Data and Artificial Intelligence

Cyber Shujaa Program

Week 3 Assignment
Exploratory data analysis

Student Name: Deborah kwamboka omae

Student ID: CS_DA02_25075

Table of contents

Table of Contents

Data and Artificial Intelligence 1

Cyber Shujaa Program 1

Week 3 Assignment Exploratory data analysis 1

Introduction 5

Objectives 5

Tasks Completed 5

Step 1: Initial data exploration 5

Code 5

code explanation 6

Screenshots of step 1 6

Step 2: handling missing values and outliers 8

Code 9

code explanation 9

Screenshots of step 2: 10

Step 3: univariate analysis 11

Code 11

code explanation 12

Screenshots to the code: 13

Step 4: Bivariate Analysis 15

code explanation 17

Screenshots: 17

Step 5: Multivariate Analysis 19

Code 19

code explanation 19

screenshot to code: 19

Step 6: Outlier detection and handling 20

Code 20

code explanation 21

Screenshots of code 21

Step 7: target variable exploration 23

Code 23

code explanation 24

screenshots to code: 24

Link to Code 27
Conclusion.....27

Table of figures

Figure 1: screenshot showing the first 20 rows by running the command df.head(20) 7
Figure 2: screenshot showing the result of df.describe() 7
Figure 3: screenshot showing the result of df.nunique() and df.duplicated() 8
Figure 4: screenshot showing the result of df.isnull().sum() 8
Figure 5: screenshot showing age distribution while checking for skewness 10
Figure 6: screenshot showing the results after filling in the missing age values with median, and dropping the cabin column. It can be seen that now there are no null values and the columns are now 11 confirming the cabin column is dropped 11
Figure 7: screenshot showing age distribution of passengers 14
Figure 8: screenshot showing the no of passengers embarking on each location 14
Figure 9: screenshot showing the distribution of ticket prices 15
Figure 10: A boxplot of passenger ticket prices 15
Figure 11: screenshot showing distribution of sibsp 22
Figure 12: screenshot showing distribution of parch 22
Figure 13: relationship between passenger class and survival rate 25
Figure 14: relationship between embarkation point and survival rate 26
Figure 15: relationship between gender and class on survival (combined effect) 26

Introduction

The assignment this week was on exploratory data analysis. Exploratory data analysis is the process of analysing datasets to summarize their main characteristics often using visual methods. Some common visuals in exploratory data analysis include histogram, boxplot, bar chat, KDE. In this week's assignment I developed hands-on experience on Exploratory Data Analysis using Kaggle Data Set and I published my work on Kaggle.

Objectives

The purpose of this assignment is to practice exploratory data analysis steps:

1. initial data exploration
2. Handling Missing Values and Outliers
3. Univariate analysis
4. Bivariate analysis
5. Multivariate analysis
6. Target variable analysis

Tasks Completed

These are the steps I took:

Step 1: Initial data exploration

In this step, I explored my dataset to have an overview of it and to better understand it. I checked for the number of rows and columns, I checked for the data type of each column, I checked for duplicated rows, for null values and its descriptive summary such as mean and standard deviation. This allowed me to understand my dataset before performing any analysis

Code

df.head(20)

df.shape

df.info()

df.describe()

df.nunique()

df.duplicated()

df.duplicated().sum()

df.isnull()

df.isnull().sum()

code explanation

The line df.head(20) returns the first 20 rows of the dataset, allowing you to preview sample records. The command df.shape returns a tuple showing the total number of rows and columns. When you run df.info(), it prints a summary of the dataset including column names, data types, and the count of non-null values. The line df.describe() returns statistical summaries such as mean, standard deviation, minimum, and maximum values for all numeric columns. Using df.nunique() returns the number of unique values in each column. The command df.duplicated() returns a Boolean Series indicating whether each row is a duplicate, while df.duplicated().sum() returns the total number of duplicate rows. Similarly, df.isnull() returns a DataFrame of True or False values showing where data is missing, and df.isnull().sum() returns the total count of missing values in each column.

Screenshots of step 1

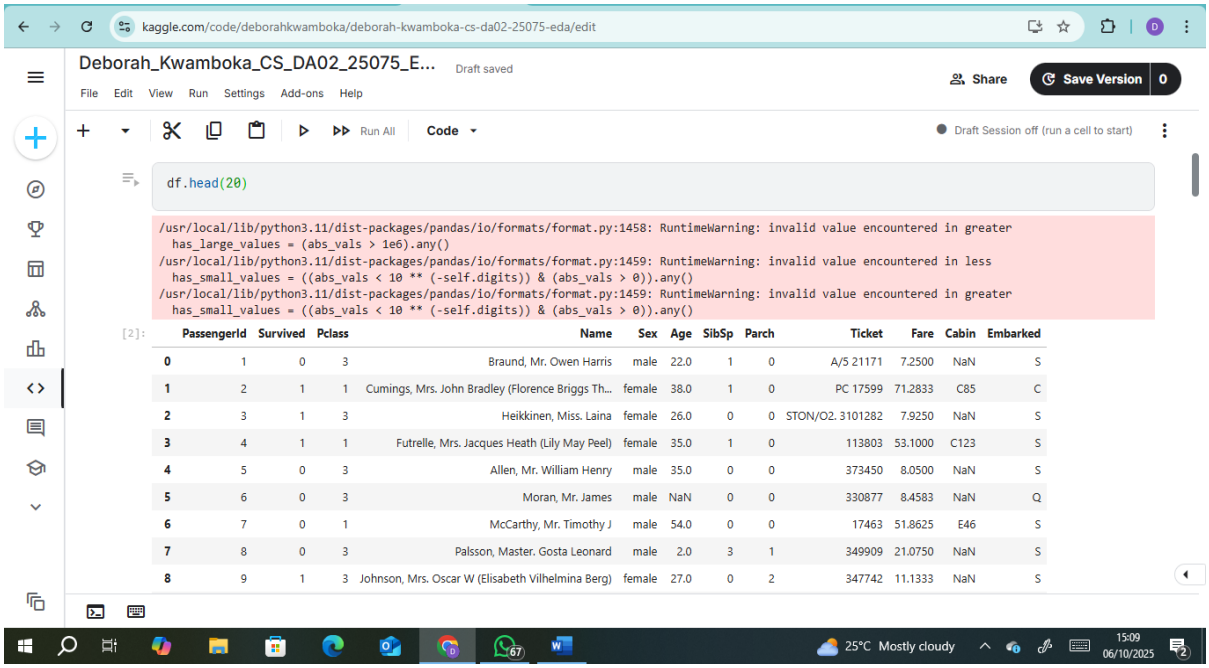


Figure 1: screenshot showing the first 20 rows by running the command df.head(20)

