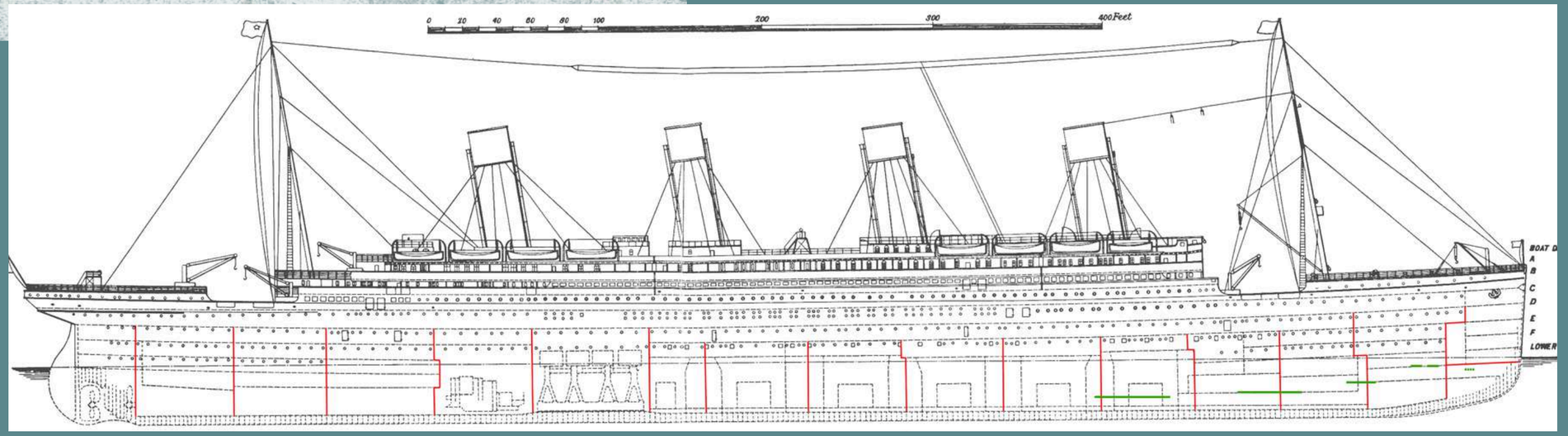
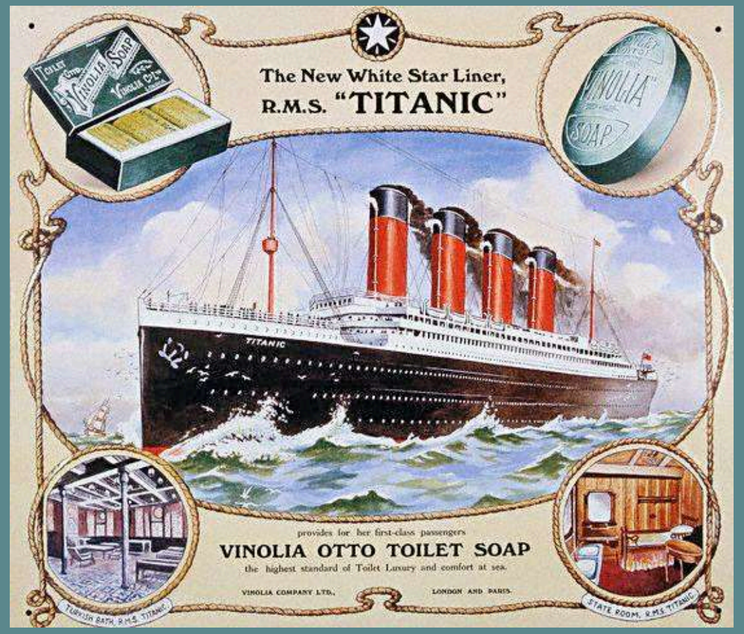
TITANIC

Python Unit Summary Exercise / Yui Hanamura



INTRODUCTION



Unknown - http://www.uwants.com/viewthread.php, Public Domain, https://commons.wikimedia.org/w/index.php

" RMS Titanic was a British ocean liner that sank in the early hours of 15 April 1912 as a result of striking an iceberg on her maiden voyage from Southampton, England, to New York City, United States." [https://en.wikipedia.org/wiki/Titanic]

In this unit summary, we will analyze the Titanic data and see if there is a correlation between the chances of survival and certain characteristics of each passenger (gender, cabin class, ticket price, etc.)

DATA FRAME OF KEY ANALYSIS

OVERVIEW

They have several colums: PassengerId, Pclass, Survived, Sex etc. The most important fact is 'Survived' or not, so this analysis focus on this column though this report.

COLUMN

In this analysis, column name [Cabin] is delete.
The following columns contain text.
Name, Sex, Ticket, Cabin, Embarked
Survived columns indicate survival as 1 or 0.

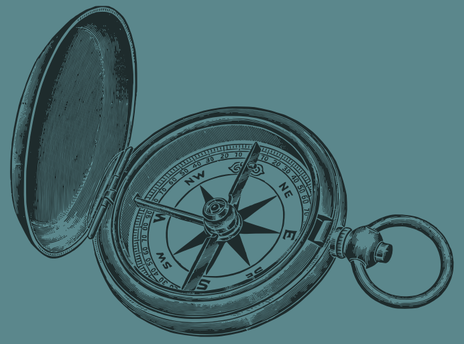
CLEANING

Some columns are blank. Fill in with suitable numbers, as the quantity and quality may affect the analysis.
If the number of blanks is small in relation to the total quantity, fill in with the mode number.

