5CS037 - Concepts and Technologies of AI.
Exploratory Data Analysis - Part -II.
Advance operations with Pandas and Review of Matplotlib.

Prepared By: Siman Giri {Module Leader - 5CS037}

November 26, 2025

1 Instructions

{Disclaimer:Exploratory Data Analysis is designed to be two part exercise and this is Part 2, which mostly focuses on advance operation with pandas for sorting and subsetting of data and applying group-by method on data for exploration and analysis of data.This worksheet also provides a reference on Matplotlib to revise your understanding from Level 4 Computational Mathematics Course.}

This worksheet contains programming exercises on Data cleaning and Data Transformation with pandas based on the material discussed from the slides.This is not a graded exercise and submission are optional but highly recommended as it will be the base of your first assignment.
Please answer the questions below using python in the Jupyter Notebook and follow the guidelines below:

- This worksheet must be completed individually.
- All the solutions must be written in Jupyter Notebook.
- Our Recommendation - Google Colaboratory.
- Dataset used for this session can be downloaded from shared drive.

Figure 1: Pandas.

————————– You have to think about one thing: what's that information worth?– (Moneyball) ————————-

2 Advance Operations with Pandas.

This Section contains all the sample code from the slides and are here for your reference, you are highly recommended to run all the code with some of the input changed in order to understand the meaning of the operations and also to be able to solve all the exercises from further sections.

- **Cautions!!!:**
  - This Guide may not contain sample output, as we expect you to re-write the code and observe the output.
  - If found: any error or bugs, please report to your instructor and Module leader. {Will hugely appreciate your effort.}

2.1 Sorting and Subsetting:

1. **Sorting:**

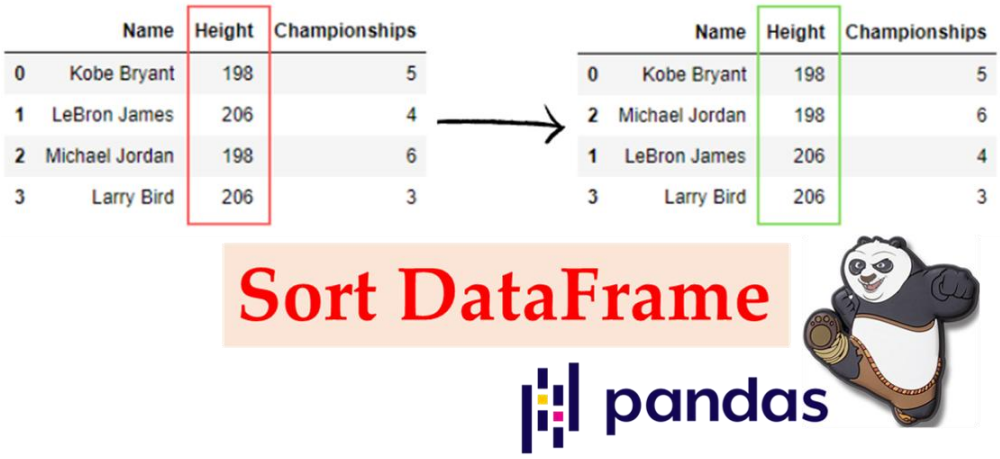


Figure 2: Sorting of Dataframe

Sample Code from Slide - 6 - Sorting Example.

```
import pandas as pd
# Creating a sample DataFrame data = {'Name': ['Alice', 'Bob',
'Charlie', 'David'],
        'Age': [24, 19, 22, 25],
        'Score': [88, 92, 85, 95]} df =
pd.DataFrame(data) print(df.head())
# Example 1: Sort by 'Age' using sort_values() sorted_by_age =
df.sort_values(by='Age') print(sorted_by_age.head())
# Example 2: Sort by index using sort_index() sorted_by_index =
df.sort_index() print(sorted_by_index.head())
```

2. Subsetting - Indices:

Sample Code from Slide - 7 - 8 - Subsetting by indices.

```
import pandas as pd
# Creating a sample DataFrame data = {'Name': ['Alice', 'Bob',
'Charlie', 'David'],
        'Age': [24, 19, 22, 25],
        'Score': [88, 92, 85, 95]} df =
pd.DataFrame(data)
# 1.Using iloc[]:Accessing rows and columns by index subset_iloc =
df.iloc[1:3, 0:2] print(subset_iloc)
# 2.Using loc[]:Accessing rows by condition and specific columns subset_loc =
df.loc[df['Age'] > 20, ['Name', 'Score']] print(subset_loc)
# 3.Using []: Selecting specific columns subset_brackets = df[['Name',
'Age']] print(subset_brackets)
```

3. Subsetting by Values - Columns:

Sample Code from Slide - 9 - Subsetting by Columns.

```
import pandas as pd
# Creating a sample DataFrame data = {'Name': ['Alice', 'Bob',
'Charlie', 'David'],
        'Age': [24, 19, 22, 25],
        'Score': [88, 92, 85, 95]} df =
pd.DataFrame(data) # Subsetting a single
column name_column = df['Name']
print(name_column)
# Subsetting multiple columns name_and_age =
df[['Name', 'Age']] print(name_and_age)
```

4. Subsetting By Rows - {aka Filtering}:

Sample Code from Slide - 10 - Subsetting by Row.

```
import pandas as pd
# df: Dataframe from slide 09
# Filter rows with a single condition (Age > 20) filtered_single =
df[df['Age'] > 20] print(filtered_single)
# Filter rows with multiple conditions (Age > 20 and Score > 85) filtered_multiple = df[(df['Age'] > 20) &
(df['Score'] > 85)] print(filtered_multiple)
```

5. Filtering on Categorical Values:

Sample Code from Slide - 11 - 12 - Filtering on Categorical Values.

```
#Transforming in-built data structures-DataFrame
#Style-1 import pandas as pd pd.DataFrame({'Bob': ['I liked it.','It was awful'], 'Sue': ['Pretty good.', 'Bland.']})
#Style-2 pd.DataFrame({'Bob': ['I liked it.', 'It was awful.'], 'Sue': ['Pretty good.', 'Bland.']},
                index=['Product A', 'Product B'])
```

2.2 The Group-By Method:

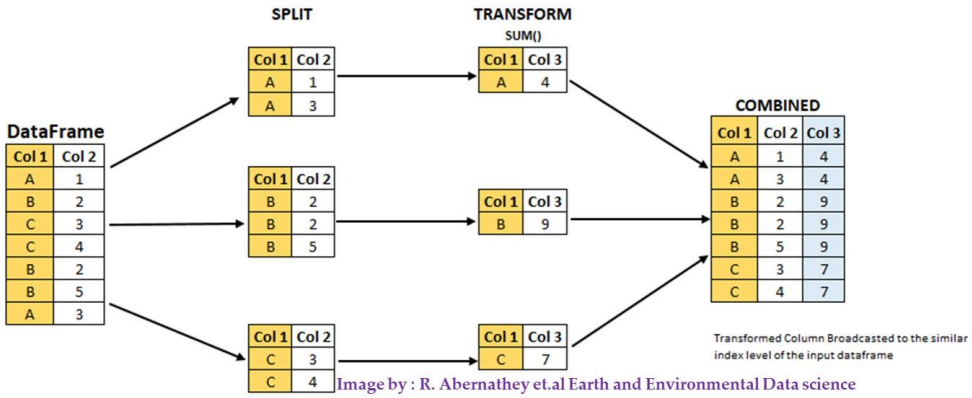


Image by : R. Abernathey et.al Earth and Environmental Data science

Figure 3: Group By: Split - Apply - Combined.

