Kampus Merdeka INDONESIA JAYA MyEduSolve

Sumary Group Numpy #6

Atika Rahmah
Daril Hana Salsabila
Donny P Tambunan
Muhammad Atthariq
Muhammad Iqbal Rustan

TABLE OF CONTENTS

12 April 2022

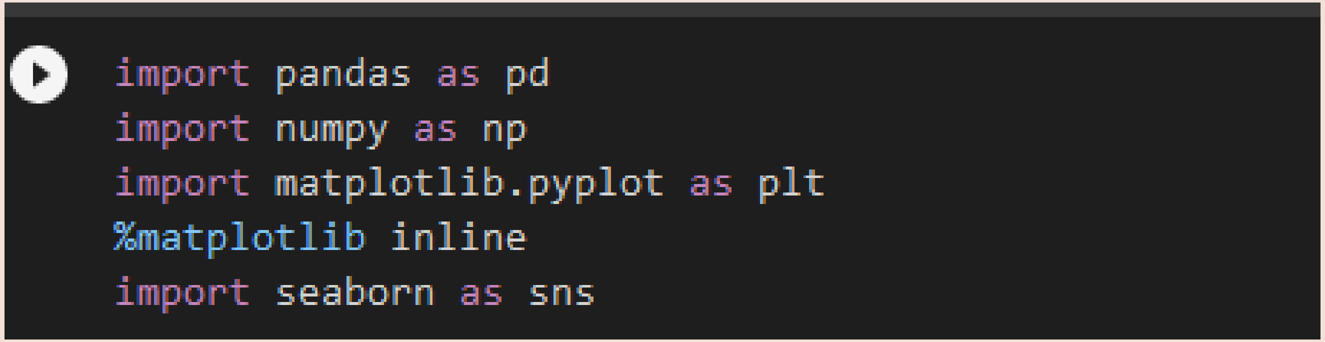
Data Cleansing, Column Age,
Column Cabin, Column Embarked

Data Cleansing

- Data Cleansing is a process that cleans data.
- The data is in the form of tabulus
- The dataset used is Titania data
- Numphy is a numerical calculation
- Data Understanding which is used to examine the data in order to identify problems in the data

Data Cleansing

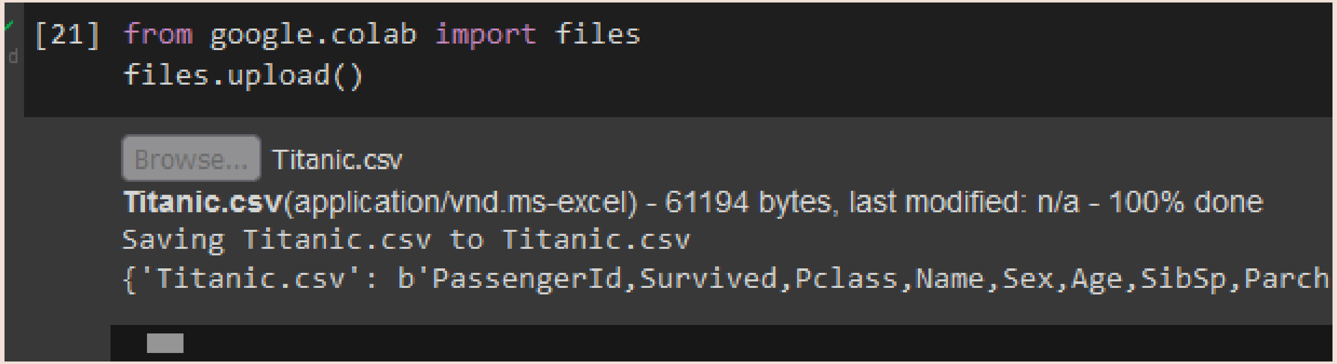
Some required libraries:



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

Data Cleansing

Import the data to be cleaned by entering the file into Google Colab

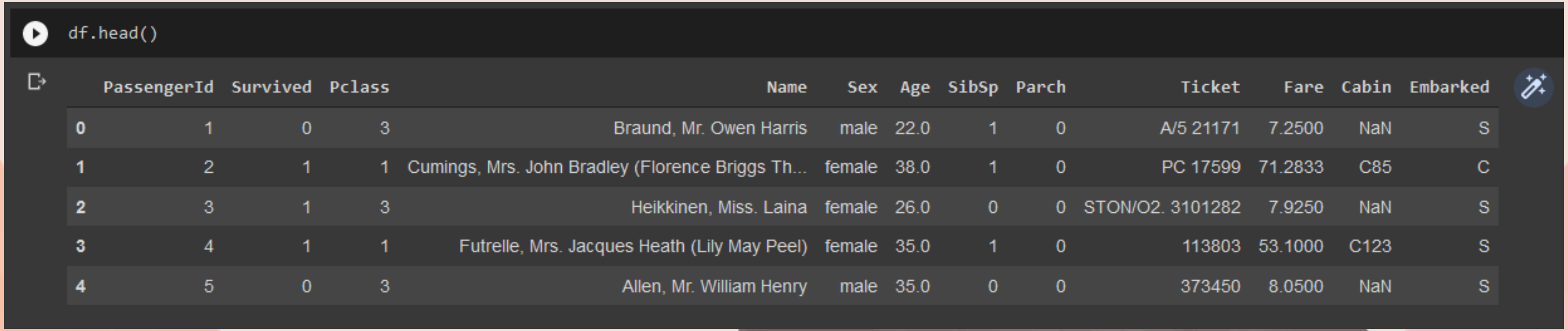


```
[21] from google.colab import files
     files.upload()
```

```
Titanic.csv
Titanic.csv(application/vnd.ms-excel) - 61194 bytes, last modified: n/a - 100% done
Saving Titanic.csv to Titanic.csv
{'Titanic.csv': b'PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch
```

Data Cleansing

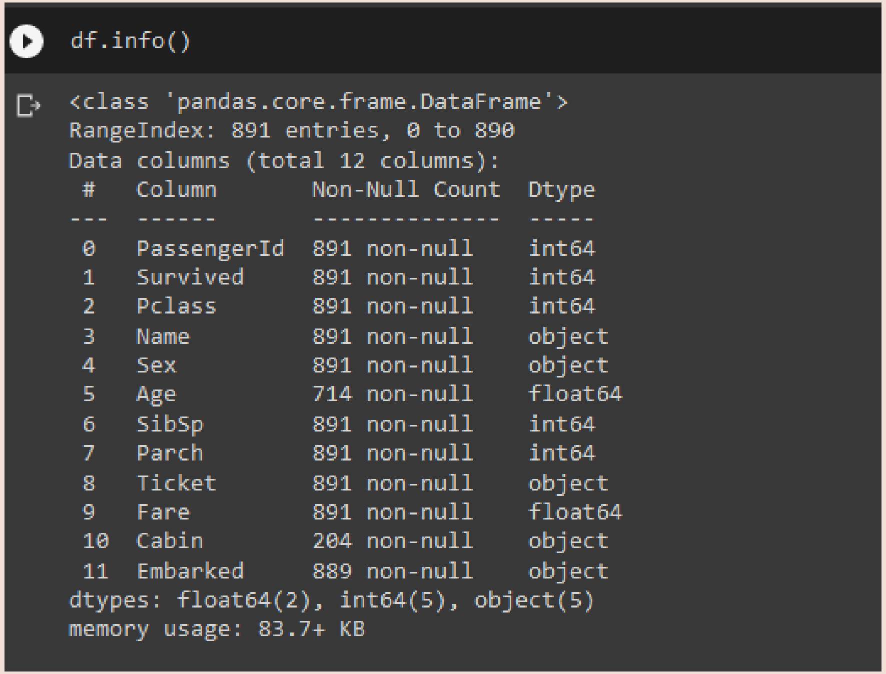
- ❑ df.head to display data that has been imported from the top
- ❑ df.head(10) indicates the top 10 inputted data
- ❑ df.fall to display the imported data from the bottom order



```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

df.info is used to check the condition of data showing columns, missing values, and data types.