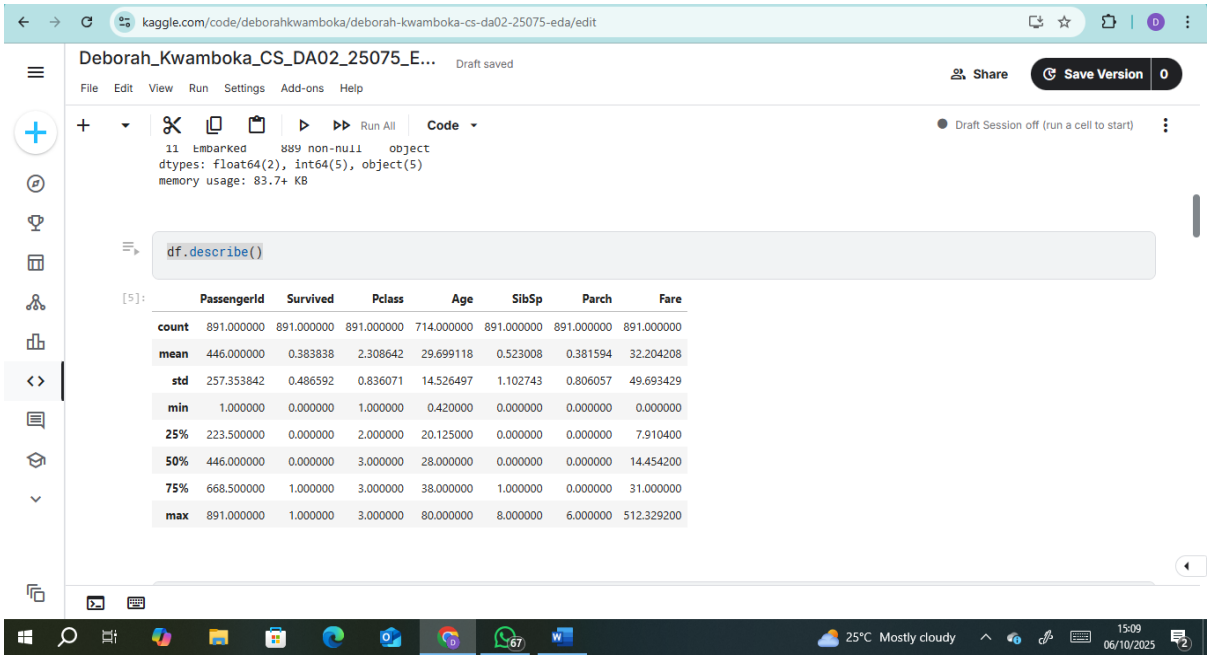


Figure 2: screenshot showing the result of df.describe()

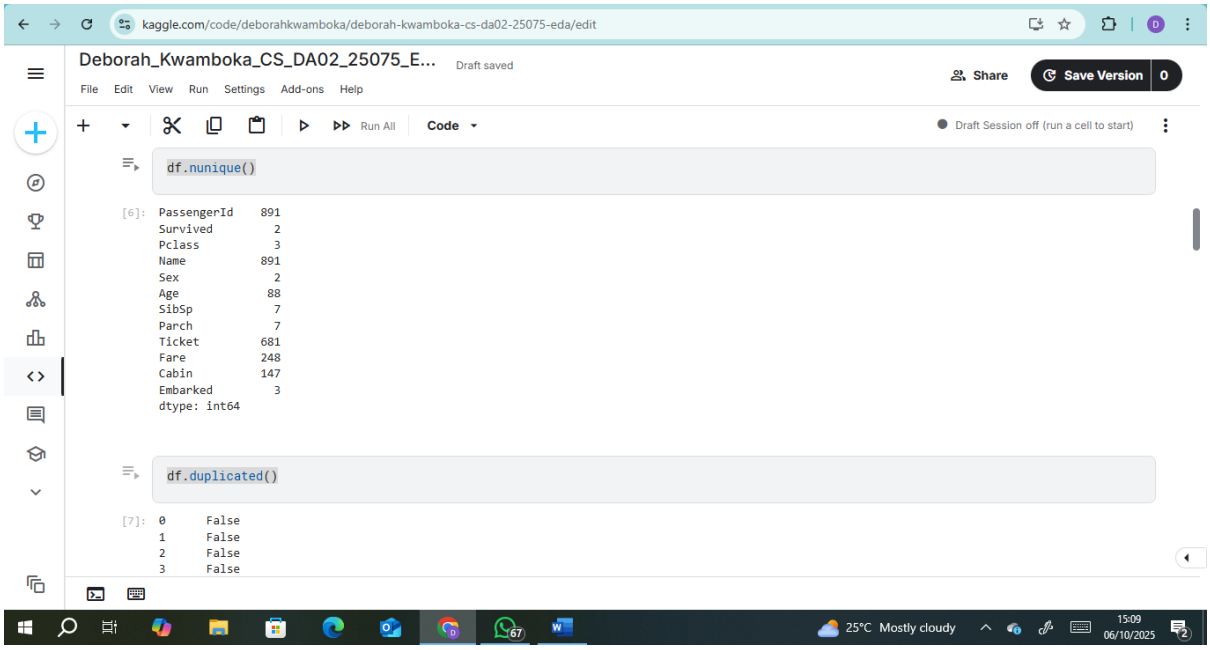


Figure 3: screenshot showing the result of df.nunique() and df.duplicated()