1. Split - Apply - Combined.

Sample Code from Slide - 16 - Split - Apply{Aggregation} - Combined.

```
import pandas as pd # Sample DataFrame data =
{'Category': ['A', 'B', 'A', 'B', 'A'],
        'Value': [10, 20, 30, 40, 50]} df =
pd.DataFrame(data)
# Aggregation: Calculate mean for each group grouped =
df.groupby('Category')['Value'].mean() print(grouped)
```

Sample Code from Slide - 17 - Split - Apply{Transformation} - Combined.

```
import pandas as pd # Sample DataFrame data =
{'Category': ['A', 'B', 'A', 'B', 'A'],
        'Value': [10, 20, 30, 40, 50]} df =
pd.DataFrame(data)
# Transformation: Normalize values within each group
df['Normalized'] = df.groupby('Category')['Value'].transform(lambda x: x / x.sum()) print(df)
```

Sample Code from Slide - 18 - Split - Apply{Filtration} - Combined.

```
import pandas as pd # Sample DataFrame data =
{'Category': ['A', 'B', 'A', 'B', 'A'],
        'Value': [10, 20, 30, 40, 50]} df =
pd.DataFrame(data)
# Filtration: Keep groups where sum of values > 60 filtered =
df.groupby('Category').filter(lambda x: x['Value'].sum() > 60) print(filtered)
```

2.3 Summary - Some Advance Operations with Pandas:

Method/Function	Syntax
sort_values()	df.sort_values(by='column_name', ascending=True)
sort_index()	df.sort_index()
head()	df.head(n)
tail()	df.tail(n)
iloc[]	df.iloc[rows, columns]
loc[]	df.loc[condition]
[] (brackets)	df['column_name']
df[df['column_name'] > value]	
groupby()	df.groupby('column_name')
sum()	df.groupby('column_name')['value_column'].sum()
mean()	df.groupby('column_name')['value_column'].mean()
count()	df.groupby('column_name')['value_column'].count()
transform()	df.groupby('column_name')['value_column'].transform(lambda x: x - x.mean()
apply()	df.groupby('column_name').apply(custom_function)
filter()	df.groupby('column_name').filter(lambda x: x['value_column'].sum() > 100)
agg()	df.groupby('column_name').agg('value_column': ['sum', 'mean'])
size()	df.groupby('column_name').size()
isin()	df[df['column_name'].isin([value1, value2])]
pd.cut()	df['new_column'] = pd.cut(df['column_name'], bins, labels)

Table 1: Summary of Common Data Manipulation Methods and Functions in Pandas

Figure 4: Summary Table.

2.4 Data Visualization with Pandas:

1. Line Plot - Barchart - Histogram - Scatter - Boxplot:

Sample Code from Slide - 27 to 31 - Various Plots.

```
import pandas as pd import
matplotlib.pyplot as plt
# Sample Data data = {'Month': ['Jan', 'Feb', 'Mar', 'Apr'],
        'Sales': [200, 220, 250, 280]} df =
pd.DataFrame(data)
# Line Plot df.plot(x='Month', y='Sales', kind='line', marker='o', title='Monthly Sales') plt.show()
#Bar chart # Sample Data data = {'Category': ['A', 'B', 'C'],
        'Values': [10, 20, 15]} df = pd.DataFrame(data) df.plot(x='Category', y='Values', kind='bar', title='Category
Comparison', color='skyblue') plt.show()
# Sample Data - Histogram data = {'Scores': [50, 60, 70, 75, 80, 85, 90,
95, 100]} df = pd.DataFrame(data)
# Histogram df['Scores'].plot(kind='hist', bins=5, title='Score Distribution', color='orange') plt.show()
# Sample Data - scatter plot data = {'Height': [150,
160, 170, 180],
        'Weight': [50, 60, 70, 80]} df =
pd.DataFrame(data)
# Scatter Plot df.plot(x='Height', y='Weight', kind='scatter', title='Height vs Weight')
plt.show()
# Sample Data data = {'Scores': [50, 60, 70, 75, 80, 85, 90, 95, 100, 105]} df =
pd.DataFrame(data)
# Box Plot
df.boxplot(column='Scores') plt.title('Score
Distribution') plt.show()
```

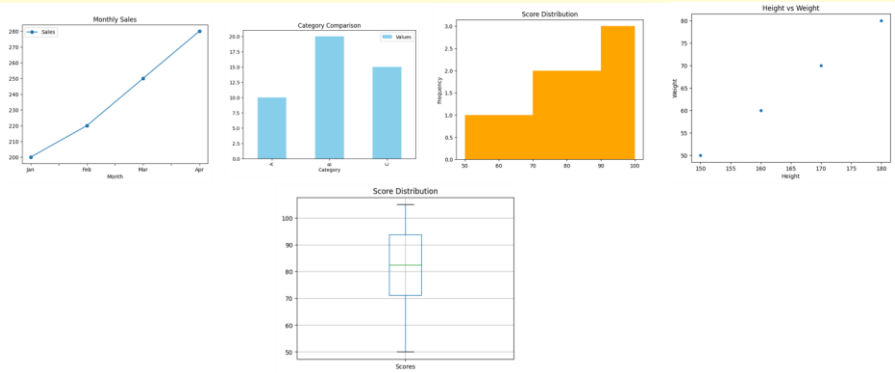


Figure 5: Sample Outputs - Line Chart - Barchart - Histogram - Scatter - Boxplot

2.5 Data Visualization with Matplotlib:

1. Plotting Your First Figure

- **Style:1**

Sample Code from Slide - 35 - Plotting Your First Figure.

```
import matplotlib.pyplot as plt days =
["Mon","Tue","Wed","Thu","Fri","Sat","Sun
"] steps_walked =
[8934,14902,3409,25672,12300,2023,6890]
plt.plot(steps_walked)
```

- **Style:2**

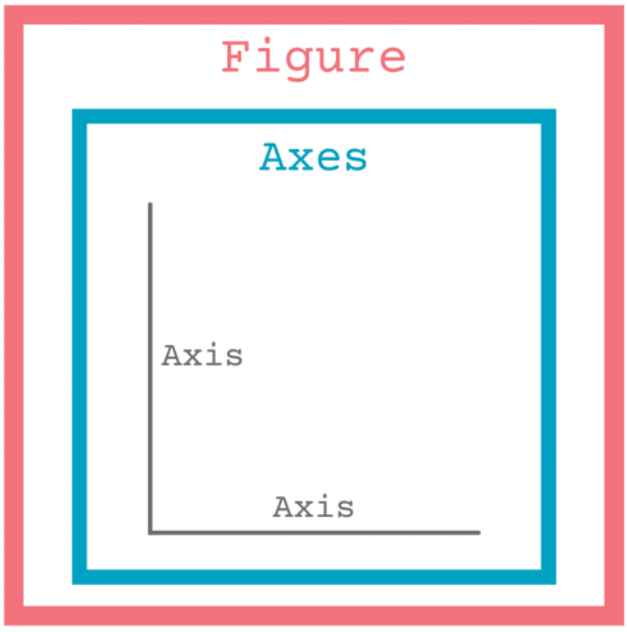
Sample Code from Slide - 35 - Plotting Your First Figure.

```
import matplotlib.pyplot as plt days = ["Mon",
"Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
steps_walked = [8934, 14902, 3409, 25672, 12300,
2023, 6890] plt.plot(days, steps_walked) plt.show()
```

Observe the difference between above two outputs.

