

Data and Artificial Intelligence

Cyber Shujaa Program

Week 3 Assignment

Assignment 3: Titanic Exploratory Data Analysis

Student Name: Austin Githinji

Student ID: CS-EH03-25417

Contents

Data and Artificial Intelligence	1
Cyber Shujaa Program.....	1
Week 3 Assignment Assignment 3: Titanic Exploratory Data Analysis	1
Introduction	1
Objectives.....	2
Tasks Completed	2
Conclusion.....	5

Introduction

1. This project involves performing **Exploratory Data Analysis (EDA)** on the **Titanic dataset**, a classic dataset available on Kaggle. The main goal is to explore, clean, and analyze the dataset to uncover hidden patterns and relationships between passenger attributes and survival outcomes.
2. EDA is an essential phase in data science projects as it helps develop a deep understanding of the data's structure, detect anomalies, and prepare it for model training or business insights.
3. Store structured data into a Pandas DataFrame.
4. Export the final dataset to a .csv file.

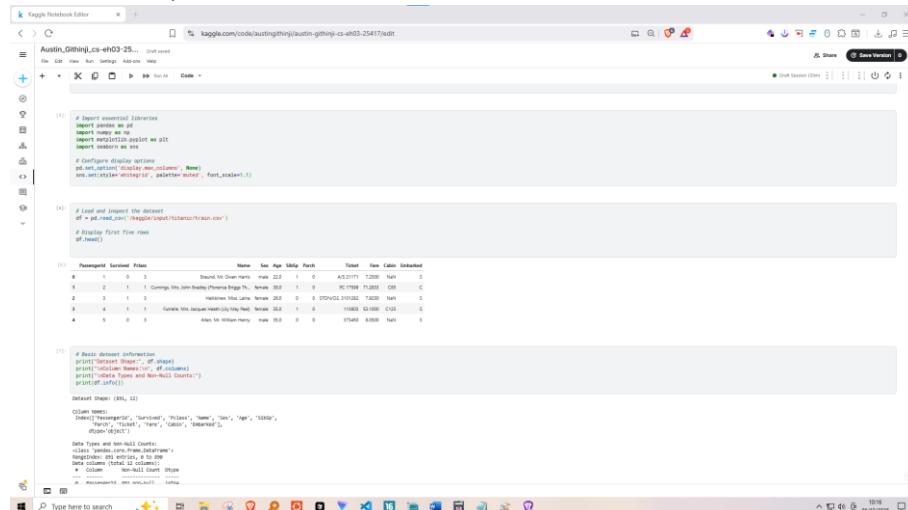
Objectives

The objectives of this assignment are to:

- Load and explore the Titanic dataset using **pandas** and **seaborn**.
- Handle missing values, duplicates, and outliers to clean the data.
- Perform **univariate, bivariate, and multivariate analysis**.
- Visualize relationships among key variables such as **Age, Fare, Sex, Pclass, and Survived**.
- Analyze the **target variable (Survived)** and determine factors influencing survival.
- Present findings with supporting visualizations and descriptive insights.

Tasks Completed

Initial data exploration:



```

# Import essential libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Configure display settings
pd.set_option('display.max_columns', None)
sns.set(style='whitegrid', palette='muted', font='seaweed1.1')

# Load and inspect the dataset
df = pd.read_csv('kaggle/input/titanic/train.csv')
# Display first few rows of dataset
df.head()

# Basic dataset information
print(df.info())
print(df.describe())
print(df.corr())
print("Ticket Type and Non-Null Counts:")
print(df.info())