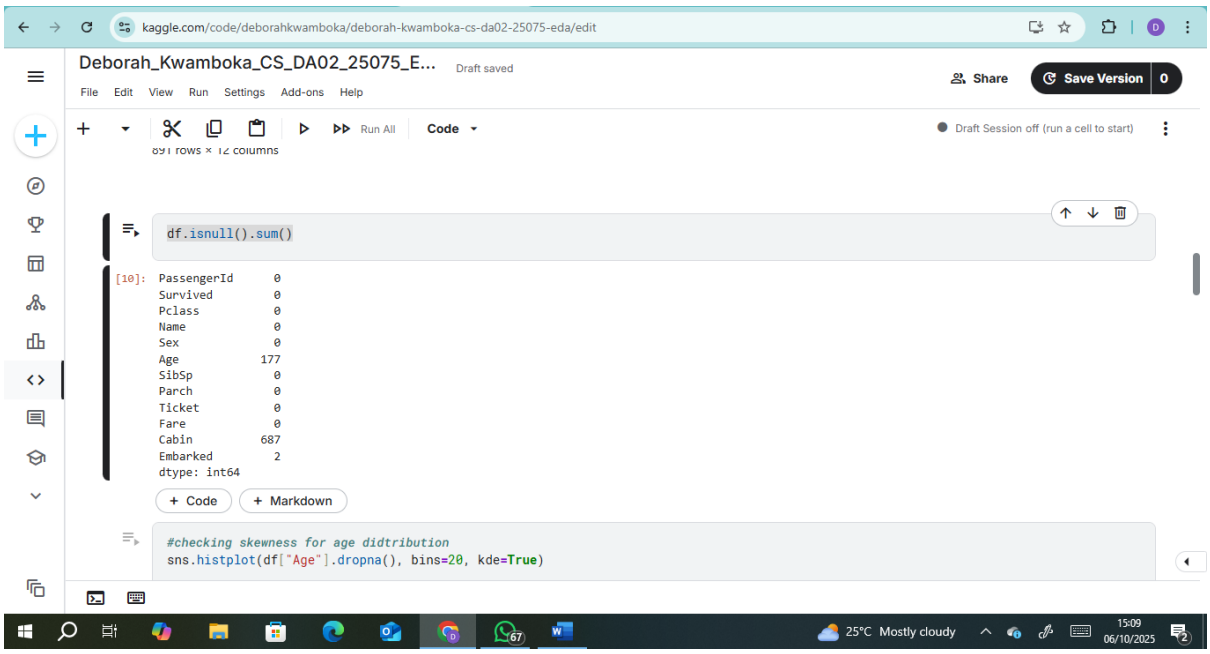


Figure 4: screenshot showing the result of df.isnull().sum()

Step 2: handling missing values and outliers

In this step I handled missing values. The age of passenger's column had 177 missing values, embarked column had 2 missing values and the cabin column had 687 missing values. I visualized the age distribution of passengers first and checked its skewness. It was right skewed hence it was appropriate to fill in the 177 missing values with the mean. I filled the

two missing values for the embarked column with the mode of that column. I dropped the 687 missing values for the cabin column since the number is high and I deemed it of no importance.

Code

```
#checking skewness for age didtribution

sns.histplot(df["Age"].dropna(), bins=20, kde=True)


plt.title("Age Distribution")

plt.xlabel("Age")

plt.ylabel("Frequency")

plt.show()

#age distribution is positively skewed so i will fill in the 177 missing values of age column using median.

df.fillna({"Age": df["Age"].median()}, inplace=True)

df["Age"].isnull().sum()

#dropping the cabin column because it has many missing values and its really of no importance

df = df.drop("Cabin", axis=1)

#confirming if the column is dropped

print(df.columns)

#filling in the two missing values for embarked column with the mode

df["Embarked"] = df["Embarked"].fillna(df["Embarked"].mode()[0])

df["Embarked"].isnull().sum()
```

code explanation

The line sns.histplot(df["Age"].dropna(), bins=20, kde=True) plots the distribution of the Age column after dropping missing values, showing how passenger ages are spread across 20

bins. The next three lines, plt.title("Age Distribution"), plt.xlabel("Age"), and plt.ylabel("Frequency"), label the plot, and plt.show() displays it. From the plotted histogram, the age distribution appears positively skewed, meaning most passengers were younger while a few were much older. To handle the 177 missing values in the Age column, the line df.fillna({"Age": df["Age"].median()}, inplace=True) replaces all missing ages with the median age, which is suitable for skewed data since the median is less affected by outliers. The line df["Age"].isnull().sum() then returns the count of remaining missing values in the Age column to confirm they were filled. The line df = df.drop("Cabin", axis=1) removes the Cabin column from the dataset because it contained too many missing values and was not useful for analysis. To verify that the column was successfully dropped, print(df.columns) returns the list of remaining column names. Finally, the line df["Embarked"] = df["Embarked"].fillna(df["Embarked"].mode()[0]) fills the two missing values in the Embarked column with the mode (most frequent value), and df["Embarked"].isnull().sum() confirms that no missing values remain.

Screenshots of step 2:

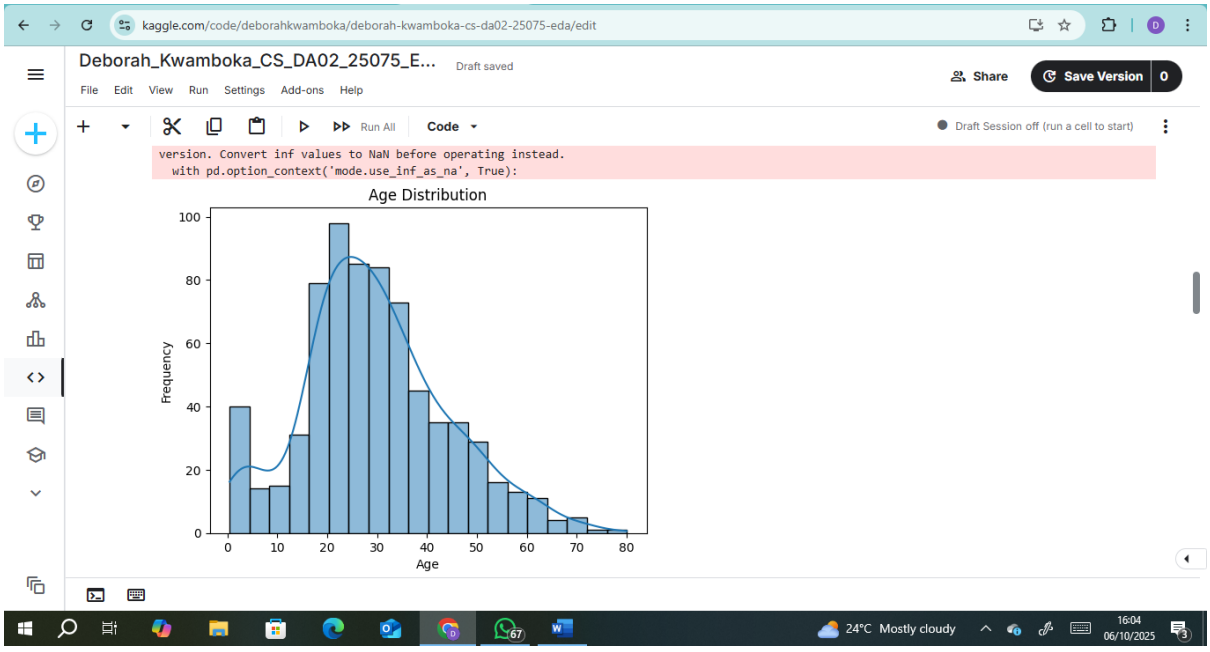


Figure 5: screenshot showing age distribution while checking for skewness

Figure 6: screenshot showing the results after filling in the missing age values with median, and dropping the cabin column. It can be seen that now there are no null values and the columns are now 11 confirming the cabin column is dropped

Step 3: univariate analysis

In this step I examined each feature at a time to understand its distribution. I analysed the age distribution of passenger. I examined the number of passengers embarking from each location. I also did analysis on the distribution of ticket prices checking for skewness.