2. Anatomy of Matplotlib Figure

When working with data visualization in Python, you'll want to have control over all aspects of your figure.
In this section, you'll learn about the main components that make up a figure in Matplotlib.
Everything in



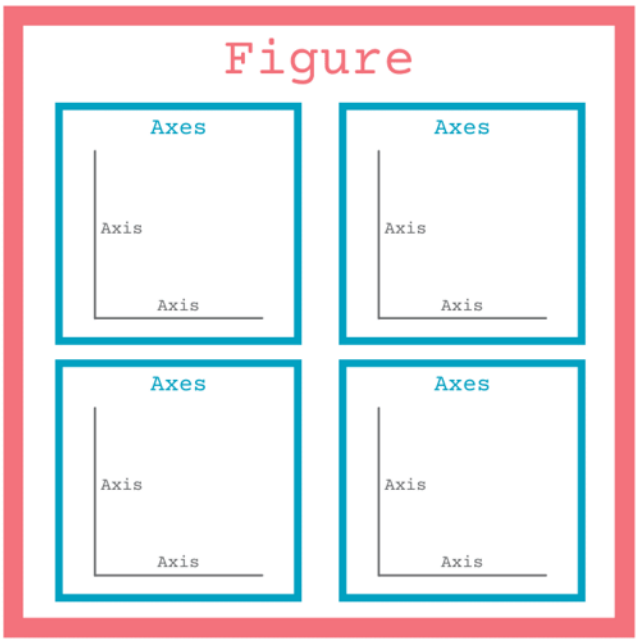


Figure 6: Components of Matplotlib Figure

Python is an object, and therefore, so is a Matplotlib figure. In fact, a Matplotlib figure is made up of several objects of different data types. There are three main parts to a Matplotlib figure:

- **Figure:** This is the whole region of space that's created when you create any figure. The Figure object is the overall object that contains everything else.
- **Axes:** An Axes object is the object that contains the x-axis and y-axis for a 2D plot. Each Axes object corresponds to a plot or a graph. You can have more than one Axes object in a Figure, as you'll see later on in this Chapter.
- **Axis:**An Axis object contains one of the axes, the x-axis or the y-axis for a 2D plot.

Customizing the plots

3. Add a custom marker

Sample Code from Slide - 38 - Add a Custom Marker.

```
import matplotlib.pyplot as plt days =
["Mon","Tue","Wed","Thu","Fri","Sat","Sun
"] steps_walked =
[8934,14902,3409,25672,12300,2023,6890]
plt.plot(days, steps_walked, "o") plt.show()
```

Cautions:Please consult matplotlib documentation for updated version and type of marker available.

4. Adding titles,labels and legends.

Sample Code from Slide - 39 - Adding titles - -

```
import matplotlib.pyplot as plt days =
["Mon","Tue","Wed","Thu","Fri","Sat","Sun"] steps_walked =
[8934,14902,3409,25672,12300,2023,6890] steps_last_week =
[9788,8710,5308,17630,21309,4002,5223] plt.plot(days, steps_walked,
"o-g") plt.plot(days, steps_last_week, "v--m") plt.title("Step count | This
week and last week") plt.xlabel("Days of the week") plt.ylabel("Steps
walked") plt.grid(True)
plt.legend(["This week", "Last week"]) plt.show()
```

Observe the output.

5. Creating a Subplots

Sample Code from Slide - 40 - Creating Subplot.

```
def f(t):
    return np.exp(-t) * np.cos(2*np.pi*t)
t1 = np.arange(0.0, 5.0, 0.1) t2 =
np.arange(0.0, 5.0, 0.02) plt.figure()
plt.subplot(211)
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k') plt.subplot(212)

plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.show()
```

Observe and find what are the arguments for plt.subplot().

3 To - Do - Task

Please Complete all the problem listed below.

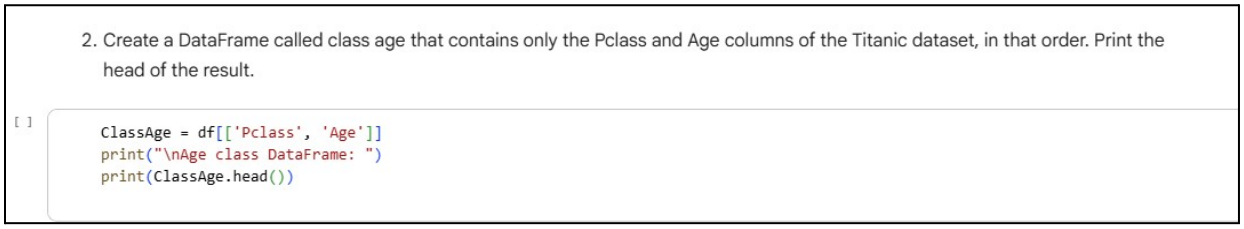
Worksheet

Problem 1 - Sorting:

1. Create a DataFrame called fare that contains only the Fare column of the Titanic dataset. Print the head of the result.

```
[2] import pandas as pd

Problem 1 - Sorting:

1. Create a DataFrame called fare that contains only the Fare column of the Titanic dataset. Print the head of the result.

[5] df = pd.read_csv('/content/drive/MyDrive/Concept and Technology Of AI/Week2/Titanic-Dataset.csv')
fare = df[['Fare']]
print("Fare DataFrame head: ")
print(fare.head())

Fare DataFrame head:
      Fare
0   7.2500
1  71.2833
2   7.9250
3  53.1000
4   8.0500

Age class DataFrame:
   Pclass   Age
0       3  22.0
1       1  38.0
2       3  26.0
3       1  35.0
4       3  35.0

SUrvived gender DataFrame:
   Survived     Sex
0         0    male
1         1  female
2         1  female
3         1  female
4         0    male
```

2. Create a DataFrame called class age that contains only the Pclass and Age columns of the Titanic dataset, in that order. Print the head of the result.



```
2. Create a DataFrame called class age that contains only the Pclass and Age columns of the Titanic dataset, in that order. Print the head of the result.

[ ] ClassAge = df[['Pclass', 'Age']]
print("\nAge class DataFrame: ")
print(ClassAge.head())
```

3. Create a DataFrame called survived gender that contains the Survived and Sex columns of the

Titanic dataset, in that order.Print the head of the result.

3. Create a DataFrame called survived gender that contains the Survived and Sex columns of the Titanic dataset, in that order.Print the head of the result

```
[14]
✓ 0s
SurvivedGender = df[['Survived', 'Sex']]
print("\nSUrvived gender DataFrame: ")
print(SurvivedGender.head())
```

```
SUrvived gender DataFrame:
   Survived     Sex
0         0    male
1         1  female
2         1  female
3         1  female
4         0    male
```

Problem - 2 - Subsetting:

Complete all the following Task:

Subsetting Rows:

1. Filter the Titanic dataset for cases where the passenger's fare is greater than 100, assigning it to fare gt 100.View the printed result.