ABOUT

Information on the crew and passengers of the Titanic is widely published as a standard dataset in data science; it is built into R language and other languages as a standard.
[https://www.kaggle.com/c/titanic]

THIS IS AN APPENDIX



All answers for assess questions are written in Jupyter notebook. This presentation is appendix.

This presentation will cover :
Data cleaning with python, visualization, and three questions and answers.

DATASET

Just now, We have this data sets. 891 rows , 12 column

# PassengerId	# Survived	# Pclass	Name	Sex	# Age	# SibSp	# Parch	Ticket	# Fare	Cabin	Embarked
Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 177 (20%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 687 (77%)	Missing: 2 (<1%)
Distinct: 891 (100%)	Distinct: 2 (<1%)	Distinct: 3 (<1%)	Distinct: 891 (100%)	Distinct: 2 (<1%)	Distinct: 88 (10%)	Distin... 7 (<1%)	Distinct: 7 (<1%)	Distinct: 681 (76%)	Distinct: 248 (28%)	Distinct: 147 (16%)	Distinct: 3 (<1%)
Min 1 Max 891	Min 0 Max 1	Min 1 Max 3	891 Distinct values	male 65% female 35%	Min 0.42 Max 80.0	Min 0 Max 8	Min 0 Max 6	681 Distinct values	Min 0.0 Max 512.3292	147 Distinct values	S 72% C 19% Q 9%
1	0	3	Braund, Mr. Owen Harr	male	22.0	1	0	A/5 21171	7.25	*Missing value*	S
2	1	1	Cumings, Mrs. John Bra	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.925	*Missing value*	S
4	1	1	Futrelle, Mrs. Jacques H	female	35.0	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. William Henr	male	35.0	0	0	373450	8.05	*Missing value*	S
6	0	3	Moran, Mr. James	male	*Missing value*	0	0	330877	8.4583	*Missing value*	Q
7	0	1	McCarthy, Mr. Timothy	male	54.0	0	0	17463	51.8625	E46	S
8	0	3	Palsson, Master. Gosta	male	2.0	3	1	349909	21.075	*Missing value*	S
9	1	3	Johnson, Mrs. Oscar W	female	27.0	0	2	347742	11.1333	*Missing value*	S
10	1	2	Nasser, Mrs. Nicholas (	female	14.0	1	0	237736	30.0708	*Missing value*	C
11	1	3	Sandstrom, Miss. Margu	female	4.0	1	1	PP 9549	16.7	G6	S
12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.55	C103	S
13	0	3	Saundercock, Mr. Willia	male	20.0	0	0	A/5. 2151	8.05	*Missing value*	S
14	0	3	Andersson, Mr. Anders	male	39.0	1	5	347082	31.275	*Missing value*	S
15	0	3	Vestrom, Miss. Hulda A	female	14.0	0	0	350406	7.8542	*Missing value*	S
16	1	2	Hewlett, Mrs. (Mary D K	female	55.0	0	0	248706	16.0	*Missing value*	S
17	0	3	Rice, Master. Eugene	male	2.0	4	1	382652	29.125	*Missing value*	Q
18	1	2	Williams, Mr. Charles Eu	male	*Missing value*	0	0	244373	13.0	*Missing value*	S
19	0	3	Vander Planke, Mrs. Juli	female	31.0	1	0	345763	18.0	*Missing value*	S
20	1	3	Masselmani, Mrs. Fatim	female	*Missing value*	0	0	2649	7.225	*Missing value*	C
21	0	2	Fynney, Mr. Joseph J	male	35.0	0	0	239865	26.0	*Missing value*	S

SELECTION OF COLUMNS WITH MISSING VALUES

```
[3]  df.info()
     ✓ 0.0s
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Cabin:
→ Derete
Age:
→ Need,
Replace
Embarcked:
→ Only 2
missing values

```
[4]  print(df.isnull().sum())
     ✓ 0.0s
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

CLEANING 1

Cabin:

→ drop column

Age:

→ Give the mean age of each by male and female and fill in the missing values according to gender.

```
df.drop(columns=['Cabin'], inplace=True)
df.info()
```

```
female_median_age = df[df['Sex'] == 'female']['Age'].median()
female_median_age
print(f"Males Age mean is {female_median_age}.")

print(f"Unique values in the 'Sex' column: {df['Sex'].unique()}")
# Confirm column name

print(f"The 'Age' column has {df['Age'].isnull().sum()} missing values.")
# Comfirm female number
```

```
male_median_age = df[df['Sex'] == 'male']['Age'].median()
male_median_age
print(f"Males Age mean is {male_median_age}.")
```

```
Males Age mean is 29.0.
```

```
df.loc[(df['Sex'] == 'male') & (df['Age'].isnull()), 'Age'] = male_median_age

print(f"""Missing values for males in the 'Age' column were filled with the mean value:
For now, the 'Age' column has {df['Age'].isnull().sum()} missing values.""")
# print(df['Age'].isnull().sum())
```