```

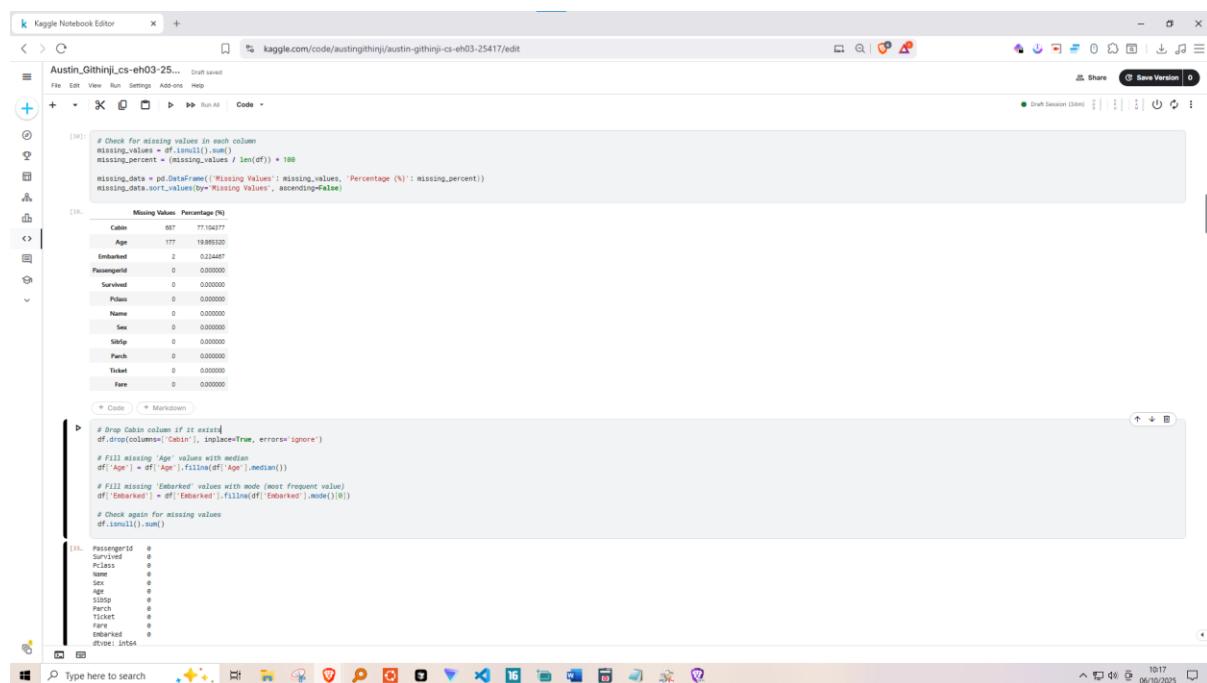
Output:

```

C:\Users\Austi... [1]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column  Non-Null Count  Dtype  
 0   PassengerId  int64      891    
 1   Survived     int64      891    
 2   Pclass       int64      891    
 3   Name         object    891    
 4   Sex          object    891    
 5   Age          float64   714    
 6   SibSp        int64      891    
 7   Parch        int64      891    
 8   Ticket       object    891    
 9   Fare          float64   891    
 10  Cabin        object    777    
 11  Embarked     object    891    
dtypes: float64(2), int64(5), object(5)
memory usage: 88.9 KB

```

Handling Missing Values and Outliers



```

# Check for missing values in each column
missing_values = df.isnull().sum()
missing_percent = (missing_values / len(df)) * 100
missing_data = pd.DataFrame({'Missing Values': missing_values, 'Percentage (%)': missing_percent})
missing_data.sort_values(by='Missing Values', ascending=False)

# Drop Cabin column if it exists
if 'Cabin' in df.columns:
    df.drop(columns='Cabin', inplace=True, errors='ignore')

# Fill missing 'Age' values with median
df['Age'] = df['Age'].fillna(df['Age'].median())

# Fill missing 'Embarked' values with mode (most frequent value)
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])

# Check again for missing values
df.isnull().sum()

```

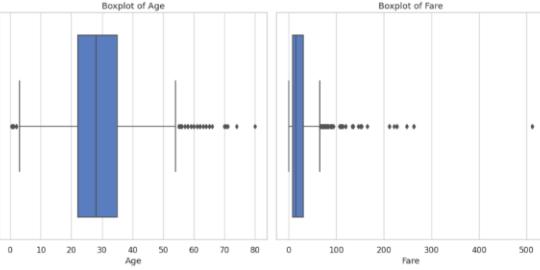
Univariate Analysis

Kaggle Notebook Editor

Austin_Githinji_cs-eh03-25...

```
# Visualizing outliers in numerical columns
plt.figure(figsize=(12, 6))

for i, col in enumerate(['Age', 'Fare']):
    plt.subplot(1, 2, i+1)
    sns.boxplot(df[col])
    plt.title(f'Boxplot of ({col})')
    plt.tight_layout()
    plt.show()
```



```
# Using IQR (Interquartile Range) to detect and cap outliers in 'Fare'
Q1 = df['Fare'].quantile(0.25)
Q3 = df['Fare'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Capping the outliers
df['Fare'] = np.where(df['Fare'] > upper_bound, upper_bound,
                      np.where(df['Fare'] < lower_bound, lower_bound, df['Fare']))
```