Problem - 2 - Subsetting: Complete all the following Task: Subsetting Rows:

1. Filter the Titanic dataset for cases where the passenger's fare is greater than 100, assigning it to fare gt 100.View the printed result.

```
[15]
✓ 0s
df=pd.read_csv('/content/drive/MyDrive/Concept and Technology Of AI/Week2/Titanic-Dataset.csv')
fare_gt_100=df[df['Fare']>100]
print("Passengers with fare > 100:")
print(fare_gt_100.head())
```

```
Passengers with fare > 100:
     PassengerId  Survived  Pclass  \
27            28         0       1
31            32         1       1
88            89         1       1
118          119         0       1
195          196         1       1

                                          Name     Sex   Age  SibSp  \
27                    Fortune, Mr. Charles Alexander    male  19.0      3
31   Spencer, Mrs. William Augustus (Marie Eugenie)  female   NaN      1
88                       Fortune, Miss. Mabel Helen  female  23.0      3
118                        Baxter, Mr. Quigg Edmond    male  24.0      0
195                            Lurette, Miss. Elise  female  58.0      0

     Parch    Ticket      Fare        Cabin Embarked
27       2     19950  263.0000  C23 C25 C27        S
31       0  PC 17569  146.5208          B78        C
88       2     19950  263.0000  C23 C25 C27        S
118      1  PC 17558  247.5208      B58 B60        C
195      0  PC 17569  146.5208          B80        C
```

2. Filter the Titanic dataset for cases where the passenger's class (Pclass) is 1, assigning it to first class.View the printed result.

2. Filter the Titanic dataset for cases where the passenger's class (Pclass) is 1, assigning it to first class.View the printed result.

[16] ✓ 0s

```
first_class=df[df['Pclass']==1]
print("\nPassengers in first class:")
print(first_class.head())
```

```
Passengers in first class:
    PassengerId  Survived  Pclass  \
1             2         1       1
3             4         1       1
6             7         0       1
11           12         1       1
23           24         1       1

                                                 Name     Sex   Age  SibSp  \
1   Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
3        Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
6                             McCarthy, Mr. Timothy J    male  54.0      0
11                           Bonnell, Miss. Elizabeth  female  58.0      0
23                        Sloper, Mr. William Thompson    male  28.0      0

    Parch    Ticket     Fare Cabin Embarked
1       0  PC 17599  71.2833   C85        C
3       0    113803  53.1000  C123        S
6       0     17463  51.8625   E46        S
11      0    113783  26.5500  C103        S
23      0    113788  35.5000    A6        S
```

3. Filter the Titanic dataset for cases where the passenger's age is less than 18 and the passenger is female (Sex is "female"), assigning it to female under 18.
View the printed result.

3. Filter the Titanic dataset for cases where the passenger's age is less than 18 and the passenger is female (Sex is "female"), assigning it to female under 18. View the printed result.

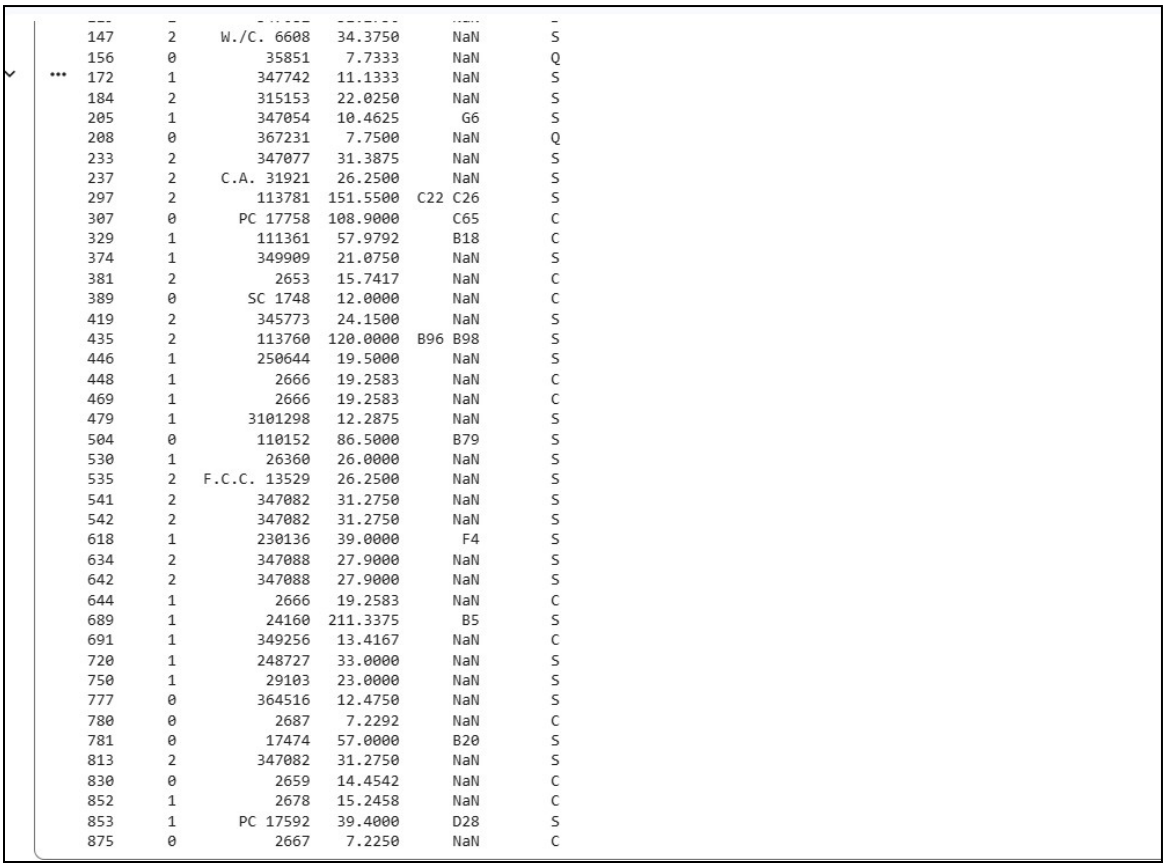
[17] ✓ 0s

```
female_under_18=df[(df['Sex']=='female') & (df['Age']<18)]
print(female_under_18)
```

```
     PassengerId  Survived  Pclass  \
9             10         1       2
10            11         1       3
14            15         0       3
22            23         1       3
24            25         0       3
39            40         1       3
43            44         1       2
58            59         1       2
68            69         1       3
71            72         0       3
84            85         1       2
111          112         0       3
114          115         0       3
119          120         0       3
147          148         0       3
156          157         1       3
172          173         1       3
184          185         1       3
205          206         0       3
208          209         1       3
233          234         1       3
237          238         1       2
297          298         0       1
307          308         1       1
329          330         1       1
374          375         0       3
381          382         1       3
389          390         1       2
419          420         0       3
435          436         1       1
446          447         1       2
448          449         1       3
469          470         1       3
479          480         1       3
504          505         1       1
530          531         1       2
535          536         1       2
541          542         0       3
542          543         0       3
618          619         1       2
634          635         0       3
```

```
     642          643         0       3
     644          645         1       3
...  689          690         1       1
     691          692         1       3
     720          721         1       2
     750          751         1       2
     777          778         1       3
     780          781         1       3
     781          782         1       1
     813          814         0       3
     830          831         1       3
     852          853         0       3
     853          854         1       1
     875          876         1       3

                                                 Name     Sex    Age  SibSp  \
9                    Nasser, Mrs. Nicholas (Adele Achem)  female  14.00      1
10                       Sandstrom, Miss. Marguerite Rut  female   4.00      1
14                   Vestrom, Miss. Hulda Amanda Adolfina  female  14.00      0
22                           McGowan, Miss. Anna "Annie"  female  15.00      0
24                          Palsson, Miss. Torborg Danira  female   8.00      3
39                            Nicola-Yarred, Miss. Jamila  female  14.00      1
43               Laroche, Miss. Simonne Marie Anne Andree  female   3.00      1
58                           West, Miss. Constance Mirium  female   5.00      1
68                     Andersson, Miss. Erna Alexandra  female  17.00      4
71                          Goodwin, Miss. Lillian Amy  female  16.00      5
84                                  Ilett, Miss. Bertha  female  17.00      0
111                                Zabour, Miss. Hileni  female  14.50      1
114                               Attalah, Miss. Malake  female  17.00      0
119                  Andersson, Miss. Ellis Anna Maria  female   2.00      4
147                    Ford, Miss. Robina Maggie "Ruby"  female   9.00      2
156                    Gilnagh, Miss. Katherine "Katie"  female  16.00      0
172                        Johnson, Miss. Eleanor Ileen  female   1.00      1
184                 Kink-Heilmann, Miss. Luise Gretchen  female   4.00      0
205                          Strom, Miss. Telma Matilda  female   2.00      0
208                           Carr, Miss. Helen "Ellen"  female  16.00      0
233                     Asplund, Miss. Lillian Gertrud  female   5.00      4
237                    Collyer, Miss. Marjorie "Lottie"  female   8.00      0
297                        Allison, Miss. Helen Loraine  female   2.00      1
307  Penasco y Castellana, Mrs. Victor de Satode (M...  female  17.00      1
329                          Hippach, Miss. Jean Gertrude  female  16.00      0
374                            Palsson, Miss. Stina Viola  female   3.00      3
381                          Nakid, Miss. Maria ("Mary")  female   1.00      0
```

```
     389                              Lehmann, Miss. Bertha  female  17.00      0
     419                         Van Impe, Miss. Catharina  female  10.00      0
...  435                          Carter, Miss. Lucile Polk  female  14.00      1
     446                  Mellinger, Miss. Madeleine Violet  female  13.00      0
     448                     Baclini, Miss. Marie Catherine  female   5.00      2
     469                      Baclini, Miss. Helene Barbara  female   0.75      2
     479                           Hirvonen, Miss. Hildur E  female   2.00      0
     504                              Maioni, Miss. Roberta  female  16.00      0
     530                            Quick, Miss. Phyllis May  female   2.00      1
     535                              Hart, Miss. Eva Miriam  female   7.00      0
     541                Andersson, Miss. Ingeborg Constanzia  female   9.00      4
     542                   Andersson, Miss. Sigrid Elisabeth  female  11.00      4
     618                         Becker, Miss. Marion Louise  female   4.00      2
     634                                  Skoog, Miss. Mabel  female   9.00      3
     642                      Skoog, Miss. Margit Elizabeth  female   2.00      3
     644                           Baclini, Miss. Eugenie  female   0.75      2
     689                    Madill, Miss. Georgette Alexandra  female  15.00      0
     691                                 Karun, Miss. Manca  female   4.00      0
     720                  Harper, Miss. Annie Jessie "Nina"  female   6.00      0
     750                               Wells, Miss. Joan  female   4.00      1
     777                      Emanuel, Miss. Virginia Ethel  female   5.00      0
     780                               Ayoub, Miss. Banoura  female  13.00      0
     781           Dick, Mrs. Albert Adrian (Vera Gillespie)  female  17.00      1
     813                  Andersson, Miss. Ebba Iris Alfrida  female   6.00      4
     830             Yasbeck, Mrs. Antoni (Selini Alexander)  female  15.00      1
     852                             Boulos, Miss. Nourelain  female   9.00      1
     853                           Lines, Miss. Mary Conover  female  16.00      0
     875                    Najib, Miss. Adele Kiamie "Jane"  female  15.00      0

          Parch         Ticket      Fare    Cabin Embarked
9         0         237736   30.0708      NaN        C
10        1        PP 9549   16.7000       G6        S
14        0         350406    7.8542      NaN        S
22        0         330923    8.0292      NaN        Q
24        1         349909   21.0750      NaN        S
39        0           2651   11.2417      NaN        C
43        2  SC/Paris 2123   41.5792      NaN        C
58        2     C.A. 34651   27.7500      NaN        S
68        2        3101281    7.9250      NaN        S
71        2        CA 2144   46.9000      NaN        S
84        0     SO/C 14885   10.5000      NaN        S
111       0           2665   14.4542      NaN        C
114       0           2627   14.4583      NaN        C
```