CLEANING 2

Embarked: Fill in with the most frequent value “S” .

```
[49]  df['Embarked'].describe()
```

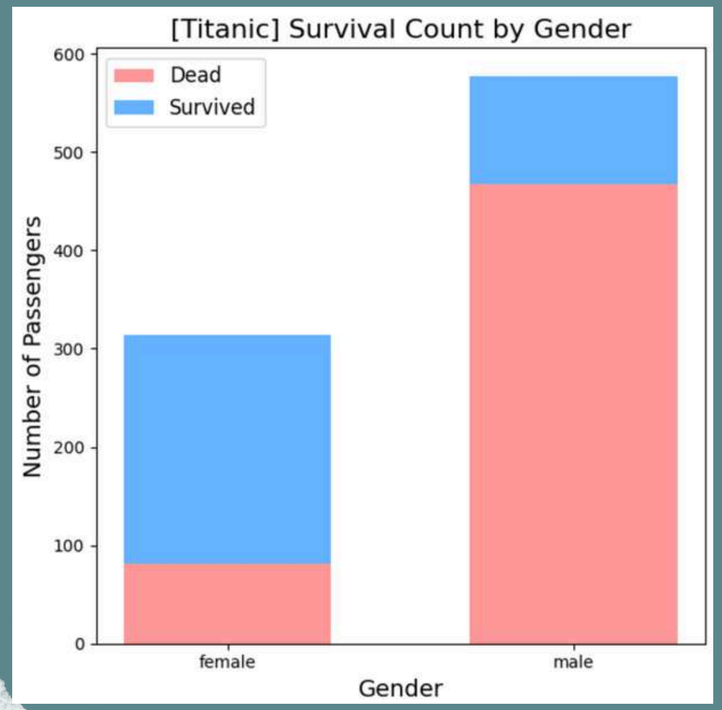
	Embarked
count	889
unique	3
top	S
freq	644

```
[51]  # Fill missing values in 'Embarked' with the most common value
      df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)

      print(f"Missing values in 'Embarked' column were filled with the most common value: {df['Embarked'].mode()[0]}.")

      print(df['Embarked'].value_counts())
      print(f"'Embarked' column is added 2 (644 -> 646).")
```

```
Missing values in 'Embarked' column were filled with the most common value: S.
Embarked
S    646
C    168
Q     77
Name: count, dtype: int64
'Embarked' column is added 2 (644 -> 646).
```

VISUALIZATION 1

A chart to show the distribution of passengers by gender (Sex) and whether they survived (Survived)



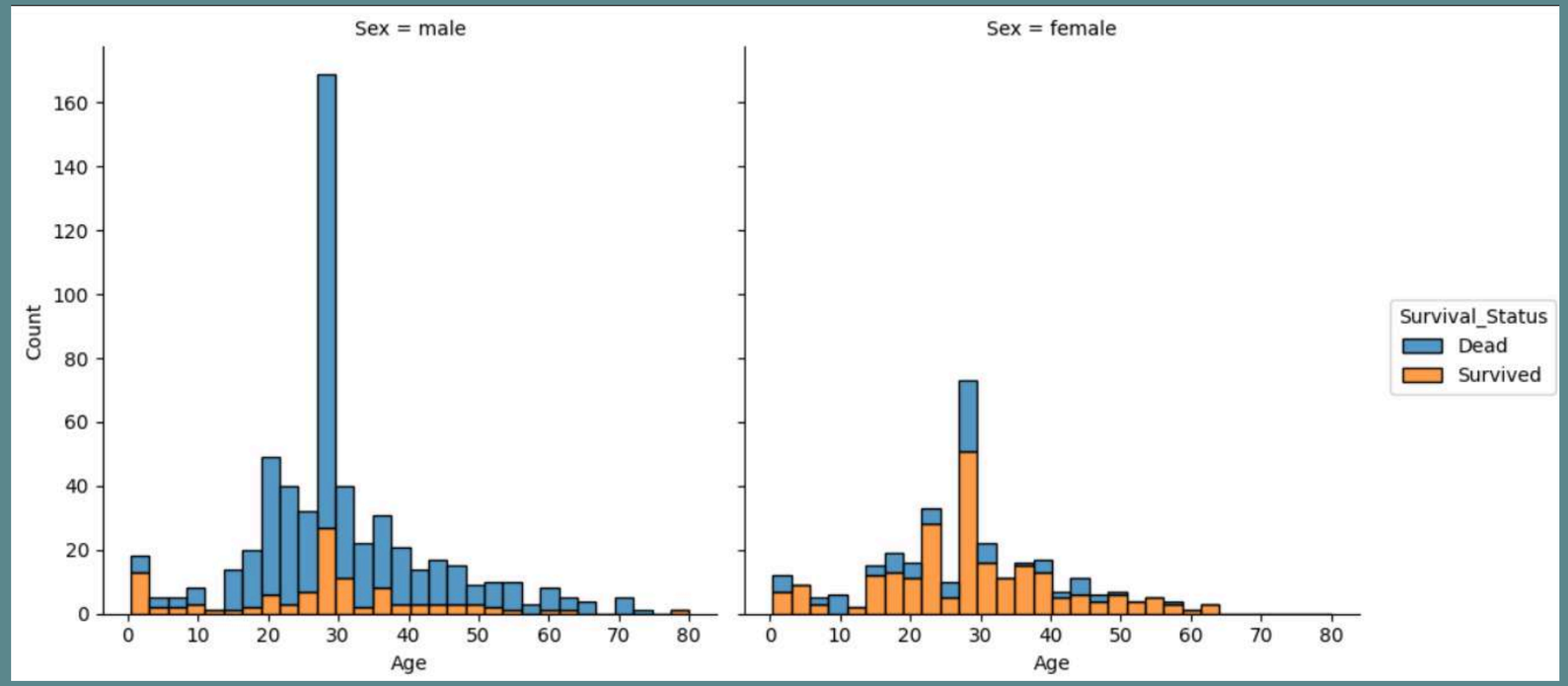

```python
grouped = df.groupby(['Sex', 'Survived']).size().unstack()
# Group by sex and survived status
plt.figure(figsize=(6, 6))

x = np.arange(len(grouped.index))
# dead
plt.bar(grouped.index,
        grouped[0],
        label='Dead',
        color='#ff9999',
        width=0.6)

# survived stacked on top of dead
plt.bar(grouped.index,
        grouped[1],
        bottom=grouped[0],
        label='Survived',
        color='#66b3ff',
        width=0.6)

plt.xticks(x, grouped.index)
plt.title('[Titanic] Survival Count by Gender', fontsize=16)
plt.xlabel('Gender', fontsize=14)
plt.ylabel('Number of Passengers', fontsize=14)
plt.legend(fontsize=12)
plt.tight_layout()
plt.show()
```