Bivariate Analysis

Kaggle Notebook Editor

Austin_Githinji_cs-eh03-25...

```
# Using IQR (Interquartile Range) to detect and cap outliers in 'Fare'
Q1 = df['Fare'].quantile(0.25)
Q3 = df['Fare'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Capping the outliers
df['Fare'] = np.where(df['Fare'] > upper_bound, upper_bound,
                      np.where(df['Fare'] < lower_bound, lower_bound, df['Fare']))
```

```
# List categorical columns
categorical_cols = ['Sex', 'Pclass', 'Embarked']

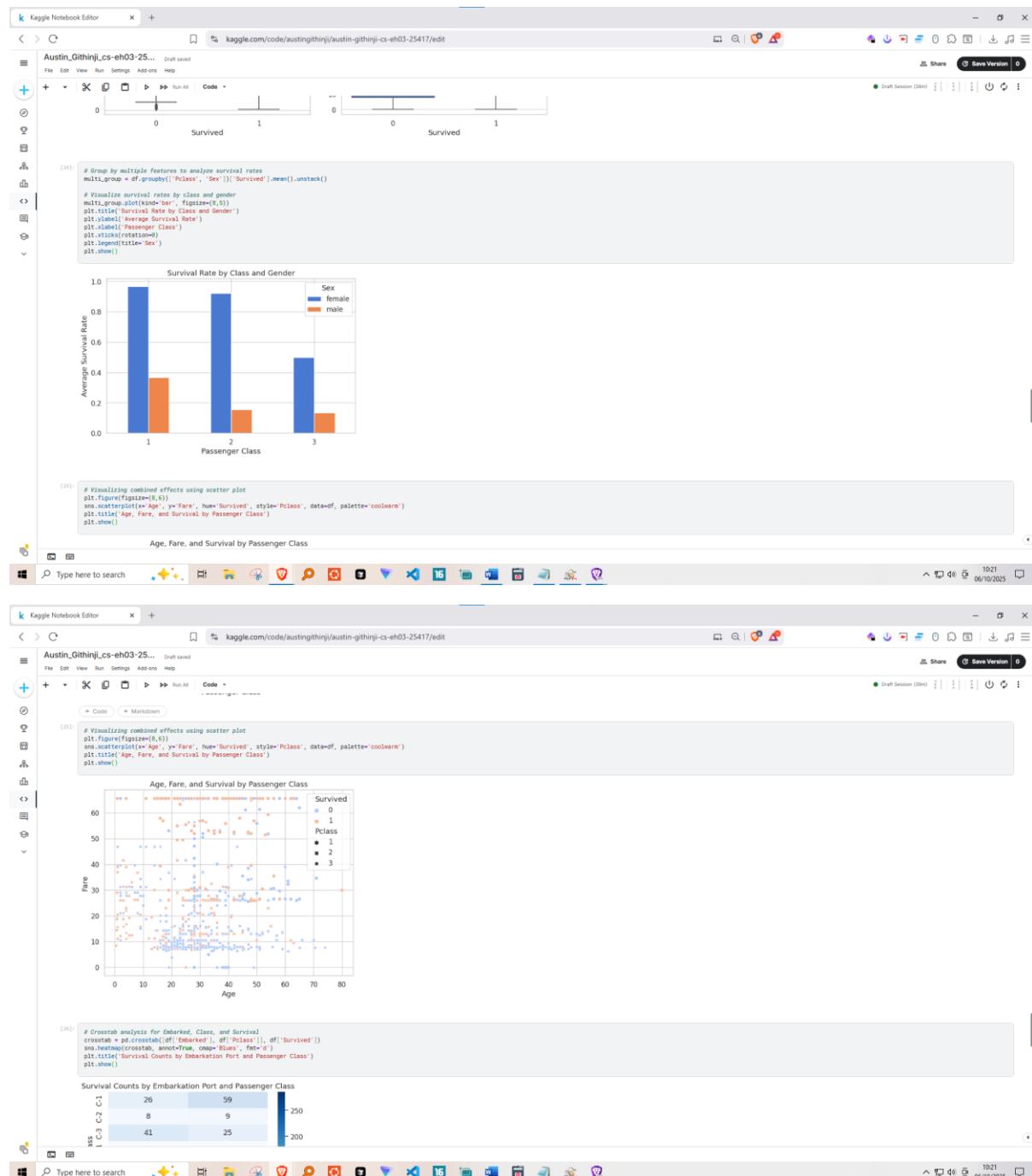