Code

```
#visualizing the age distribution of passengers through a histogram

sns.histplot(data=df, x="Age", bins=20, kde=True, color="blue")

plt.title("Age Distribution of Passengers")

plt.xlabel("Age")

plt.ylabel("Count")

plt.show()


#how many passengers embarked from each location

embark_counts = df["Embarked"].value_counts()
```

```
print(embark_counts)


#checking if ticket prices are evenly distributed, or are they skewed?
# Histogram
plt.figure(figsize=(8,5))
sns.histplot(df["Fare"], bins=50, kde=True, color="purple")
plt.title("Distribution of Passenger Ticket Prices (Fare)")
plt.xlabel("Fare")
plt.ylabel("Count")
plt.show()


# Boxplot to see skew & outliers
plt.figure(figsize=(8,3))
sns.boxplot(x= df["Fare"], color="lightblue")
plt.title("Boxplot of Passenger Ticket Prices")
plt.xlabel("Fare")
plt.show()
```

code explanation

The line sns.histplot(data=df, x="Age", bins=20, kde=True, color="blue") creates a histogram to visualize the distribution of passenger ages in the dataset df. The parameter x="Age" specifies that the Age column is being plotted, bins=20 divides the ages into 20 equal intervals, and kde=True adds a Kernel Density Estimate (KDE) curve to show the smooth shape of the distribution. The color="blue" argument sets the histogram bars and KDE line to blue. The next three lines — plt.title("Age Distribution of Passengers"), plt.xlabel("Age"), and plt.ylabel("Count") add a title and label the x- and y-axes for better readability. Finally, plt.show() displays the plot. This visualization helps identify how ages

are spread across passengers, whether the distribution is skewed, and if there are any noticeable age groups or gaps in the data.

The line embark_counts = df["Embarked"].value_counts() returns the number of passengers who embarked from each location (C = Cherbourg, Q = Queenstown, S = Southampton) by counting how many times each embarkation point appears in the Embarked column. The line print(embark_counts) then displays these counts in descending order, helping identify which port most passengers boarded from.

Next, the code checks whether ticket prices (Fare) are evenly distributed or skewed. The block starting with plt.figure(figsize=(8,5)) creates a figure for the histogram, and sns.histplot(df["Fare"], bins=50, kde=True, color="purple") plots the distribution of passenger fares across 50 bins. The kde=True adds a smooth density curve, and color="purple" sets the histogram color. The following lines — plt.title(), plt.xlabel(), and plt.ylabel() — label the plot, while plt.show() displays it. This histogram helps visualize whether most fares are concentrated in lower or higher ranges, revealing if the data is right-skewed.

The next block uses a boxplot to examine skewness and detect outliers. The line plt.figure(figsize=(8,3)) defines the figure size, and sns.boxplot(x=df["Fare"], color="lightblue") creates a horizontal boxplot showing the median, quartiles, and outliers of the Fare variable. The title and label clarify the plot, and plt.show() displays it. The boxplot visually confirms whether there are extreme fare values (outliers) and supports conclusions about data skewness observed in the histogram.

Screenshots to the code:

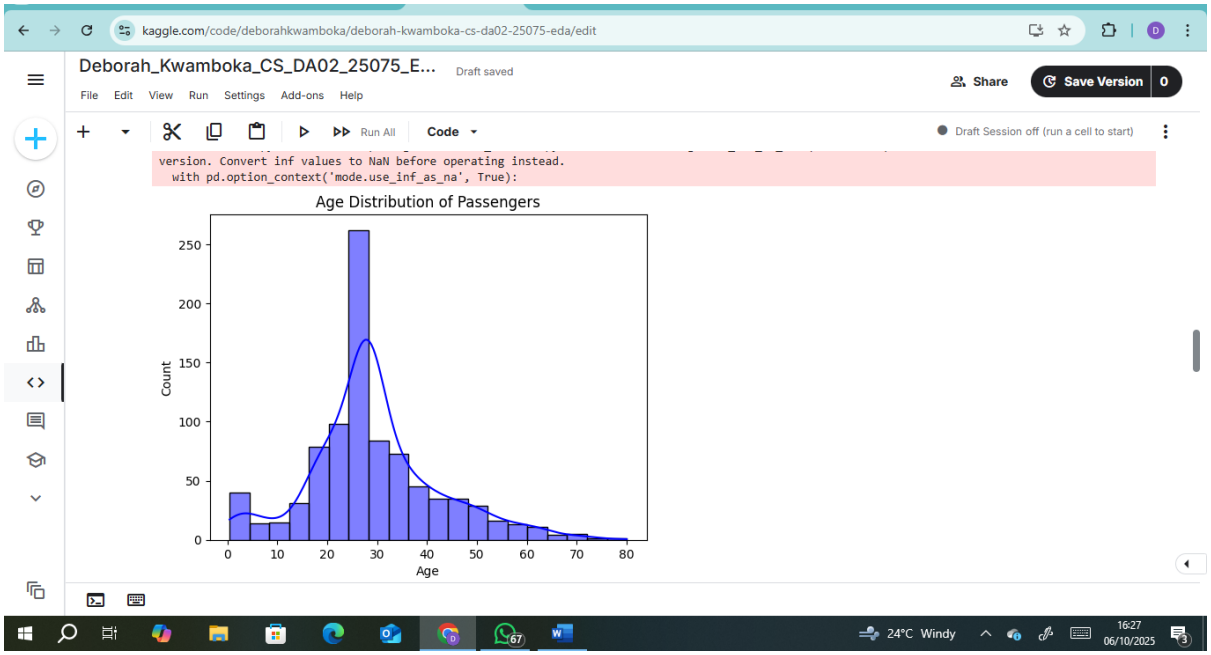


Figure 7: screenshot showing age distribution of passengers

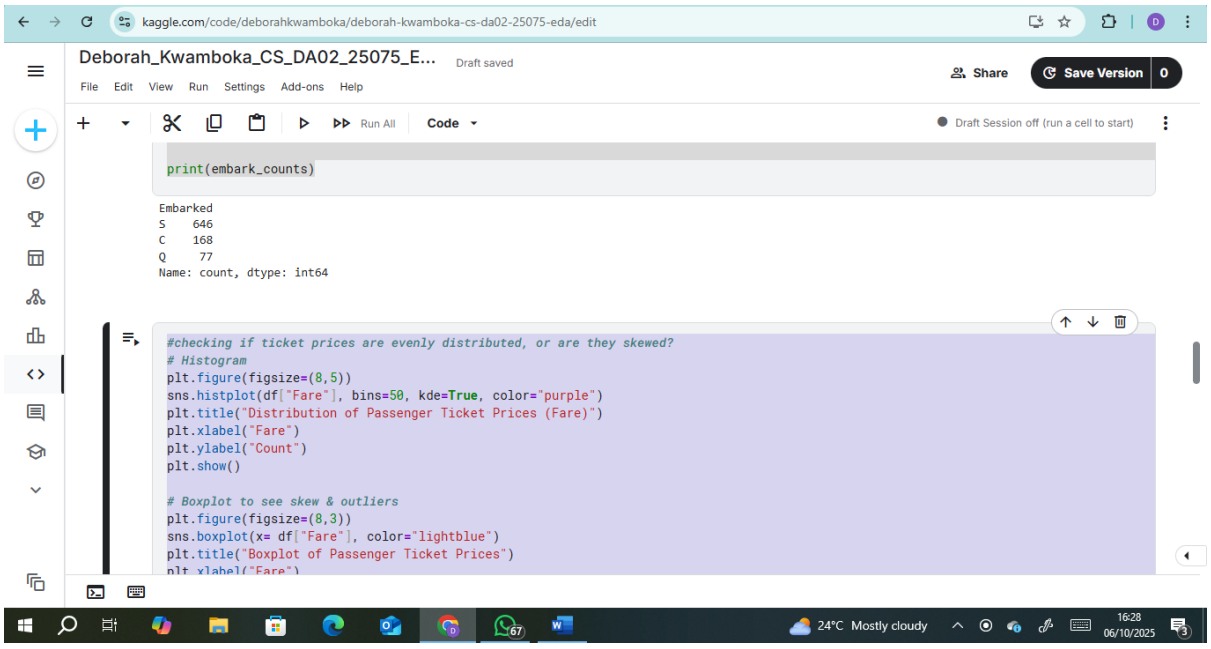


Figure 8: screenshot showing the no of passengers embarking on each location

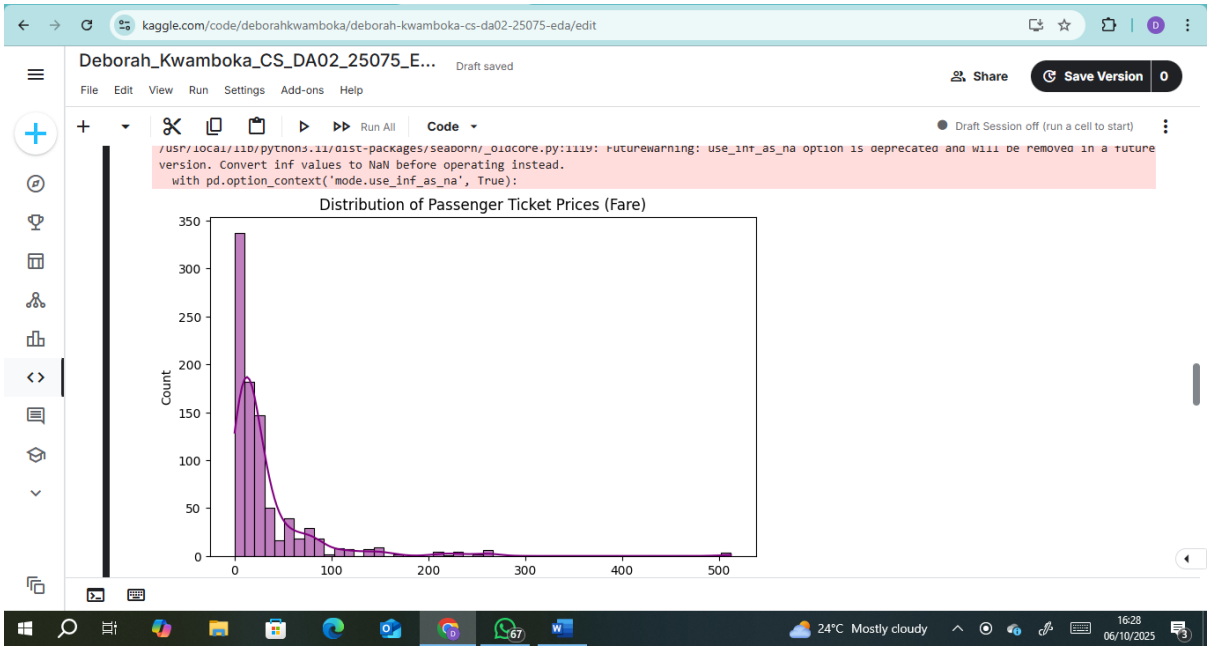


Figure 9: screenshot showing the distribution of ticket prices

Figure 10: A boxplot of passenger ticket prices

Step 4: Bivariate Analysis

In this step, I conducted bivariate analysis to explore how pairs of variables relate to each other. I examined pairs of features of interest. I analysed two variables at a time. I examined

if the fare changes depending on the Passenger class. I examined if age, younger or older affected the likelihood of survival. I analysed if the embarked location affected survival rate.