VISUALIZATION 2

10. Use a displot chart to view the age distribution of passengers by age.

```
df['Survival_Status'] = df['Survived'].map({0: 'Dead', 1: 'Survived'})
# Define 'Survival_Status'
g = sns.displot(
    data = df,
    x = 'Age',
    hue = 'Survival_Status',
    multiple = 'stack',
    kind = 'hist',
    col = 'Sex'
)
g._legend.set_frame_on(True)
```

* Separate it into two charts according to the values in the gender column
* Split the data into series according to the data in the survived column
* The data is stacked



VISUALIZATION 3

Create a correlation table by applying the corr () method to the data.

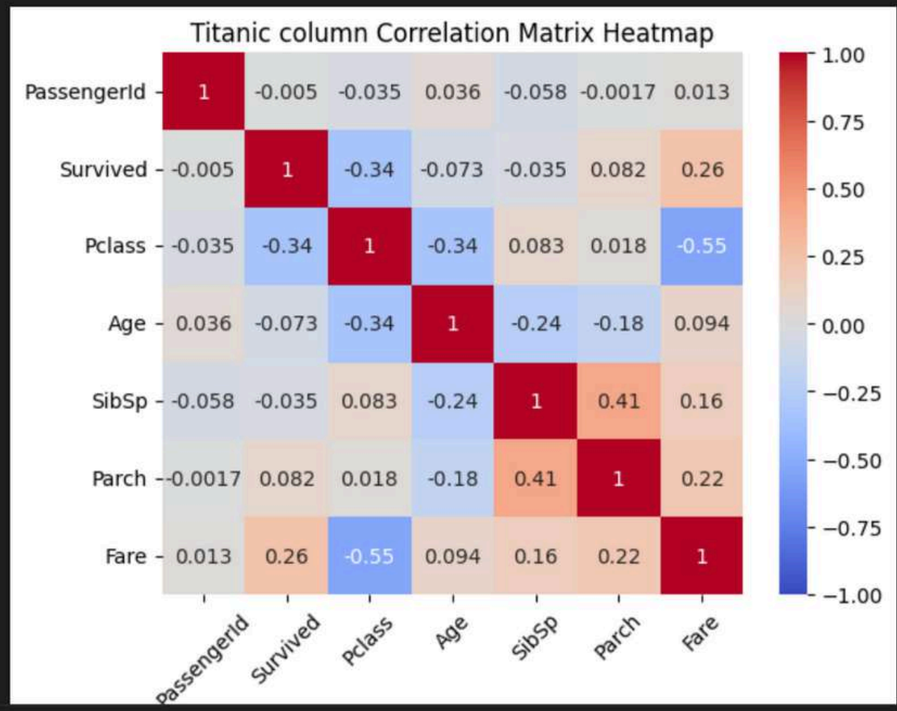
I choose “Age” and “ Survived”.

```
corr_mtx = df.corr(numeric_only=True)
print(corr_mtx)
# Age and Survived
```

[56]

```
...              PassengerId  Survived    Pclass       Age     SibSp     Parch  \
PassengerId     1.000000 -0.005007 -0.035144  0.035734 -0.057527 -0.001652
Survived       -0.005007  1.000000 -0.338481 -0.073296 -0.035322  0.081629
Pclass         -0.035144 -0.338481  1.000000 -0.338056  0.083081  0.018443
Age             0.035734 -0.073296 -0.338056  1.000000 -0.236376 -0.176038
SibSp          -0.057527 -0.035322  0.083081 -0.236376  1.000000  0.414838
Parch          -0.001652  0.081629  0.018443 -0.176038  0.414838  1.000000
Fare            0.012658  0.257307 -0.549500  0.094161  0.159651  0.216225

                 Fare
PassengerId  0.012658
Survived     0.257307
Pclass      -0.549500
Age          0.094161
SibSp        0.159651
Parch        0.216225
Fare         1.000000
```

```
sns.heatmap(corr_mtx, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.xticks(rotation=45)
plt.title("Titanic column Correlation Matrix Heatmap")
plt.show()
```



ORIGINAL QUESTION 1

Name includes title.

