TITANIC CSV Data Processing using Pandas, Numpy and Matplotlib Commands

**EXPLORATORY DATA ANALYSIS (EDA):**

As per definition, Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations.

The major purpose of performing EDA is to understand what data are we dealing with. Because if we don’t, then what is the purpose of even proceeding with it?

Getting insights about the dataset in use can be done through visualization as well as tabular formats through pandas and numpy commands.

If our dataset is not perfect, i.e., it has some null values, some empty cells, or some duplicate rows, then using certain commands we can normalize the data, hence ensuring the most accurate analysis possible and therefore, make life easier for us.

So, let’s proceed with EDA of Titanic’s ‘train.csv’.

Before everything, import pandas and numpy for accessing their commands, and matplotlib.pyplot for visualisation when and where required.



Then, we read the csv file into a variable:



Whenever we want to perform any actions on the data of this file, we need to know this variable too.

1. describe():

The describe() method returns description of the data in the DataFrame.

If the DataFrame contains numerical data, the description contains these information for each column:

count - The number of not-empty values.

mean - The average (mean) value.

std - The standard deviation.

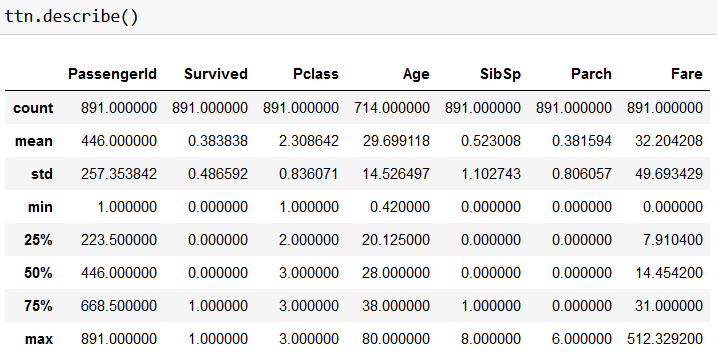
min - the minimum value.

25% - The 25% percentile\*.

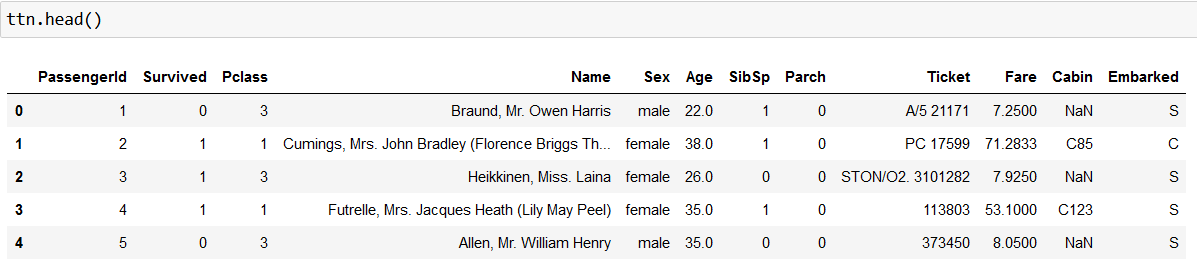
50% - The 50% percentile\*.

75% - The 75% percentile\*.

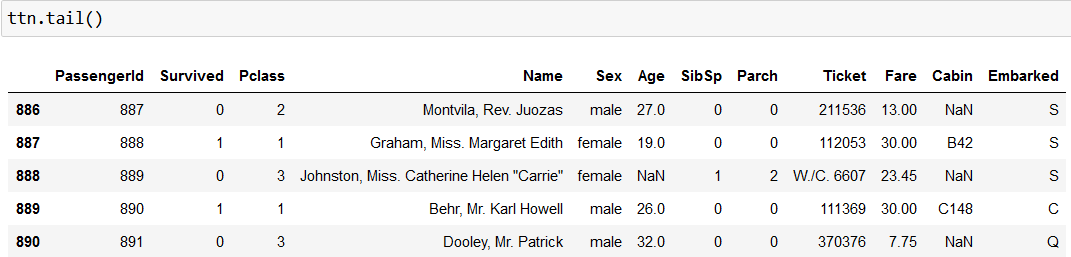
max - the maximum value.