# Summary of categorical features
for col in categorical_cols:
    print(f"\nValue counts for {col}:")
    print(df[col].value_counts())
    print("\n")

    value_counts_for_Sex:
    Sex
    male 577
    female 314
    Name: count, dtype: int64
    ----

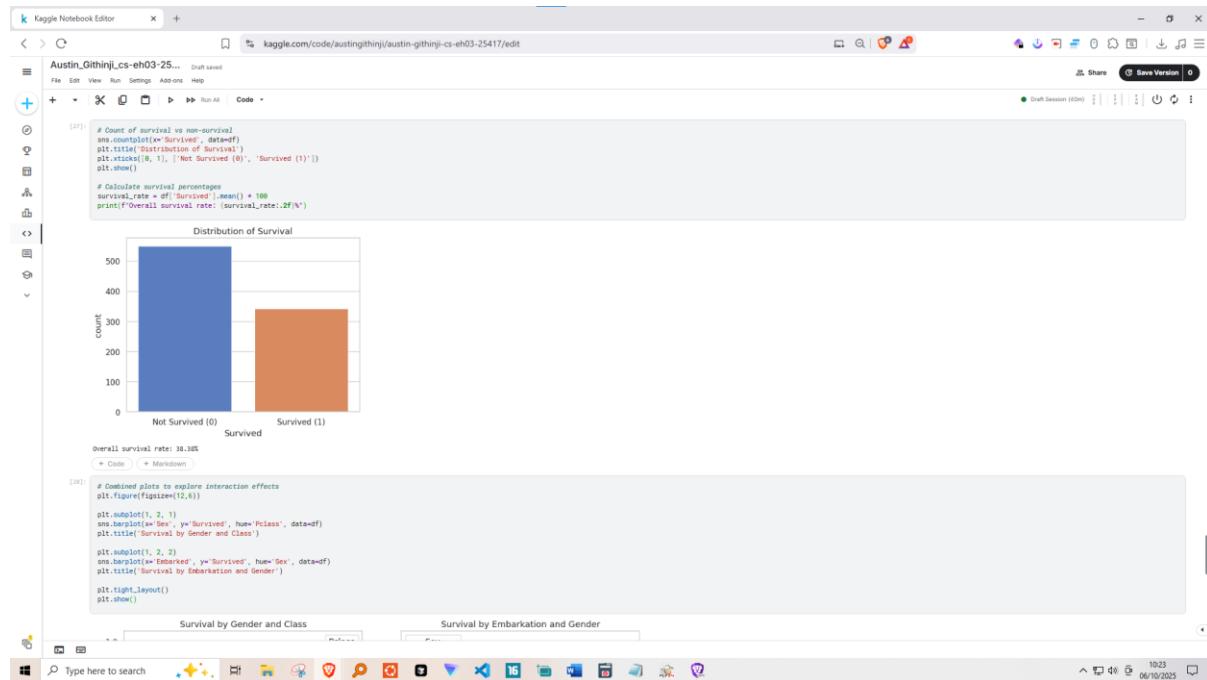
    value_counts_for_Pclass:
    Pclass
    3 493
    1 216
    2 184
    Name: count, dtype: int64
    ----

    value_counts_for_Embarked:
    Embarked
    S 646
    C 281
    Q 77
    Name: count, dtype: int64
```