Is this related to age?

Title	# count
Mr	517
Miss	182
Mrs	125
Master	40
Dr	7
Rev	6
Col	2
Mlle	2
Major	2
Ms	1
Mme	1
Don	1
Lady	1
Sir	1
Capt	1
the Countess	1
Jonkheer	1

17 rows x 1 cols 25 per page

```
# "Braund, Mr. Owen Harris" → "Mr"
df['Title'] = df['Name'].str.extract(r',\s*([^\.]+)\.')
df['Title'].info()
df['Title'].value_counts()
```

Average age calculation

```
# Calculate the average age for each title (excluding NaN)
title_stats = df.groupby('Title')['Age'].agg(['mean', 'count'])
title_stats.columns = ['Age_Mean', 'Count']
title_stats
```

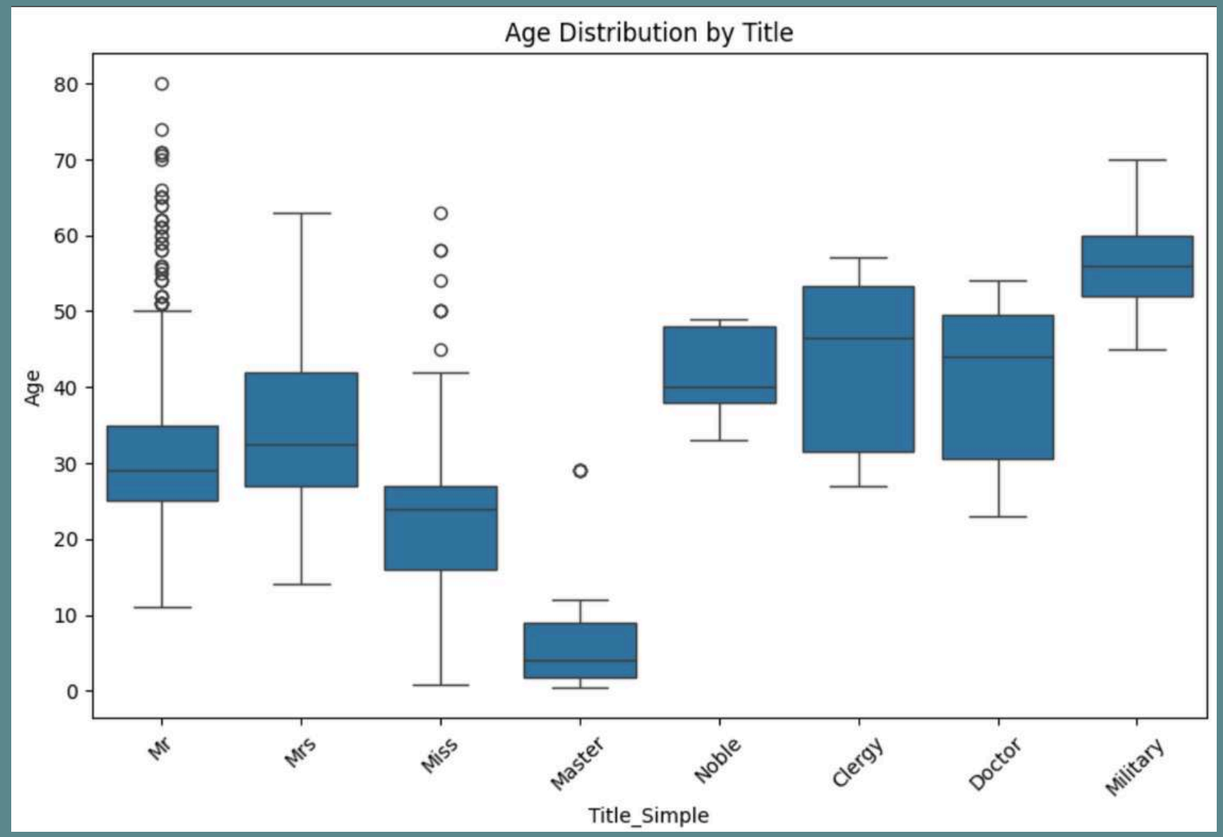
Title	# Age_Mean	# Count
Capt	70.0	1
Col	58.0	2
Don	40.0	1
Dr	40.142857142857146	7
Jonkheer	38.0	1
Lady	48.0	1
Major	48.5	2
Master	7.01675	40
Miss	22.807692307692307	182
Mlle	24.0	2
Mme	24.0	1
Mr	31.59284332688588	517
Mrs	34.688	125
Ms	28.0	1
Rev	43.166666666666664	6
Sir	49.0	1
the Countess	33.0	1