	147	2	W./C. 6608	34.3750	NaN	S
	156	0	35851	7.7333	NaN	Q
...	172	1	347742	11.1333	NaN	S
	184	2	315153	22.0250	NaN	S
	205	1	347054	10.4625	G6	S
	208	0	367231	7.7500	NaN	Q
	233	2	347077	31.3875	NaN	S
	237	2	C.A. 31921	26.2500	NaN	S
	297	2	113781	151.5500	C22 C26	S
	307	0	PC 17758	108.9000	C65	C
	329	1	111361	57.9792	B18	C
	374	1	349909	21.0750	NaN	S
	381	2	2653	15.7417	NaN	C
	389	0	SC 1748	12.0000	NaN	C
	419	2	345773	24.1500	NaN	S
	435	2	113760	120.0000	B96 B98	S
	446	1	250644	19.5000	NaN	S
	448	1	2666	19.2583	NaN	C
	469	1	2666	19.2583	NaN	C
	479	1	3101298	12.2875	NaN	S
	504	0	110152	86.5000	B79	S
	530	1	26360	26.0000	NaN	S
	535	2	F.C.C. 13529	26.2500	NaN	S
	541	2	347082	31.2750	NaN	S
	542	2	347082	31.2750	NaN	S
	618	1	230136	39.0000	F4	S
	634	2	347088	27.9000	NaN	S
	642	2	347088	27.9000	NaN	S
	644	1	2666	19.2583	NaN	C
	689	1	24160	211.3375	B5	S
	691	1	349256	13.4167	NaN	C
	720	1	248727	33.0000	NaN	S
	750	1	29103	23.0000	NaN	S
	777	0	364516	12.4750	NaN	S
	780	0	2687	7.2292	NaN	C
	781	0	17474	57.0000	B20	S
	813	2	347082	31.2750	NaN	S
	830	0	2659	14.4542	NaN	C
	852	1	2678	15.2458	NaN	C
	853	1	PC 17592	39.4000	D28	S
	875	0	2667	7.2250	NaN	C

Subsetting Rows by Categorical variables:

1. Filter the Titanic dataset for passengers whose Embarked port is either "C" (Cherbourg) or "S" (Southampton), assigning the result to embarked c or s. View the printed result.

Subsetting Rows by Categorical variables:

1. Filter the Titanic dataset for passengers whose Embarked port is either "C" (Cherbourg) or "S" (Southampton), assigning the result to embarked c or s. View the printed result.

```
[7] df = pd.read_csv('/content/drive/MyDrive/Concept and Technology Of AI/Week2/Titanic-Dataset.csv')
    embarked_port_C_or_S = df[(df['Embarked']=='C') | (df['Embarked'] == 'S')]
    print("Passengers embarked at Cherbourg or Southampton: ")
    print(embarked_port_C_or_S.head())
```

```
Passengers embarked at Cherbourg or Southampton:
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S

Passengers in first or second class:
    PassengerId  Survived  Pclass  \
1             2         1       1
3             4         1       1
6             7         0       1
9            10         1       2
11           12         1       1

                                                Name     Sex   Age  SibSp  \
1   Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
6                            McCarthy, Mr. Timothy J    male  54.0      0
9                 Nasser, Mrs. Nicholas (Adele Achem)  female  14.0      1
11                          Bonnell, Miss. Elizabeth  female  58.0      0

    Parch    Ticket     Fare Cabin Embarked
1       0  PC 17599  71.2833   C85        C
3       0    113803  53.1000  C123        S
6       0     17463  51.8625   E46        S
9       0    237736  30.0708   NaN        C
11      0    113783  26.5500  C103        S
```

2. Filter the Titanic dataset for passengers whose Pclass is in the list [1, 2] (indicating first or second class), assigning the result to first second class.View the printed result.