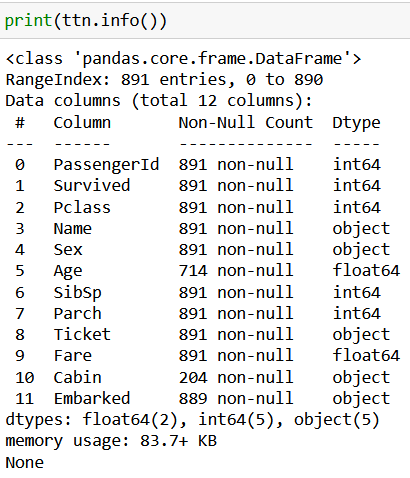
1. head(): This command displays the top n (default 5) rows of data.



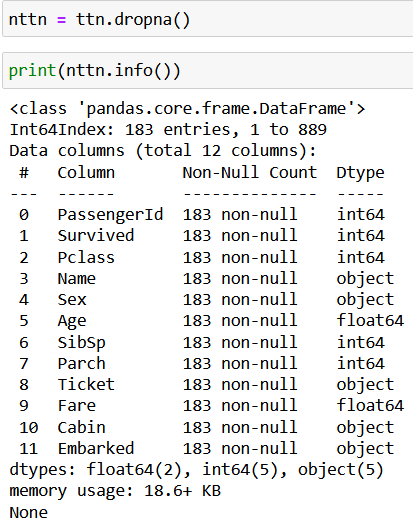
1. tail(): This command displays the bottom n (default 5) rows of data.



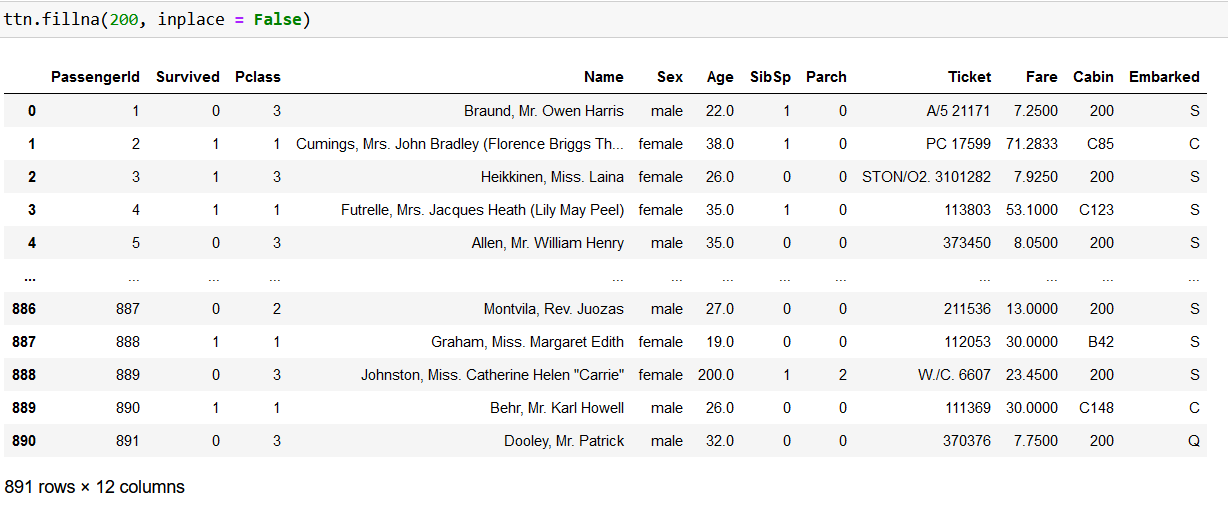
1. info(): This command gives you more information about the dataset. Also tells us number of non-null values.



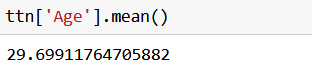
1. dropna(): This command drops values(rows) with empty cells. It doesn’t change the original data frame.

