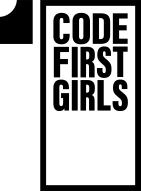
**Homework Week 1**

**Question 1 [20 points]**

In no more than 500 words, explain the differences between a Jupyter Notebook (.ipynb extension) and a Python file (.py extension). Focus on the advantages and disadvantages of one or the other, particularly for data analysts/scientists. You are encouraged to use examples and screenshots of the different scripts to support your arguments.

Jupyter Notebooks are excellent for interactive exploration, data visualization, and documentation, making them a favorite among data analysts and scientists.Each cell in a notebook can be executed separately, allowing for incremental development and real-time output display. It will be beneficial for data analyst/scientist to conduct visualisation and documentation with detailed explanation.

Python files are more suitable for structured programming, modular design, version control, automation, and production-level deployment, as it focuses solely on code execution without the interactive features of notebooks, which is more suitable for automating tasks and running processes in a repeatable manner.

For example, when I want to establish a whole workflow of customer analysis, I will use Jupyter Notebook to load the dataset, draw a descriptive analysis and build machine learning model.

When I want to build a API booking system for a salon, I will use Python file to achieve the integration and automation of functions triggering.

**Question 2 [10 points]**

What is the difference between a Pandas DataFrame and a Pandas series. Show an example of how you create each of them.

Pandas Series is a one-dimensional labeled array

# Creating a Series from a list

data = [10, 20, 30, 40, 50]

series = pd.Series(data)

Pandas DataFrame is a two-dimensional labeled data structure with columns and rows.

# Creating a DataFrame from a dictionary

data = {

'Name': ['Alice', 'Bob', 'Charlie', 'David'],

'Age': [25, 30, 22, 28],

'City': ['New York', 'Los Angeles', 'Chicago', 'Houston']

}

df = pd.DataFrame(data)

**Question 3 [10 points]**

Starting from the argument that a Pandas DataFrame represents rectangular data, use the internet and other resources to describe in no more than a few sentences the difference between rectangular and non-rectangular data.

A Pandas DataFrame represents rectangular data organized in a tabular format with consistent rows and columns. Non-rectangular data, in contrast, lacks this uniform structure and can include nested or varying attributes, making it more complex to analyze and process. Examples of non-rectangular data include hierarchical, nested, or graph-based datasets.

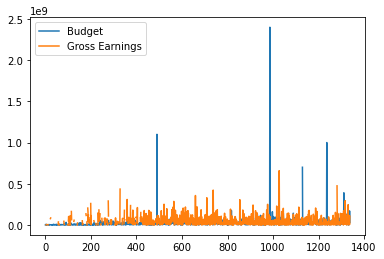
**Question 4 [10 points]**

Starting from the data visualisation usage from Session 2, give examples of when figures could be:

1. Of use to the data scientist to identify patterns in the data/highlight the important parts of a data set;
2. Tell a story and create business presentations.

For example, we used multi-line plot to visualize budget and gross earnings of movies. Thus we can identify a business insight between expense and revenue.

movies\_df[['Budget', 'Gross Earnings']].plot.line()



**Question 5 [50 points]**

**Each sub-question is worth 10 points.**

Using the *titanic dataset* which you can read into your notebook using the following code,