ORIGINAL QUESTION 1

Use if branches to shorten each title. Aggregate into 8 categories.

```python
# Categorized by title
def simplify_title(title):
    if title == 'Mr':
        return 'Mr'
    elif title in ['Mrs', 'Mme']:
        return 'Mrs'
    elif title in ['Miss', 'Mlle', 'Ms']:
        return 'Miss'
    elif title == 'Master':
        return 'Master'
    elif title in ['Capt', 'Col', 'Major']:
        return 'Military'
    elif title in ['Don', 'Lady', 'Sir', 'Jonkheer', 'the Countess']:
        return 'Noble'
    elif title == 'Rev':
        return 'Clergy'
    elif title == 'Dr':
        return 'Doctor'
    else:
        return 'Other'

df['Title_Simple'] = df['Title'].apply(simplify_title)

df['Title_Simple'].value_counts()
```

```python
plt.figure(figsize=(10,6))
sns.boxplot(data=df, x='Title_Simple', y='Age')
plt.xticks(rotation=45)
plt.title('Age Distribution by Title')

plt.show()
```



ORIGINAL QUESTION 2

Are there differences in survival rates based on age by title, gender?

```python
# Recalculating the average age by title
title_simple_stats = df.groupby('Title_Simple')['Age'].agg(['mean', 'count']).sort_values('count', ascending=False)
title_simple_stats.columns = ['Age_Mean', 'Count']
print(title_simple_stats)
```

Python

Title_Simple	Age_Mean	Count
Mr	31.592843	517
Miss	22.848649	185
Mrs	34.603175	126
Master	7.016750	40
Doctor	40.142857	7
Clergy	43.166667	6
Military	56.600000	5
Noble	41.600000	5

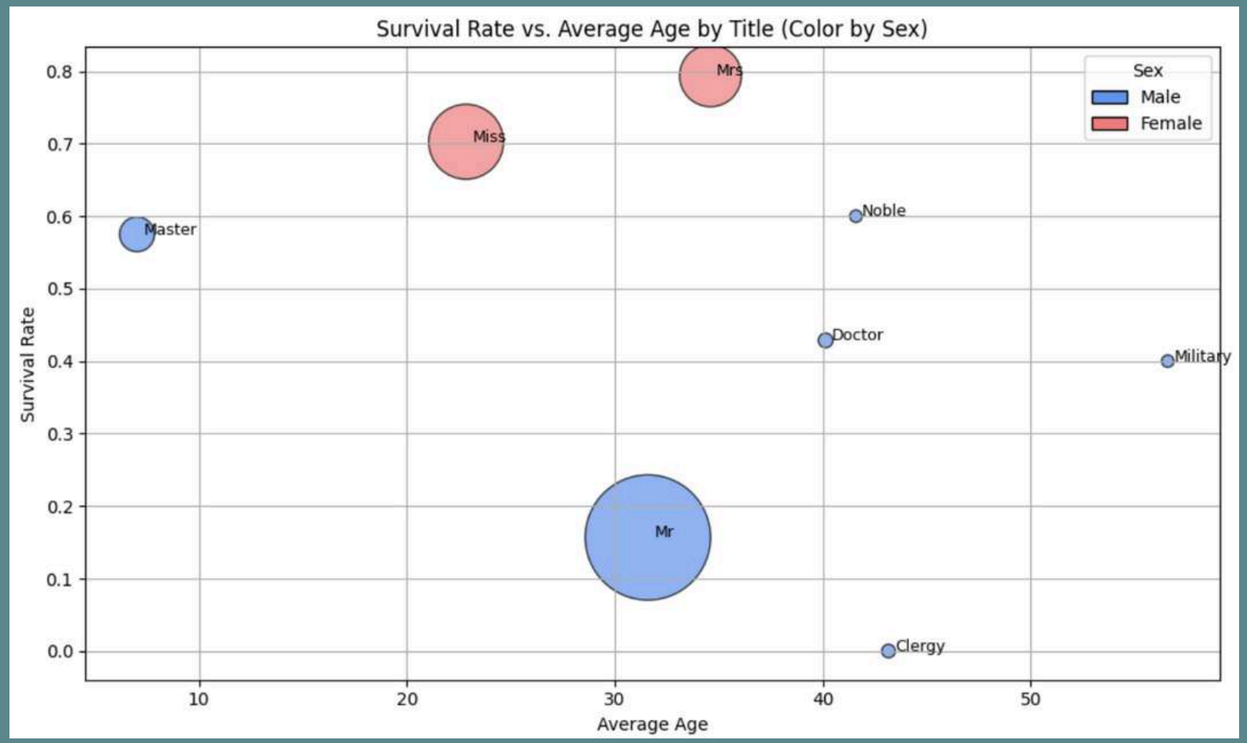
Give the average age of each title.

Make a bubble chart of survival rate, title, and age. The size of the bubble is the number of people.