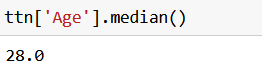


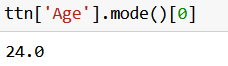
1. fillna(): This command fills desired data into the empty cells. It can also be used to fill the empty cells of a particular column.



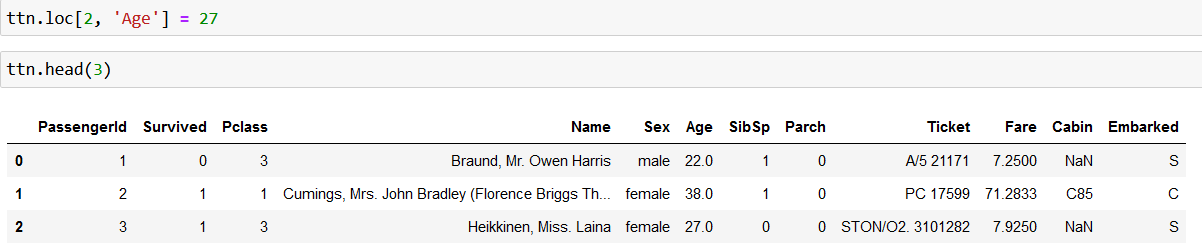
1. mean(), median(), mode():

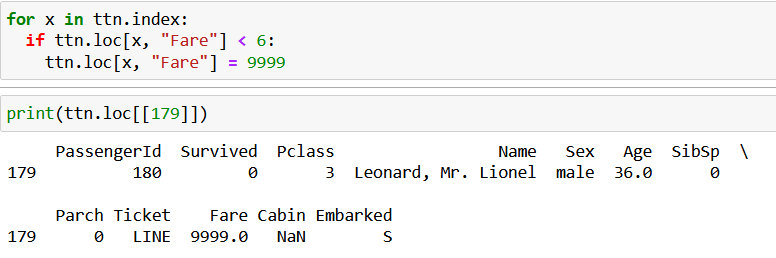




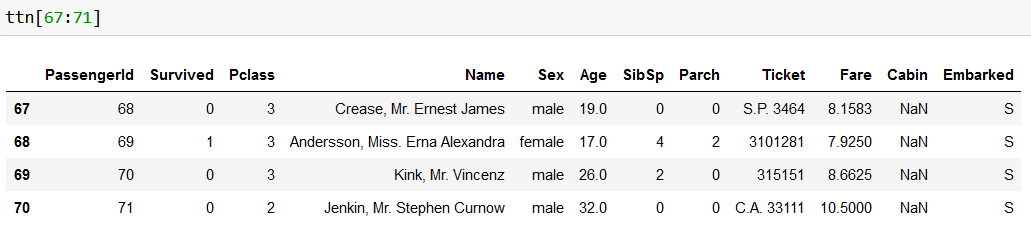


1. loc[]: This command is used to replace particular value in a cell.

