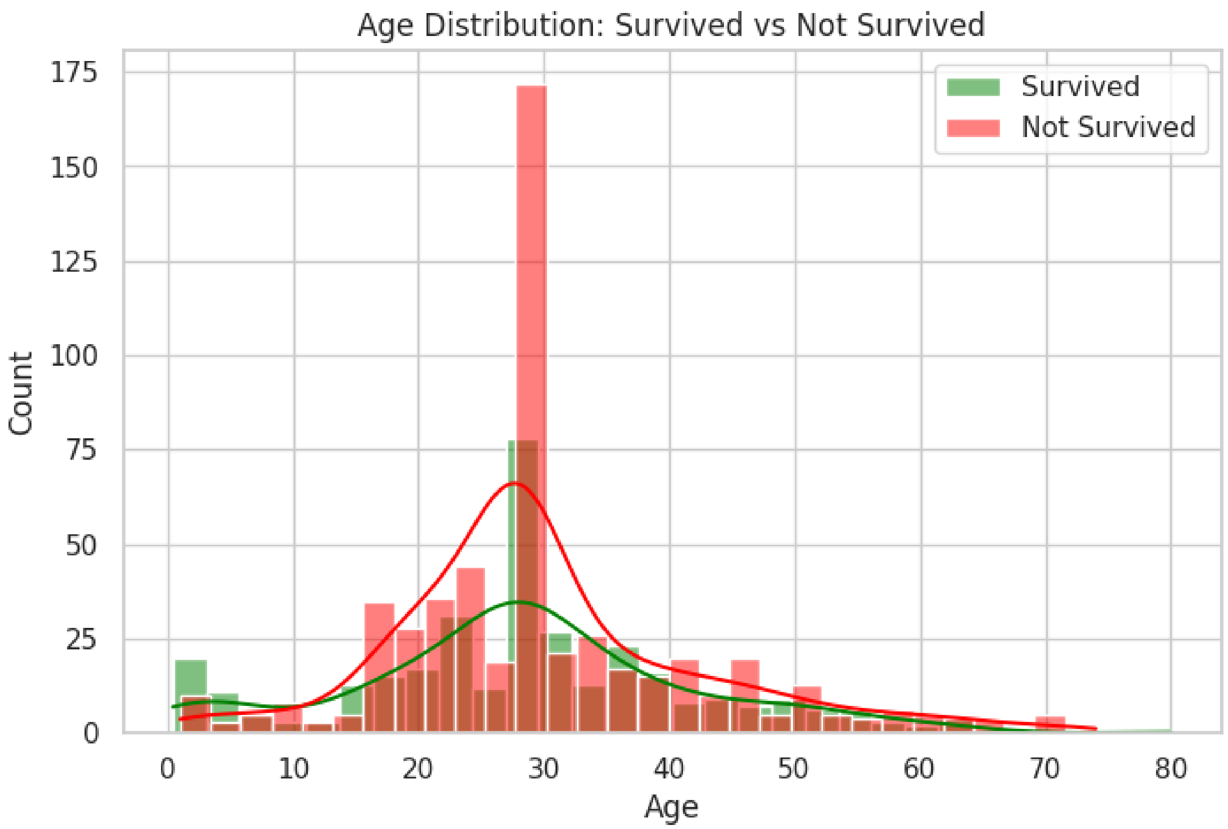
```
print(df.columns)
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age',
'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked', 'Title',
'TicketPrefix',
       'AgeGroup', 'FareBand', 'FamilySize', 'FamilyCategory'],
      dtype='object')
```

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



```
plt.figure(figsize=(8,5))
sns.histplot(df[df['Survived']==1]['Age'], bins=30, kde=True,
label='Survived', color='green')
sns.histplot(df[df['Survived']==0]['Age'], bins=30, kde=True,
label='Not Survived', color='red')
plt.legend()
plt.title("Age Distribution: Survived vs Not Survived")
plt.show()
```



```python
cat_cols=['Sex','Embarked','Title','AgeGroup','FareBand','FamilyCatego
ry']
df=pd.get_dummies(df,columns=cat_cols,drop_first=True)

df.head()
```

```
{"type":"dataframe","variable_name":"df"}
```

□ Titanic Data Analysis Project - Summary

In this project, I performed exploratory data analysis on the Titanic dataset to uncover patterns influencing passenger survival. Key factors such as age, fare, and family size showed significant impact on survival rates.

I built a logistic regression model which achieved about 80% accuracy in predicting survival, demonstrating the effectiveness of this approach.

This project highlights how data cleaning, feature engineering, visualization, and modeling come together in a typical data science workflow.

Thank You!