import pandas as pd

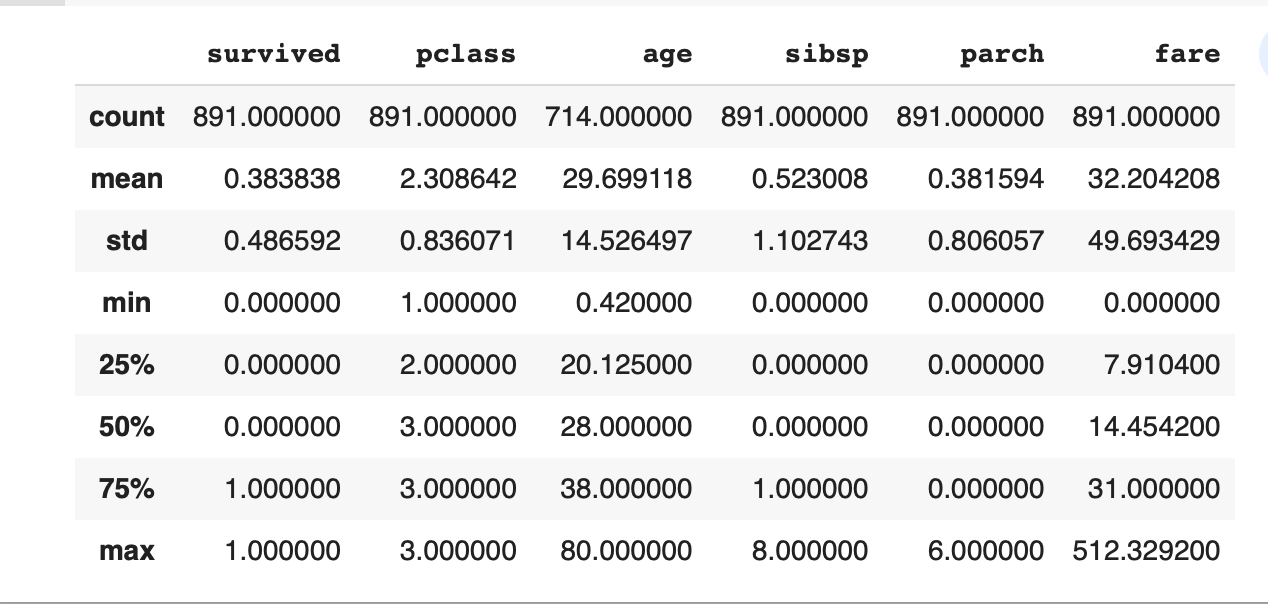
titanic = pd.read\_csv('https://raw.githubusercontent.com/mwaskom/seaborn-data/master/titanic.csv')

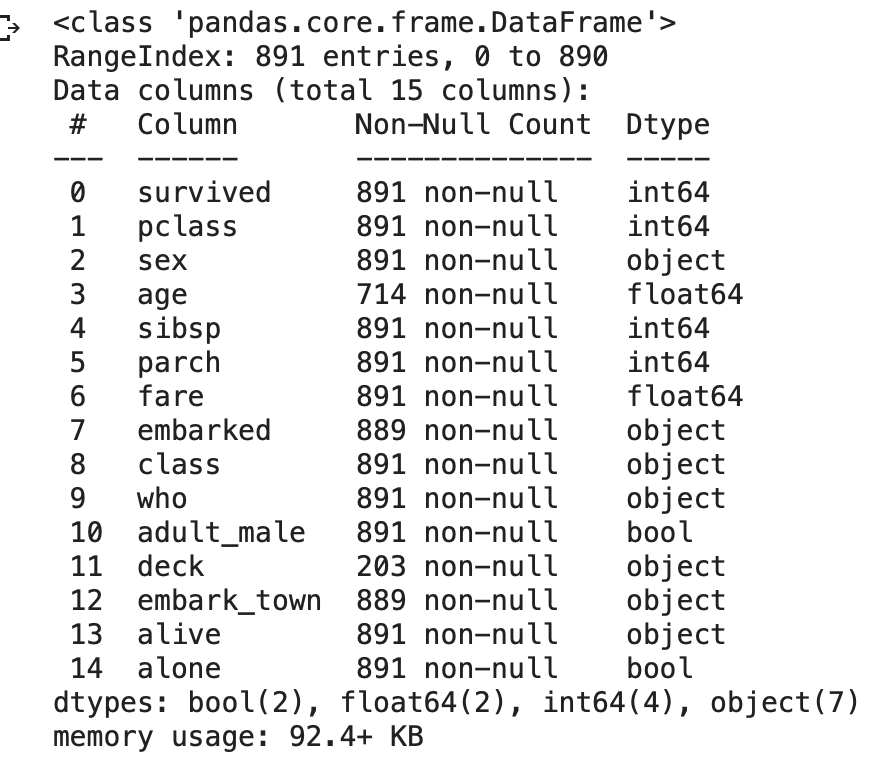
answer the following questions:

1. How many columns and rows does the data have?

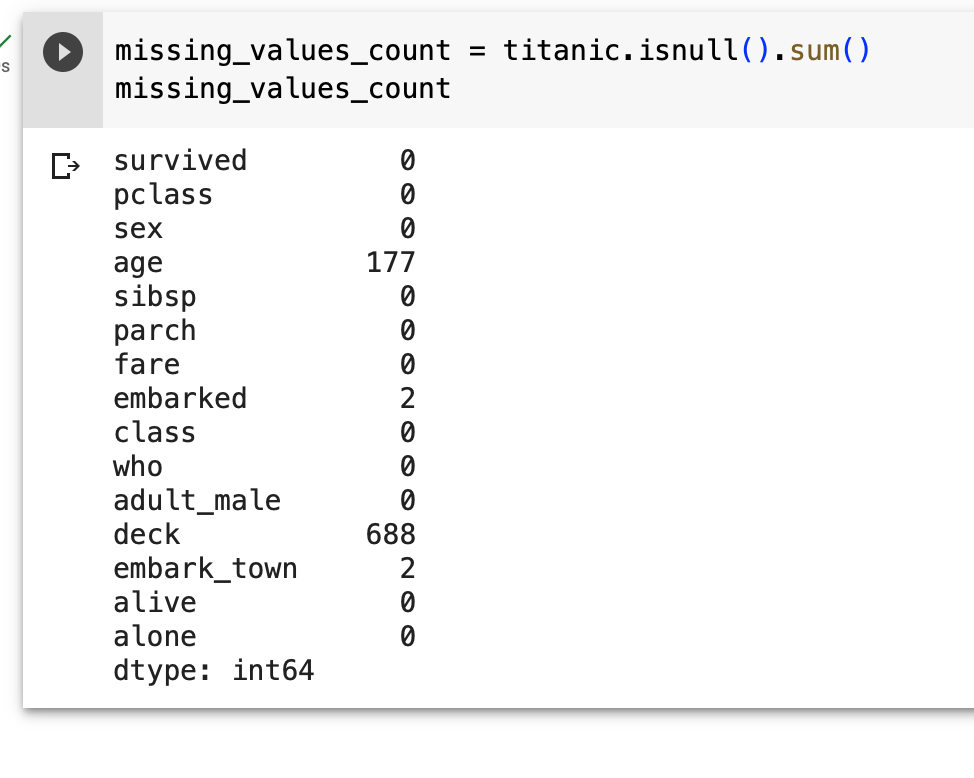
891 rows × 15 columns

1. Get a sense of your data and find the *min, max,* and *count/mean* depending on the data type.





1. Give an overview (code and an explanation) of all missing values in the data.



1. Delete the rows where you do not have information about the *age* of the person. Then group the passengers in a 10 year age range (for example, you can do something like 0 – 10, 11 – 20, 21 – 30, etc).

#delete rows where do not have information about the age

cleaned\_data= titanic.dropna(subset=['age'])

## Define age ranges

age\_bins = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

# Define labels for age ranges

age\_labels = ['0-10', '11-20', '21-30', '31-40', '41-50', '51-60', '61-70', '71-80', '81-90', '91-100']

# Create a new 'Age Range' column by applying the pd.cut function

cleaned\_data['Age Range'] = pd.cut(cleaned\_data['age'], bins=age\_bins, labels=age\_labels, right=False)

# Group passengers by age range

grouped\_data = cleaned\_data.groupby('Age Range').value.counts()

1. For each age category created in d), find out how many passengers are *female/male,* and how many travelled in each *class*.

# passengers by gender and class in each age range

passengers\_by\_age\_gender\_class = grouped\_data[['sex', 'pclass']].value\_counts().unstack(level=1)

passengers\_by\_age\_gender\_class

# Calculate the count of passengers by gender and class in each age range

passengers\_by\_age\_gender\_class = grouped\_data[['sex', 'pclass']].value\_counts().unstack(level=1)

passengers\_by\_age\_gender\_class