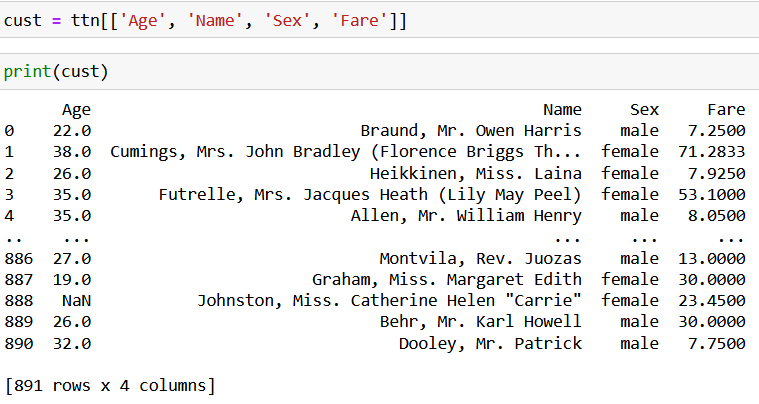




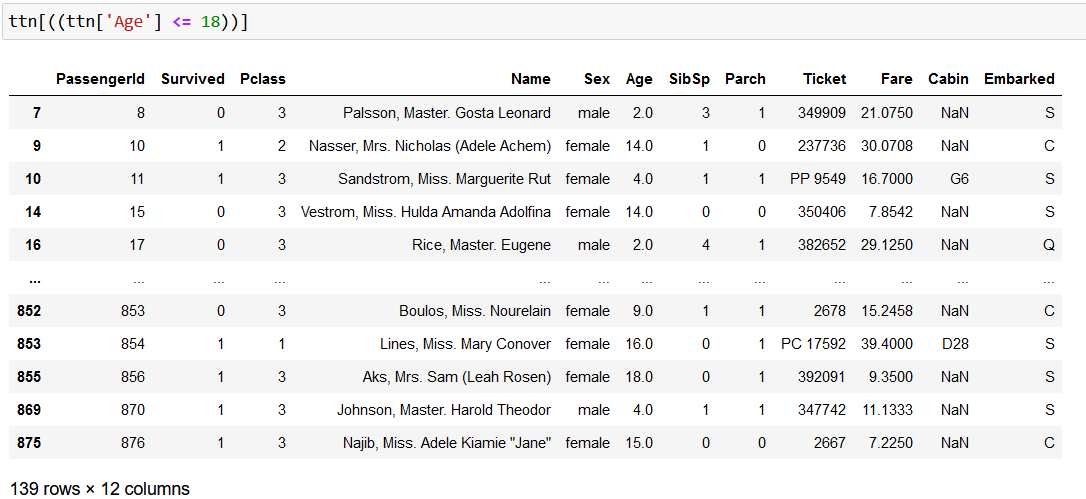
1. Slicing data: We can display only those rows we desire.



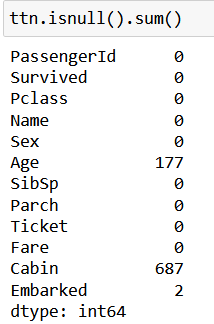
1. Slicing Data: We can display groups of columns as per our desire.



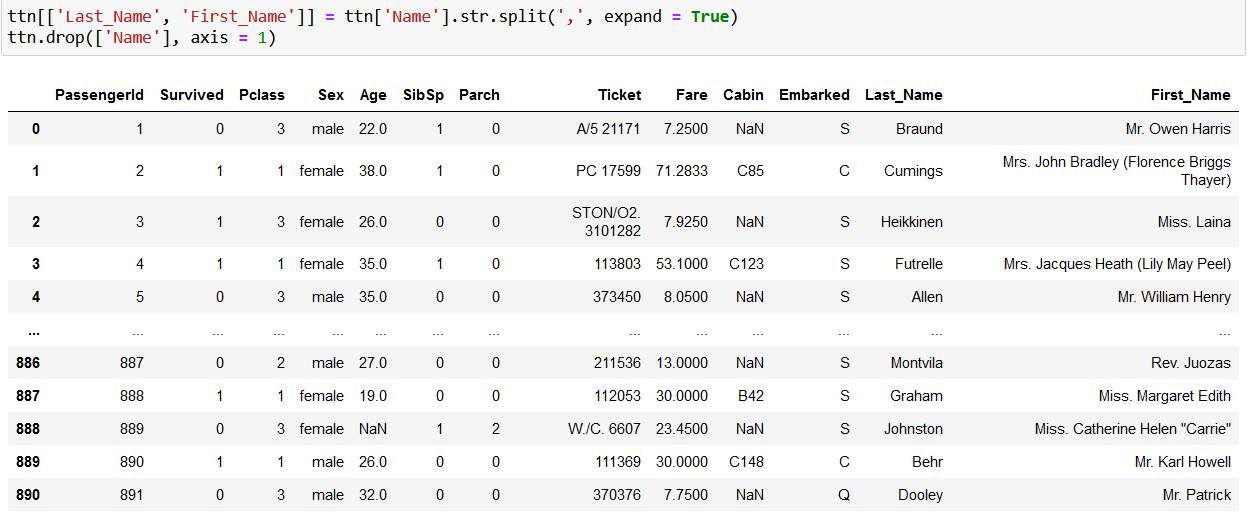
1. Slicing Data: We can filter out custom rows as per requirement.