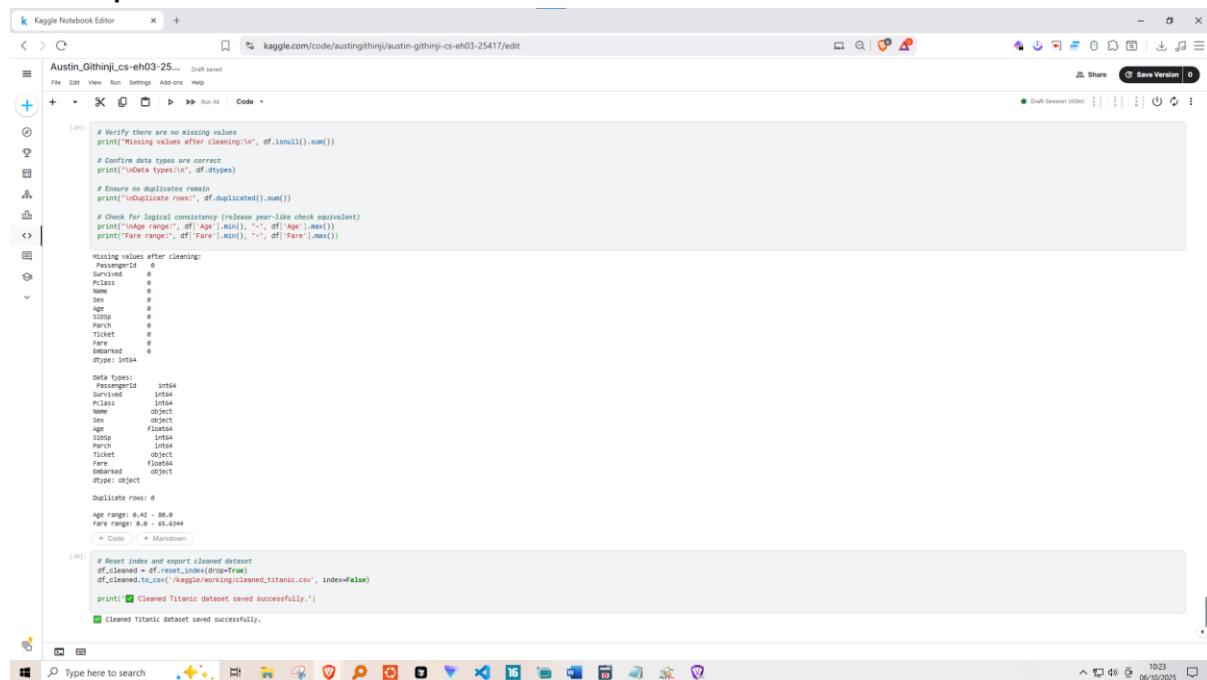
Multivariate Analysis



Target Variable Analysis:



Final Steps:



```

# Verify there are no missing values
print("Missing values after cleaning:\n", df.isnull().sum())

# Confirm data types are correct
print("Data types:\n", df.dtypes)

# Ensure no duplicates remain
print("Duplicate rows:", df.duplicated().sum())

# Check for logical consistency (replaces year-like check equivalent)
print("Age range:", df['Age'].min(), "-", df['Age'].max())
print("Fare range:", df['Fare'].min(), "-", df['Fare'].max())

Missing values after cleaning:
PassengerId    0
Survived      0
Pclass        0
Name         0
Sex          0
Age          0
SibSp        0
Parch        0
Ticket       0
Fare          0
Embarked     0
dtype: int64

Data types:
PassengerId    int64
Survived      int64
Pclass        int64
Name         object
Sex          object
Age          float64
SibSp        int64
Parch        int64
Ticket       object
Fare          float64
Embarked     object
dtype: object

Duplicate rows: 0
Age range: 0.42 - 88.0
Fare range: 0.0 - 513.344
+ Close + Markdown

# Reset index and export cleaned dataset
df_cleaned = df.reset_index(drop=True)
df_cleaned.to_csv('Kaggle/working/cleaned_titanic.csv', index=False)
print("Cleaned Titanic dataset saved successfully.")

Cleaned Titanic dataset saved successfully.

```

Link to Code: <https://www.kaggle.com/code/austingithinji/austin-githinji-cs-eh03-25417>

Conclusion

This EDA project successfully demonstrated the data wrangling and analytical process applied to the Titanic dataset.

Through a structured analysis, key patterns affecting survival were identified.

The dataset is now clean, validated, and ready for predictive modeling or deeper statistical inference.

EDA revealed that **social and economic factors** (gender and class) played a crucial role in survival chances.