2. Filter the Titanic dataset for passengers whose Pclass is in the list [1, 2] (indicating first or second class), assigning the result to first second class.View the printed result.

[18] ✓ 0s

```
first_or_second_class=df[df['Pclass'].isin([1,2])]
print("\nPassengers in first or second class:")
print(first_or_second_class.head())
```

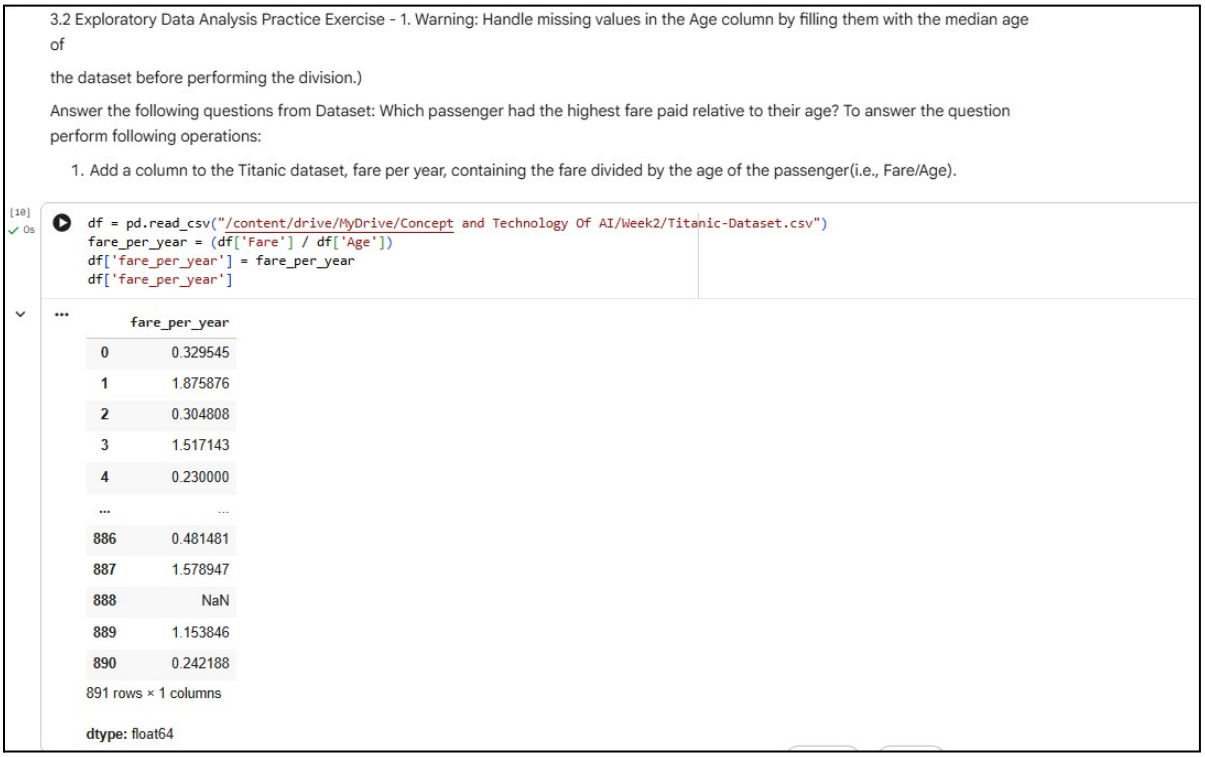
```
Passengers in first or second class:
    PassengerId  Survived  Pclass  \
1             2         1       1
3             4         1       1
6             7         0       1
9            10         1       2
11           12         1       1

                                                 Name     Sex   Age  SibSp  \
1   Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
3        Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
6                             McCarthy, Mr. Timothy J    male  54.0      0
9                 Nasser, Mrs. Nicholas (Adele Achem)  female  14.0      1
11                            Bonnell, Miss. Elizabeth  female  58.0      0

    Parch    Ticket     Fare Cabin Embarked
1       0  PC 17599  71.2833   C85        C
3       0    113803  53.1000  C123        S
6       0     17463  51.8625   E46        S
9       0    237736  30.0708   NaN        C
11      0    113783  26.5500  C103        S
```

3.2 Exploratory Data Analysis Practice Exercise - 1.

Which passenger had the highest fare paid relative to their age?

To answer the question perform following operations:

1. Add a column to the Titanic dataset, fare per year, containing the fare divided by the age of the passenger(i.e., Fare/Age).



3.2 Exploratory Data Analysis Practice Exercise - 1. Warning: Handle missing values in the Age column by filling them with the median age of

the dataset before performing the division.)

Answer the following questions from Dataset: Which passenger had the highest fare paid relative to their age? To answer the question perform following operations:

1. Add a column to the Titanic dataset, fare per year, containing the fare divided by the age of the passenger(i.e., Fare/Age).

[10] ✓ 0s

```
df = pd.read_csv("/content/drive/MyDrive/Concept and Technology Of AI/Week2/Titanic-Dataset.csv")
fare_per_year = (df['Fare'] / df['Age'])
df['fare_per_year'] = fare_per_year
df['fare_per_year']
```

	fare_per_year
0	0.329545
1	1.875876
2	0.304808
3	1.517143
4	0.230000
...	...
886	0.481481
887	1.578947
888	NaN
889	1.153846
890	0.242188

891 rows × 1 columns

dtype: float64

2.

Subset rows where fare per year is higher than 5, assigning this to high fare age.

2. Subset rows where fare per year is higher than 5, assigning this to high fare age.

[11] ✓ 0s

```
high_fare_age = df[df['fare_per_year'] > 5]
print("Fares higher than 5 : ", high_fare_age)
```

```
Fares higher than 5 :       PassengerId  Survived  Pclass                                     Name  \
7               8         0       3            Palsson, Master. Gosta Leonard
16             17         0       3                    Rice, Master. Eugene
27             28         0       1             Fortune, Mr. Charles Alexander
43             44         1       2  Laroche, Miss. Simonne Marie Anne Andree
50             51         0       3                Panula, Master. Juha Niilo
..            ...       ...     ...                                       ...
813           814         0       3        Andersson, Miss. Ebba Iris Alfrida
824           825         0       3              Panula, Master. Urho Abraham
827           828         1       2                     Mallet, Master. Andre
831           832         1       2           Richards, Master. George Sibley
850           851         0       3   Andersson, Master. Sigvard Harald Elias

        Sex    Age  SibSp  Parch           Ticket      Fare        Cabin  \
7      male   2.00      3      1           349909   21.0750          NaN
16     male   2.00      4      1           382652   29.1250          NaN
27     male  19.00      3      2            19950  263.0000  C23 C25 C27
43   female   3.00      1      2    SC/Paris 2123   41.5792          NaN
50     male   7.00      4      1          3101295   39.6875          NaN
..      ...    ...    ...    ...              ...       ...          ...
813  female   6.00      4      2           347082   31.2750          NaN
824    male   2.00      4      1          3101295   39.6875          NaN
827    male   1.00      0      2  S.C./PARIS 2079   37.0042          NaN
831    male   0.83      1      1            29106   18.7500          NaN
850    male   4.00      4      2           347082   31.2750          NaN

    Embarked  fare_per_year
7          S      10.537500
16         Q      14.562500
27         S      13.842105
43         C      13.859733
50         S       5.669643
..       ...            ...
813        S       5.212500
824        S      19.843750
827        C      37.004200
831        S      22.590361
850        S       7.818750

[68 rows x 13 columns]
```

3.