ORIGINAL Q2

```python
title_sex = df.groupby('Title_Simple')['Sex'].agg(lambda x: x.mode()[0])

# Agg: Title_Simple per count, survive, number of person
title_stats = df.groupby('Title_Simple').agg({
    'Age': 'mean',
    'Survived': 'mean',
    'PassengerId': 'count'
}).rename(columns={
    'Age': 'Age_Mean',
    'Survived': 'Survival_Rate',
    'PassengerId': 'Count'
})

# Merge the title stats with the representative
title_stats['Sex'] = title_stats.index.map(title_sex)

# Define colors based on sex
color_map = {'male': 'cornflowerblue', 'female': 'lightcoral'}
colors = title_stats['Sex'].map(color_map)

# bobble chart
plt.figure(figsize=(10, 6))
plt.scatter(
    title_stats['Age_Mean'],
    title_stats['Survival_Rate'],
    s=title_stats['Count'] * 10,
    c=colors,
    alpha=0.7,
    edgecolors='black'
)
```

```python
# A process to display the "title name (Mr, Miss, etc.)" next to each bubble in a scatter plot
for title, row in title_stats.iterrows():
    plt.text(row['Age_Mean'] + 0.3, row['Survival_Rate'], title, fontsize=9)

plt.xlabel('Average Age')
plt.ylabel('Survival Rate')
plt.title('Survival Rate vs. Average Age by Title (Color by Sex)')
plt.grid(True)

from matplotlib.patches import Patch
legend_elements = [Patch(facecolor='cornflowerblue', edgecolor='black', label='Male'),
                   Patch(facecolor='lightcoral', edgecolor='black', label='Female')]
plt.legend(handles=legend_elements, title='Sex')

plt.tight_layout()
plt.show()
```



ORIGINAL QUESTION 3

Does the survival rate differ depending on the room (including fare)?

It is predicted that groups of family and friends will be in the same cabin class or have similar ticket numbers.
Even in the same third-class cabin, fares vary depending on location and size.

```
[69]
# STEP1 : Group by same ticket ID
df['Ticket'].value_counts()
```

Ticket	# count
347082	7
1601	7
CA. 2343	7
3101295	6
CA 2144	6
347088	6
382652	5
S.O.C. 14879	5
113760	4
19950	4
349909	4
347077	4
4133	4

Calculate per capita amount

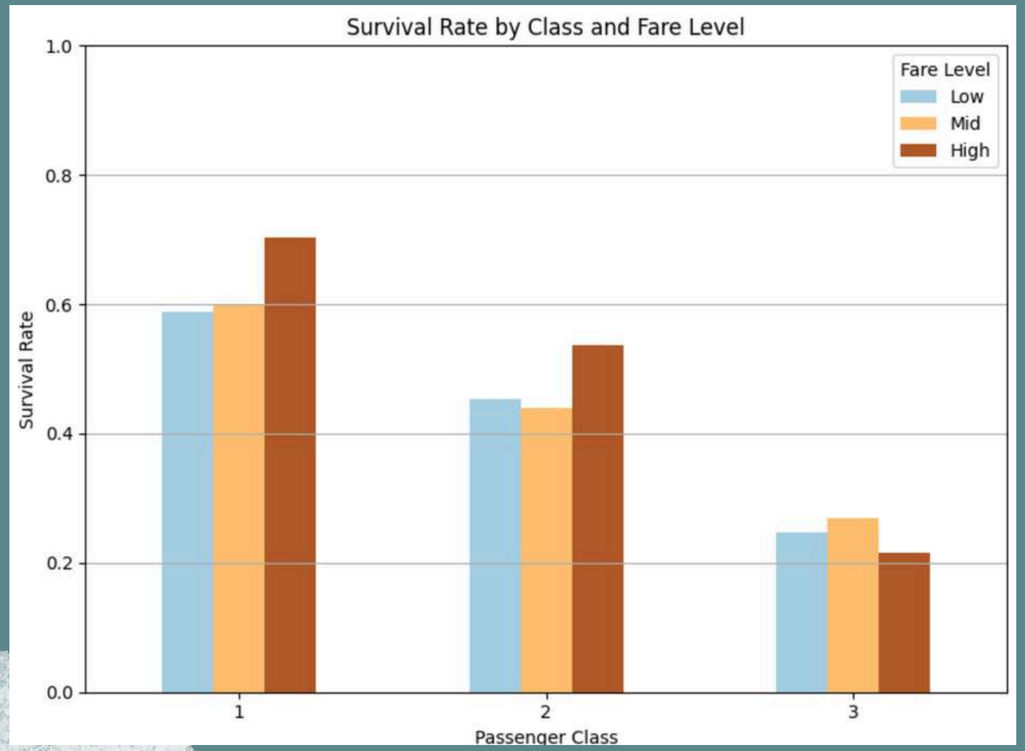
```
# STEP2: Cariculate the average fare for each group
df['Fare_per_person'] = df['Fare'] / df['GroupSize']
df[['Ticket', 'Pclass', 'Fare', 'GroupSize', 'Fare_per_person']].head(10)
```

	Ticket	# Pclass	# Fare
0	A/5 21171	3	7.25
1	PC 17599	1	71.2833
2	STON/O2. 3101282	3	7.925
3	113803	1	53.1
4	373450	3	8.05
5	330877	3	8.4583
6	17463	1	51.8625
7	349909	3	21.075
8	347742	3	11.1333
9	237736	2	30.0708

ORIGINAL QUESTION 3

In each graded cabin (1, 2, 3) High, medium and low fares, their survival probability.