Code

```
# 1. Relationship between Passenger Class and Fare

plt.figure(figsize=(8,5))

sns.boxplot(x='Pclass', y='Fare', data=df, palette='cool')

plt.title('Relationship between Passenger Class and Fare')

plt.xlabel('Passenger Class')

plt.ylabel('Fare')

plt.show()


# 2. Relationship between Age and Survival

plt.figure(figsize=(8,5))

sns.boxplot(x='Survived', y='Age', data=df, palette='viridis')

plt.title('Effect of Age on Survival')

plt.xlabel('Survival (0 = No, 1 = Yes)')

plt.ylabel('Age')

plt.show()


# 3. Relationship between Embarked Location and Survival Rate

plt.figure(figsize=(8,5))

sns.barplot(x='Embarked', y='Survived', data=df, palette='magma')

plt.title('Survival Rate by Embarked Location')

plt.xlabel('Embarked Location')

plt.ylabel('Average Survival Rate')
```

plt.show()

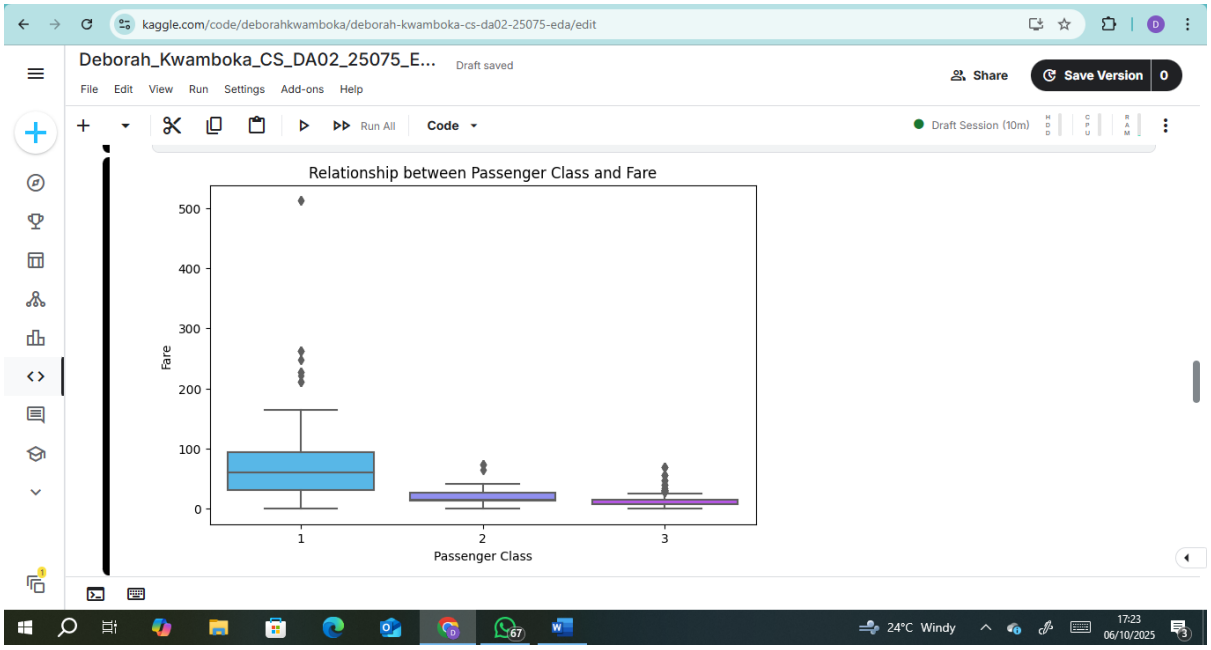
code explanation

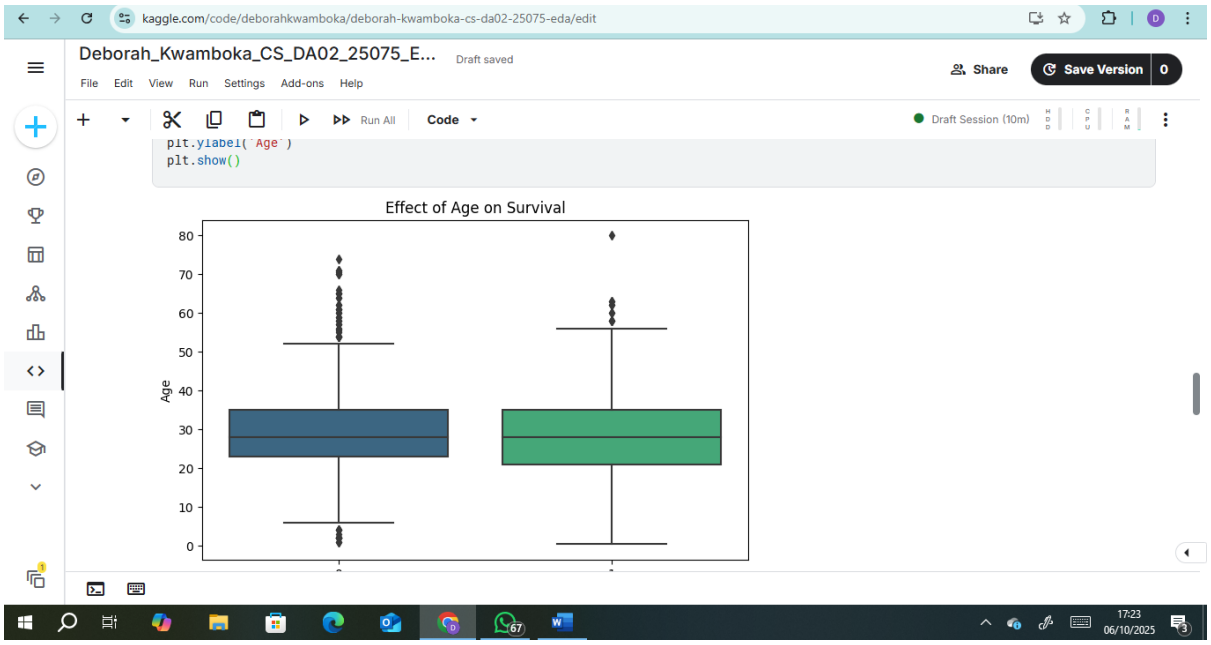
The first block, plt.figure(figsize=(8,5)), creates a figure to control the plot size, and sns.boxplot(x='Pclass', y='Fare', data=df, palette='cool') plots a boxplot showing how ticket prices (Fare) vary across different Passenger Classes (Pclass). Each box represents the distribution of fares for a specific class showing that first-class passengers paid higher fares, while third-class passengers paid the lowest. The palette='cool' adds a blue-green color theme. The following lines add a title and axis labels, and plt.show() displays the plot. This visualization helps identify the relationship between class and fare, as well as any outliers in ticket prices.

The second block examines how Age relates to Survival. The line sns.boxplot(x='Survived', y='Age', data=df, palette='viridis') creates a boxplot comparing the age distribution of passengers who survived (1) versus those who did not (0). The plot shows whether survivors tended to be younger or older, and if there are notable differences between the two groups. The axis labels and title describe the plot, and plt.show() displays it.

The third block analyzes the relationship between Embarked location and survival rate. The line sns.barplot(x='Embarked', y='Survived', data=df, palette='magma') calculates the average survival rate for each embarkation point (C, Q, S) and displays it as a bar chart. Each bar's height represents the proportion of passengers who survived from that port. The title and labels clarify the meaning of the axes, and plt.show() renders the plot. This visualization helps determine whether passengers from certain embarkation points had higher survival chances. The results shows that its from point c.