Sort high fare age by descending fare per year, assigning this to high fare age srt.

3. Sort high fare age by descending fare per year, assigning this to high fare age srt.

[12] ✓ 0s

```
high_fare_age_srt = df.sort_values(by='fare_per_year', ascending=False)
print(high_fare_age_srt)
```

```
     PassengerId  Survived  Pclass                                     Name  \
305          306         1       1           Allison, Master. Hudson Trevor
297          298         0       1             Allison, Miss. Helen Loraine
386          387         0       3          Goodwin, Master. Sidney Leonard
164          165         0       3             Panula, Master. Eino Viljami
183          184         1       2                Becker, Master. Richard F
..           ...       ...     ...                                      ...
859          860         0       3                          Razi, Mr. Raihed
863          864         0       3        Sage, Miss. Dorothy Edith "Dolly"
868          869         0       3             van Melkebeke, Mr. Philemon
878          879         0       3                         Laleff, Mr. Kristo
888          889         0       3  Johnston, Miss. Catherine Helen "Carrie"

        Sex   Age  SibSp  Parch      Ticket      Fare    Cabin Embarked  \
305    male  0.92      1      2      113781  151.5500  C22 C26        S
297  female  2.00      1      2      113781  151.5500  C22 C26        S
386    male  1.00      5      2    CA 2144   46.9000      NaN        S
164    male  1.00      4      1     3101295   39.6875      NaN        S
183    male  1.00      2      1      230136   39.0000       F4        S
..      ...   ...    ...    ...         ...       ...      ...      ...
859    male   NaN      0      0        2629    7.2292      NaN        C
863  female   NaN      8      2    CA. 2343   69.5500      NaN        S
868    male   NaN      0      0      345777    9.5000      NaN        S
878    male   NaN      0      0      349217    7.8958      NaN        S
888  female   NaN      1      2  W./C. 6607   23.4500      NaN        S

     fare_per_year
305     164.728261
297      75.775000
386      46.900000
164      39.687500
183      39.000000
..             ...
859            NaN
863            NaN
868            NaN
878            NaN
888            NaN

[891 rows x 13 columns]
```

4.

Select only the Name and fare per year columns of high fare age srt and save the result as result.

5. Look at the result.

4. Select only the Name and fare per year columns of high fare age srt and save the result as result.
5. Look at the result.

```
[13]
✓ 0s
result = df[['Name', 'fare_per_year']]
print(result)
```

```
                                                Name  fare_per_year
0                            Braund, Mr. Owen Harris       0.329545
1    Cumings, Mrs. John Bradley (Florence Briggs Th...       1.875876
2                               Heikkinen, Miss. Laina       0.304808
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)       1.517143
4                             Allen, Mr. William Henry       0.230000
..                                                 ...            ...
886                              Montvila, Rev. Juozas       0.481481
887                       Graham, Miss. Margaret Edith       1.578947
888           Johnston, Miss. Catherine Helen "Carrie"            NaN
889                              Behr, Mr. Karl Howell       1.153846
890                                Dooley, Mr. Patrick       0.242188

[891 rows x 2 columns]
```

Which adult male passenger (age ≥ 18 and Sex is ’male’) paid the highest fare relative to their class?

To answer the question perform following operations:

1. Add a column to the Titanic dataset, fare per class, containing the fare divided by the passenger class i.e. Fare / Pclass.

Which adult male passenger (age ≥ 18 and Sex is 'male') paid the highest fare relative to their class? To answer the question perform following operations:

1. Add a column to the Titanic dataset, fare per class, containing the fare divided by the passenger class i.e. Fare / Pclass.

```
[19]
✓ 0s
fare_per_class = df['Fare'] / df['Pclass']
df['fare_per_class'] = fare_per_class
df
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	fare_per_class
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	2.416667
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	71.283300
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	2.641667
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	53.100000
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	2.683333
...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S	6.500000
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S	30.000000
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S	7.816667
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C	30.000000
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q	2.583333

891 rows × 13 columns

5.

2. Subset rows where the passenger is male (Sex is ”male”) and an adult (Age is greater than or equal to 18), assigning this to adult males.

2. Subset rows where the passenger is male (Sex is “male”) and an adult (Age is greater than or equal to 18), assigning this to adult males.

[20] ✓ 0s

```
adult_males = df[(df['Age'] > 18) & (df['Sex'] == 'male')]
print(adult_males)
```

```
     PassengerId  Survived  Pclass                          Name   Sex  \
0              1         0       3       Braund, Mr. Owen Harris  male
4              5         0       3      Allen, Mr. William Henry  male
6              7         0       1       McCarthy, Mr. Timothy J  male
12            13         0       3  Saundercock, Mr. William Henry  male
13            14         0       3     Andersson, Mr. Anders Johan  male
..           ...       ...     ...                             ...   ...
883          884         0       2  Banfield, Mr. Frederick James  male
884          885         0       3         Sutehall, Mr. Henry Jr  male
886          887         0       2          Montvila, Rev. Juozas  male
889          890         1       1          Behr, Mr. Karl Howell  male
890          891         0       3            Dooley, Mr. Patrick  male

      Age  SibSp  Parch            Ticket     Fare Cabin Embarked  \
0    22.0      1      0         A/5 21171   7.2500   NaN        S
4    35.0      0      0            373450   8.0500   NaN        S
6    54.0      0      0             17463  51.8625   E46        S
12   20.0      0      0         A/5. 2151   8.0500   NaN        S
13   39.0      1      5            347082  31.2750   NaN        S
..    ...    ...    ...               ...      ...   ...      ...
883  28.0      0      0  C.A./SOTON 34068  10.5000   NaN        S
884  25.0      0      0   SOTON/OQ 392076   7.0500   NaN        S
886  27.0      0      0            211536  13.0000   NaN        S
889  26.0      0      0            111369  30.0000  C148        C
890  32.0      0      0            370376   7.7500   NaN        Q

     fare_per_class
0          2.416667
4          2.683333
6         51.862500
12         2.683333
13        10.425000
..              ...
883        5.250000
884        2.350000
886        6.500000
889       30.000000
890        2.583333

[382 rows x 13 columns]
```

3. Sort adult males by descending fare per class, assigning this to adult males srt.

3. Sort adult males by descending fare per class, assigning this to adult males srt.

```
[21] adult_males_srt = adult_males.sort_values(by='fare_per_class', ascending=False)
     print(adult_males_srt)
```

```
     PassengerId  Survived  Pclass                                Name   Sex  \
737          738         1       1             Lesurer, Mr. Gustave J  male
679          680         1       1  Cardeza, Mr. Thomas Drake Martinez  male
438          439         0       1                  Fortune, Mr. Mark  male
27            28         0       1     Fortune, Mr. Charles Alexander  male
118          119         0       1           Baxter, Mr. Quigg Edmond  male
..           ...       ...     ...                                ...   ...
597          598         0       3               Johnson, Mr. Alfred  male
302          303         0       3    Johnson, Mr. William Cahoone Jr  male
271          272         1       3       Tornquist, Mr. William Henry  male
806          807         0       1             Andrews, Mr. Thomas Jr  male
263          264         0       1              Harrison, Mr. William  male

      Age  SibSp  Parch    Ticket      Fare        Cabin Embarked  \
737  35.0      0      0  PC 17755  512.3292         B101        C
679  36.0      0      1  PC 17755  512.3292  B51 B53 B55        C
438  64.0      1      4     19950  263.0000  C23 C25 C27        S
27   19.0      3      2     19950  263.0000  C23 C25 C27        S
118  24.0      0      1  PC 17558  247.5208      B58 B60        C
..    ...    ...    ...       ...       ...          ...      ...
597  49.0      0      0      LINE    0.0000          NaN        S
302  19.0      0      0      LINE    0.0000          NaN        S
271  25.0      0      0      LINE    0.0000          NaN        S
806  39.0      0      0    112050    0.0000          A36        S
263  40.0      0      0    112059    0.0000          B94        S

     fare_per_class
737        512.3292
679        512.3292
438        263.0000
27         263.0000
118        247.5208
..              ...
597          0.0000
302          0.0000
271          0.0000
806          0.0000
263          0.0000

[382 rows x 13 columns]
```

4. Select only the Name, Age, and fare per class columns of adult males sr and save the result as result.
5. Look at the result.

4. Select only the Name, Age, and fare per class columns of adult males sr and save the result as result.
5. Look at the result.