Higher fares also have a higher survival rate in first and second class cabins



Graded by room level

```
# STEP3: Fare_per_person is ranked into three levels within the same Pclass
df['Fare_Level'] = df.groupby('Pclass')['Fare_per_person'].transform(
    lambda x: pd.qcut(x, q=3, labels=['Low', 'Mid', 'High'])
)
df[['Pclass', 'Fare_per_person', 'Fare_Level']].head(10)
```

	# Pclass	# Fare_per_person	Fare_Level
0	3	7.25	Low
1	1	71.2833	High
2	3	7.925	Mid
3	1	26.55	Low
4	3	8.05	High
5	3	8.4583	High
6	1	51.8625	High
7	3	5.26875	Low
8	3	3.7111	Low
9	2	15.0354	High

10 rows x 3 cols 10 per page Page 1 of 1

Plot

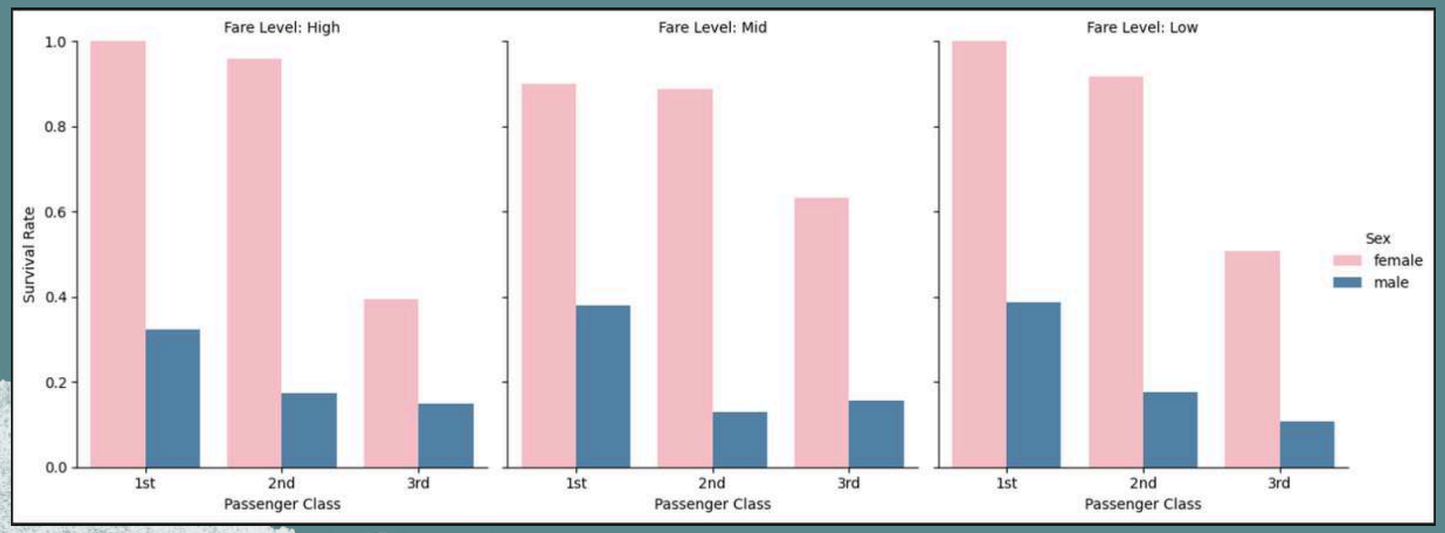
```
# STEP4: Illustrate the bar chart
survival_by_fare = df.groupby(['Pclass', 'Fare_Level'])['Survived'].mean().unstack()

survival_by_fare.plot(kind='bar',
                      figsize=(8, 6),
                      colormap='Paired',
                      rot=0)
plt.title('Survival Rate by Class and Fare Level')
plt.ylabel('Survival Rate')
plt.xlabel('Passenger Class')
plt.legend(title='Fare Level')
plt.ylim(0, 1)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

ORIGINAL QUESTION 3

Plotting survival rate by Pclass, Fare Level, and Sex

```python
# Calculate average survival rate grouped by Pclass, Fare_Level, and Sex
grouped = df.groupby(['Pclass', 'Fare_Level', 'Sex'])['Survived'].mean().reset_index()

# Create bar plots for each Fare_Level, showing survival rate by Pclass and separated by Sex
g = sns.catplot(
    data=grouped,
    x='Pclass',
    y='Survived',
    hue='Sex',
    col='Fare_Level',
    col_order=['High', 'Mid', 'Low'],
    kind='bar',
    height=5,
    aspect=0.8,
    palette={'female': 'lightpink', 'male': 'steelblue'},
    ci=None
)
```

```python
# Customize plot appearance
g.set_titles('Fare Level: {col_name}')
g.set_axis_labels('Passenger Class', 'Survival Rate')
g.set(ylim=(0, 1))

# Customize x-axis labels
for ax in g.axes.flatten():
    ax.set_xticklabels(['1st', '2nd', '3rd'])

# Move legend outside the plot
g._legend.set_bbox_to_anchor((1.05, 0.5))
g._legend.set_loc('center right')

plt.tight_layout()
plt.show()
```



- All cabin levels have higher survival rates for women.
- Cheaper 1st class fares have lower survival rates for women than other cabin levels

THANK YOU