Screenshots:


kaggle.com/code/deborahkwamboka/deborah-kwamboka-cs-da02-25075-eda/edit
Deborah_Kwamboka_CS_DA02_25075_E...
Draft saved
File Edit View Run Settings Add-ons Help
Share
Save Version 0
Run All
Code
Draft Session (10m)
Relationship between Passenger Class and Fare
Fare
Passenger Class
0
100
200
300
400
500
1
2
3
24°C Windy
17:23
06/10/2025


kaggle.com/code/deborahkwamboka/deborah-kwamboka-cs-da02-25075-eda/edit
Deborah_Kwamboka_CS_DA02_25075_E...
Draft saved
File Edit View Run Settings Add-ons Help
Share
Save Version 0
Run All
Code
Draft Session (10m)
plt.ylabel('Age')
plt.show()
Effect of Age on Survival
Age
0
10
20
30
40
50
60
70
80
17:23
06/10/2025

Step 5: Multivariate Analysis

In this step I explored more complex relationships between three or more variables simultaneously. I aimed to detect interactions, combined effects, and hidden patterns that were not visible in bivariate analysis for example I analysed how Pclass, Age, and Fare jointly affect survival.

Code

sns.scatterplot(data=df, x='Age', y='Fare', hue='Survived', style='Pclass', palette='coolwarm')

plt.title("Combined Effect of Age, Fare, and Pclass on Survival")

plt.show()

code explanation

This line of code sns.scatterplot(data=df, x='Age', y='Fare', hue='Survived', style='Pclass', palette='coolwarm') creates a scatter plot that shows how Age and Fare jointly affect survival, where the color (hue) represents whether a passenger survived and the marker style (style) represents the passenger class, helping to visualize the combined influence of age, fare, and class on survival patterns.

screenshot to code:

Step 6: Outlier detection and handling

The intention of this step was to explore different ways to handle outliers, which include removing, capping, imputing, or leaving them as is. Removing outliers in Fare may help for predictive models, but could hide important insights for understanding passenger wealth.

The Age variable had a few mild outliers but most values were within a reasonable human range, so I decided to retain them to preserve data authenticity.

Code

```
plt.figure(figsize=(10, 8))


# SibSp distribution

plt.subplot(2, 2, 1)

sns.histplot(df['SibSp'], bins=8, kde=True, color='skyblue')

plt.title("Distribution of SibSp (Siblings/Spouses Aboard)")

plt.xlabel("SibSp")

plt.ylabel("Count")
```

```
plt.subplot(2, 2, 2)

sns.boxplot(x=df['SibSp'], color='lightcoral')

plt.title("Boxplot of SibSp")

plt.xlabel("SibSp")


# Parch distribution

plt.subplot(2, 2, 3)

sns.histplot(df['Parch'], bins=8, kde=True, color='lightgreen')

plt.title("Distribution of Parch (Parents/Children Aboard)")

plt.xlabel("Parch")

plt.ylabel("Count")


plt.subplot(2, 2, 4)

sns.boxplot(x=df['Parch'], color='orange')

plt.title("Boxplot of Parch")

plt.xlabel("Parch")


plt.tight_layout()

plt.show()
```

code explanation

The above block of code detects outliers in siblings/spouse and parent/children distribution

Screenshots of code

Figure 11: screenshot showing distribution of sibsp

Figure 12: screenshot showing distribution of parch

From the screenshots above it can be seen that sibsp and parch have a few outliers. I decided to retain them as they are true values that show the relationship that was there among the passengers that boarded.

Step 7: target variable exploration

In this step I analysed the Target/Dependent Variable Survived and explored the distribution of the target variable (Survived) using countplots and bar plots. I examined how balanced or imbalanced the dataset is and what factors (like age, gender, class, or embarkation point) may influence survival. I also used combined plots to detect interaction effects.

Code

```
#target variable exploration

sns.countplot(data=df, x='Survived', palette='pastel')

plt.title("Distribution of Survival")

plt.xlabel("Survival (0 = No, 1 = Yes)")

plt.ylabel("Count")

plt.show()


sns.catplot(data=df, x='Embarked', y='Survived', hue='Pclass', kind='bar', palette='viridis')

plt.title("Survival Rate by Embarkation Port and Passenger Class")

plt.show()


#factors influencing survival

sns.barplot(data=df, x='Sex', y='Survived', palette='coolwarm')

plt.title("Survival Rate by Gender")

plt.show()


sns.barplot(data=df, x='Pclass', y='Survived', palette='mako')

plt.title("Survival Rate by Passenger Class")
```

plt.show()

sns.barplot(data=df, x='Embarked', y='Survived', palette='viridis')

plt.title("Survival Rate by Embarkation Point")

plt.show()

#combined plots to determine interaction effects

sns.catplot(data=df, x='Pclass', y='Survived', hue='Sex', kind='bar', palette='husl')

plt.title("Interaction between Gender and Class on Survival")

plt.show()

code explanation

The line sns.catplot(data=df, x='Embarked', y='Survived', hue='Pclass', kind='bar', palette='viridis') generates a bar chart comparing survival rates across embarkation ports while distinguishing passengers by class, showing how both factors interact to influence survival.

The line sns.barplot(data=df, x='Sex', y='Survived', palette='coolwarm') displays the average survival rate for each gender, revealing that females had a much higher chance of survival than males. Similarly, sns.barplot(data=df, x='Pclass', y='Survived', palette='mako') shows the survival rate by passenger class, where first-class passengers survived more often than those in third class, while sns.barplot(data=df, x='Embarked', y='Survived', palette='viridis') examines survival differences by embarkation point.