```
[22] result = adult_males_srt[['Name', 'Age', 'Fare']]
     print("\nAdult Males : ")
     print(result)
```

```
Adult Males :
                                  Name   Age      Fare
737             Lesurer, Mr. Gustave J  35.0  512.3292
679  Cardeza, Mr. Thomas Drake Martinez  36.0  512.3292
438                  Fortune, Mr. Mark  64.0  263.0000
27      Fortune, Mr. Charles Alexander  19.0  263.0000
118           Baxter, Mr. Quigg Edmond  24.0  247.5208
..                                 ...   ...       ...
597               Johnson, Mr. Alfred  49.0    0.0000
302    Johnson, Mr. William Cahoone Jr  19.0    0.0000
271       Tornquist, Mr. William Henry  25.0    0.0000
806             Andrews, Mr. Thomas Jr  39.0    0.0000
263              Harrison, Mr. William  40.0    0.0000

[382 rows x 3 columns]
```

3.3 Exploratory Data Analysis with Group-by Method Practice Exercise:

What percent of the total fare revenue came from each passenger class?

To answer the question perform following operation:

1. Calculate the total Fare paid across all passengers in the Titanic dataset.

3.3 Exploratory Data Analysis with Group-by Method Practice Exercise: Based on the dataset Answer the following question: What percent of the total fare revenue came from each passenger class? To answer the question perform following operation:

1. Calculate the total Fare paid across all passengers in the Titanic dataset.

```
[23]
✓ 0s
df = pd.read_csv('/content/drive/MyDrive/Concept and Technology Of AI/Week2/Titanic-Dataset.csv')
total_fare = df['Fare'].sum()
print("Total Fare paid across all passengers in the Titanic dataset : ", total_fare)

Total Fare paid across all passengers in the Titanic dataset :  28693.9493
```

2. Subset for passengers in first class (Pclass is 1) and calculate their total fare.

2. Subset for passengers in first class (Pclass is 1) and calculate their total fare.

```
[24]
✓ 0s
Pclass1_total_fare = df[df['Pclass'] == 1]['Fare'].sum()
print("Total fare of passengers of Pclass 1 : ")
print(Pclass1_total_fare)

Total fare of passengers of Pclass 1 :
18177.4125
```

3. Do the same for second class (Pclass is 2) and third class (Pclass is 3).

3. Do the same for second class (Pclass is 2) and third class (Pclass is 3).

```
[26]
Pclass2_total_fare = df[df['Pclass'] == 2]['Fare'].sum()
print("Total fare of passengers of Pclass 1 : ")
print(Pclass2_total_fare)

Pclass3_total_fare = df[df['Pclass'] == 3]['Fare'].sum()
print("Total fare of passengers of Pclass 1 : ")
print(Pclass3_total_fare)

Total fare of passengers of Pclass 1 :
3801.8417
Total fare of passengers of Pclass 1 :
6714.6951
```

4. Combine the fare totals from first, second, and third classes into a list.

4. Combine the fare totals from first, second, and third classes into a list.

```
[27]
✓ 0s
fare_class_total = pd.Series([Pclass1_total_fare, Pclass2_total_fare, Pclass3_total_fare])
print(fare_class_total)

0    18177.4125
1     3801.8417
2     6714.6951
dtype: float64
```

5. Divide the totals for each class by the overall total fare to get the proportion of fare revenue by class.

```
[28]
✓ 0s
proportion_fare = lambda x : x/x.sum()
print(proportion_fare(fare_class_total))

0    0.633493
1    0.132496
2    0.234011
dtype: float64
```

Based on the dataset Answer the following question:

What percent of the total number of passengers on the Titanic belonged to each age group

(e.g., child, adult, senior)?

To answer the question perform following operation:

1. Create a new column, age group, that categorizes passengers into "child" (age < 18), "adult" (age 18{64), and "senior" (age 65 and above).

Based on the dataset Answer the following question: What percent of the total number of passengers on the Titanic belonged to each age group (e.g., child, adult, senior)? To answer the question perform following operation:

1. Create a new column, age group, that categorizes passengers into "child" (age < 18), "adult" (age 18{64), and "senior" (age 65 and above).

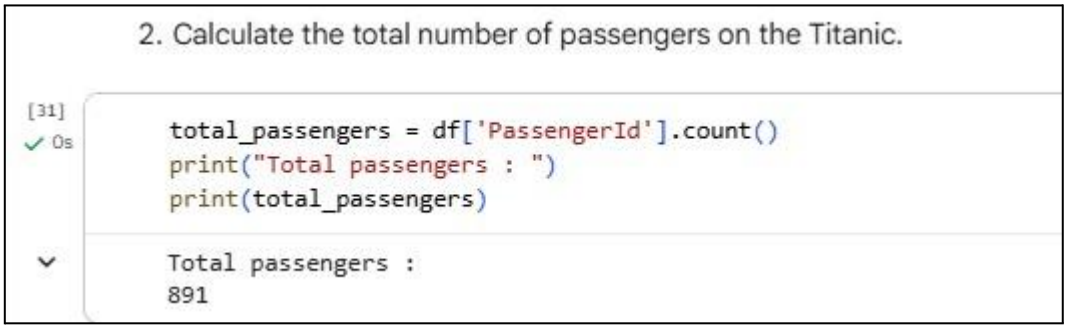
[30] ✓ 0s

```
import numpy as np
age_group = np.where(df['Age'] < 18, 'child', np.where(df['Age'] < 65, 'adult', 'senior'))
df['age_group'] = age_group
df
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	age_group
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	adult
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	adult
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	adult
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	adult
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	adult
...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S	adult
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S	adult
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S	senior
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C	adult
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q	adult

891 rows × 13 columns

2. Calculate the total number of passengers on the Titanic.



3. Count the number of passengers in each age group.

3. Count the number of passengers in each age group.

```
[32]
✓ 0s
num_of_passengers_age_group = df.groupby('age_group').size()
print(num_of_passengers_age_group)

age_group
adult     590
child     113
senior    188
dtype: int64
```

4. Divide the count of each age group by the total number of passengers to get the proportion of passengers in each age group.

```
4. Divide the count of each age group by the total number of passengers to get the proportion of passengers in each age group.

[34]
✓ 0s
proportion_age = lambda x : x/x.sum()
print(proportion_age(num_of_passengers_age_group))

age_group
adult     0.662177
child     0.126824
senior    0.210999
dtype: float64
```

5. Display the proportion as a percentage.

```
5. Display the proportion as a percentage.

[36]
✓ 0s
print(proportion_age(num_of_passengers_age_group) * 100)

age_group
adult     66.217733
child     12.682379
senior    21.099888
dtype: float64
```