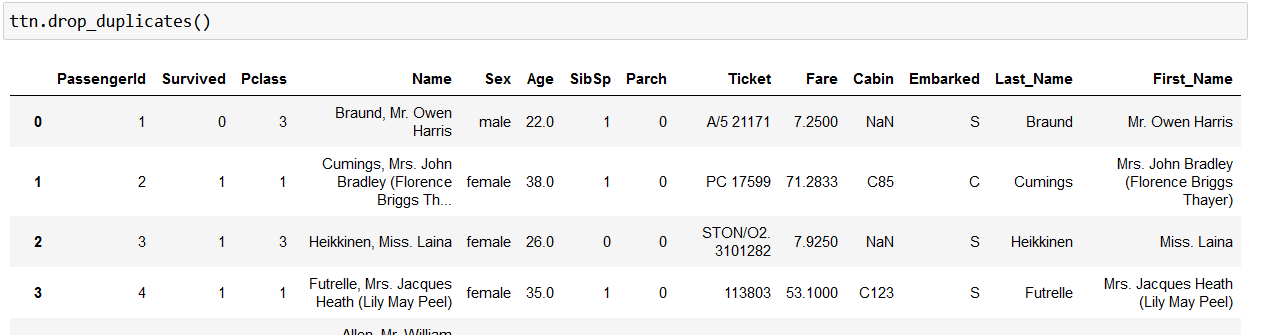
1. isnull(): Check null valued rows



1. Splitting one column into two, and dropping a column.

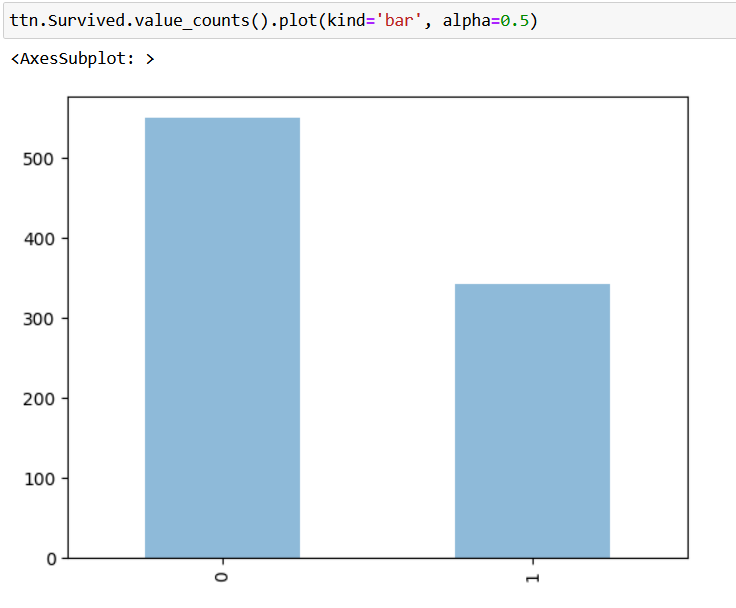


1. drop\_duplicates(): This command drops rows that are exact copies/duplicates of each other.