Finally, sns.catplot(data=df, x='Pclass', y='Survived', hue='Sex', kind='bar', palette='husl') creates a combined bar plot that shows the interaction between gender and class on survival, illustrating that females in higher classes had the highest survival rates while males in lower classes had the lowest.

screenshots to code:

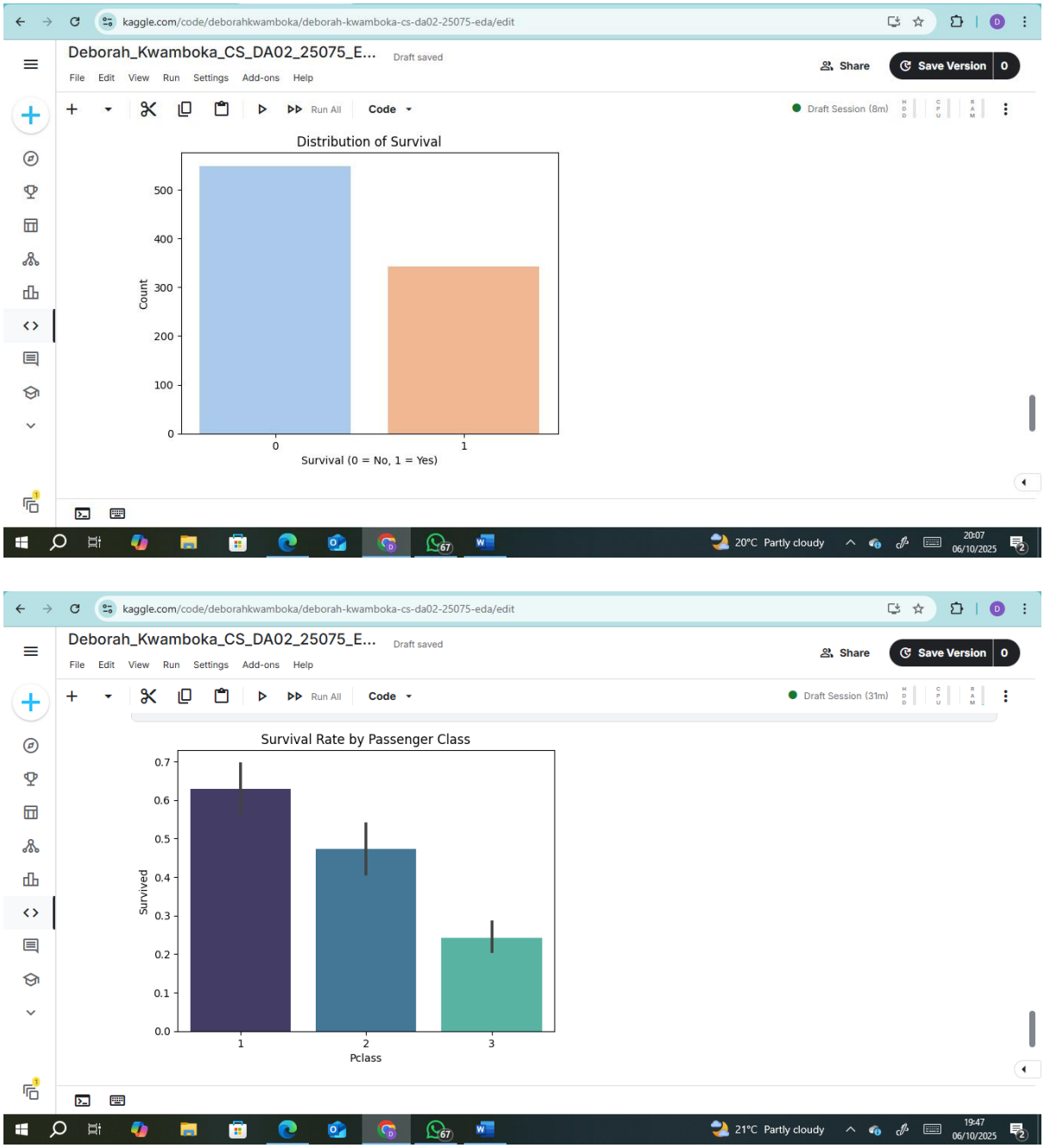


Figure 13: relationship between passenger class and survival rate

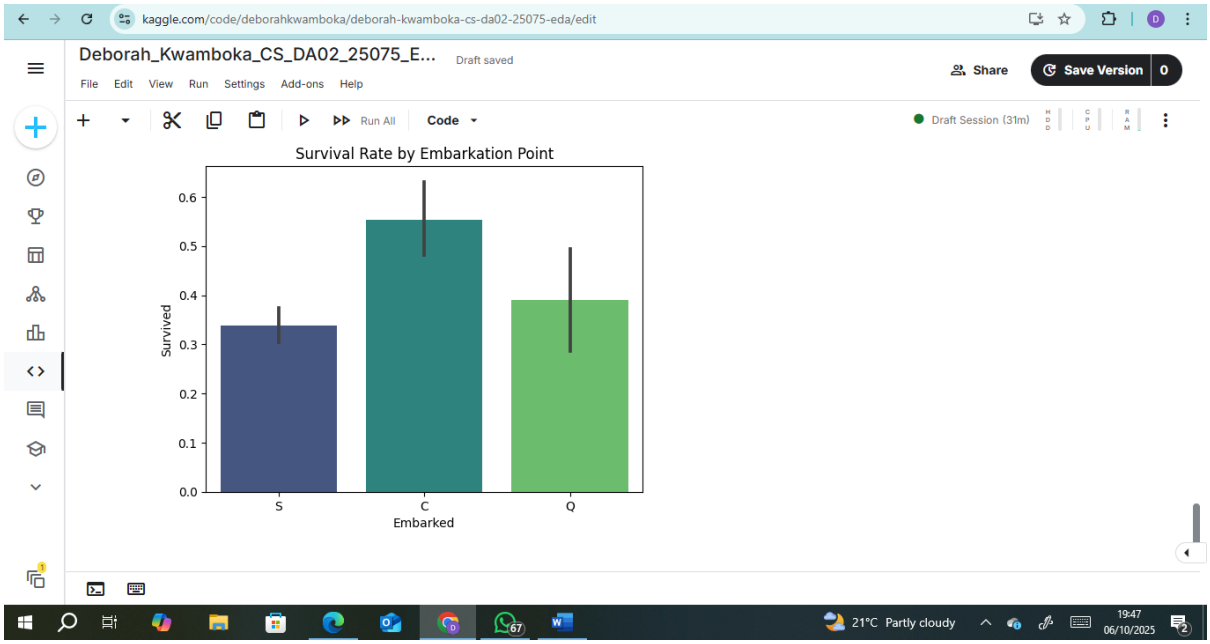


Figure 14: relationship between embarkation point and survival rate

Figure 15: relationship between gender and class on survival (combined effect)

Link to Code

Link to Code: https://www.kaggle.com/code/deborahkwamboka/deborah-kwamboka-cs-da02-25075-eda

Conclusion

This week I gained insights and hands on experience on exploratory data analysis. It was interesting to get an overview of a dataset and dive deeper into summarizing and understanding the data with visualizations. Through exploratory data analysis I was able to analyse the relationship between one or more variables in a data set. This week's assignment has given me practical experience of visualizing data and detecting and handling outliers.