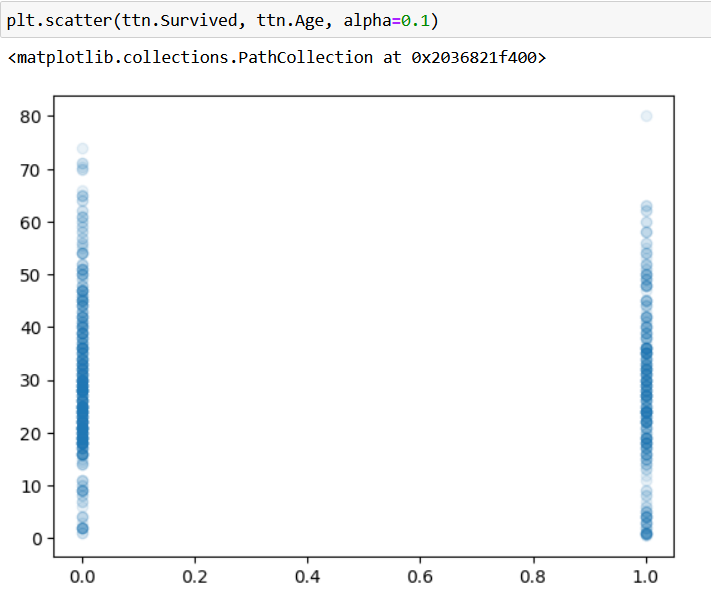


Now let’s make graphs on the given data

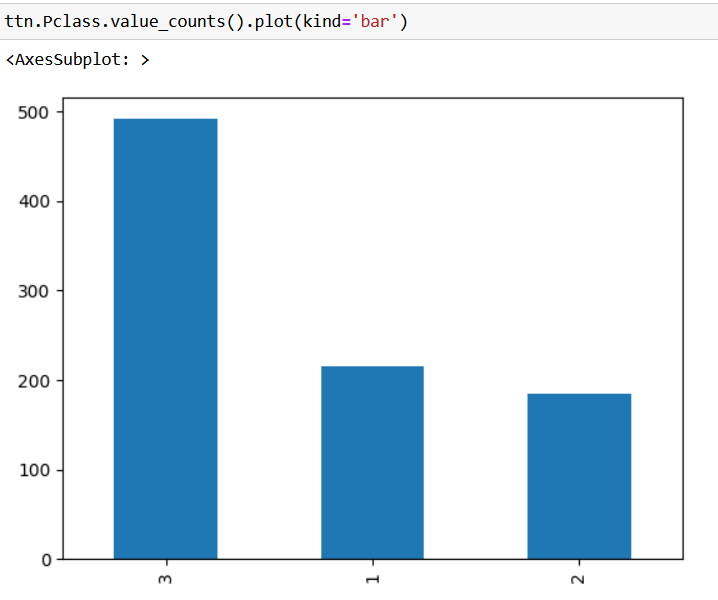
1. Survivors count. Made using value\_counts()



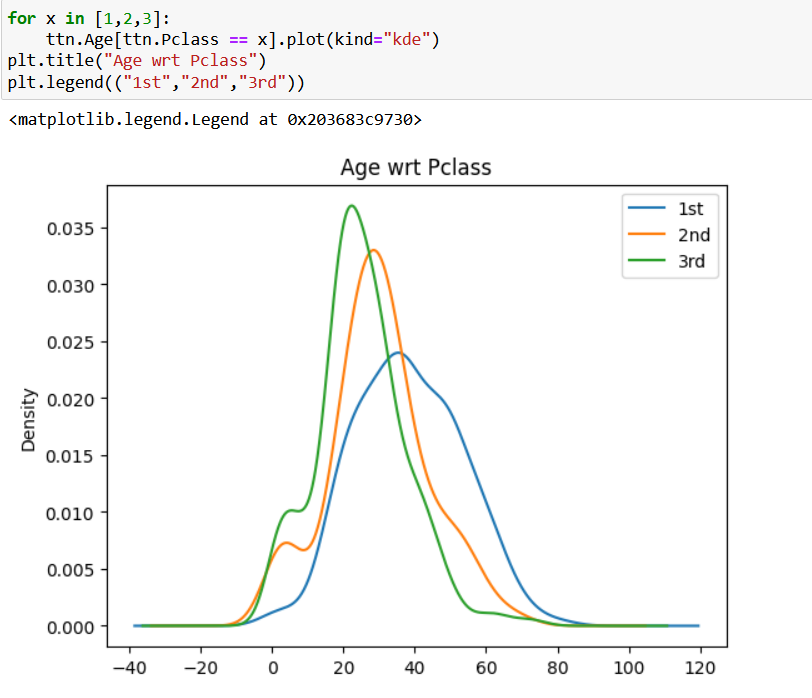
1. Survivor count: Made using matplotlib scatter



1. Passenger class: Number of people in each class



1. Age vs Passenger Class: Made using for loop along with kde graph



1. Gender-wise survival statistics, Made using seaborn.