```
df.info()
```

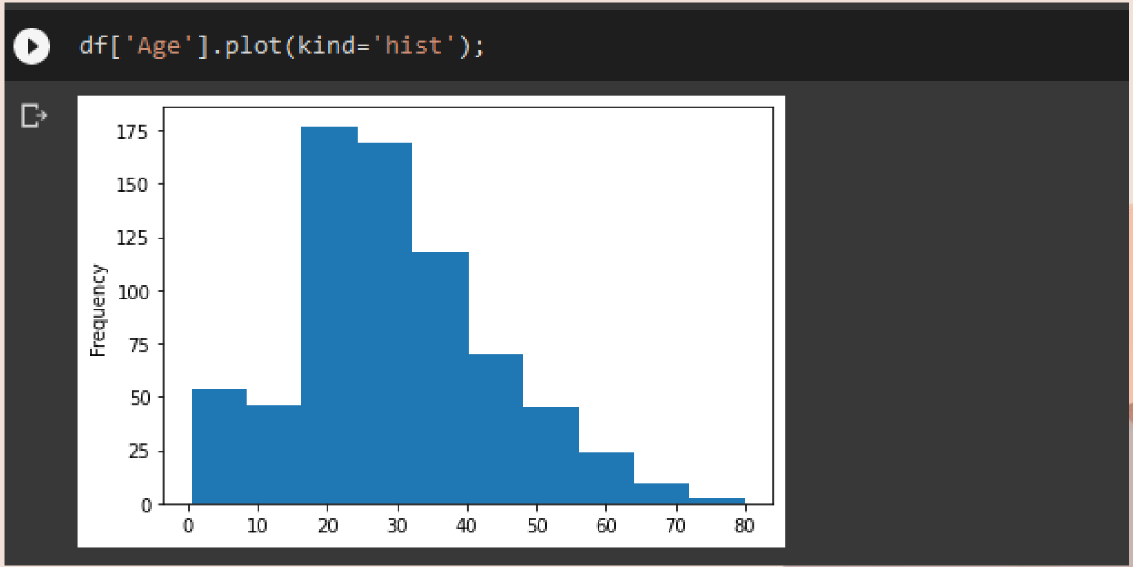
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    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
```

Column Age

Age conditions on imported data

Column Age

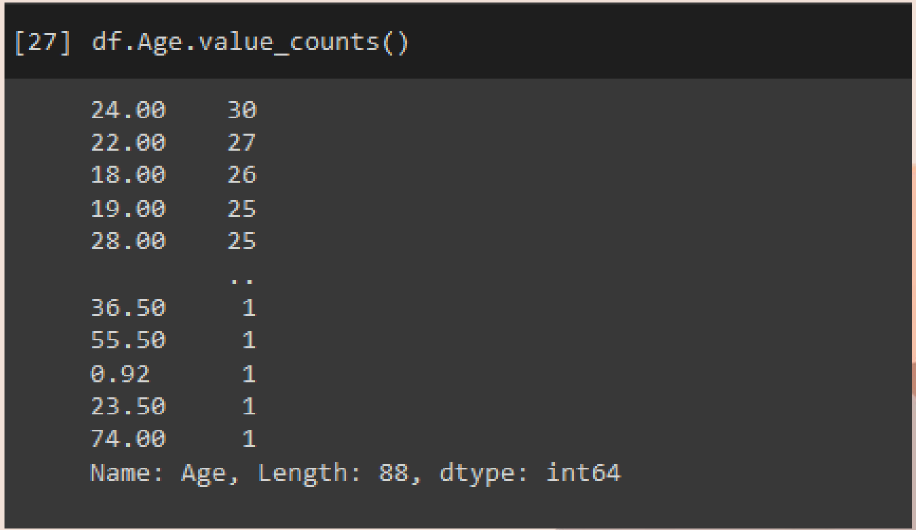
Displaying age data in the form of a histogram



```
df['Age'].plot(kind='hist');
```

Column Age

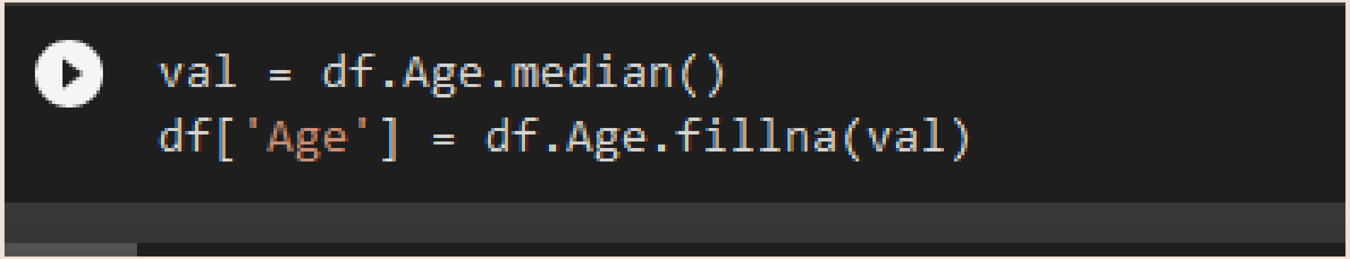
df.age.value(counts) to see the proportional age



```
[27] df.Age.value_counts()

     24.00    30
     22.00    27
     18.00    26
     19.00    25
     28.00    25
              ..
     36.50     1
     55.50     1
     0.92      1
     23.50     1
     74.00     1
     Name: Age, Length: 88, dtype: int64
```

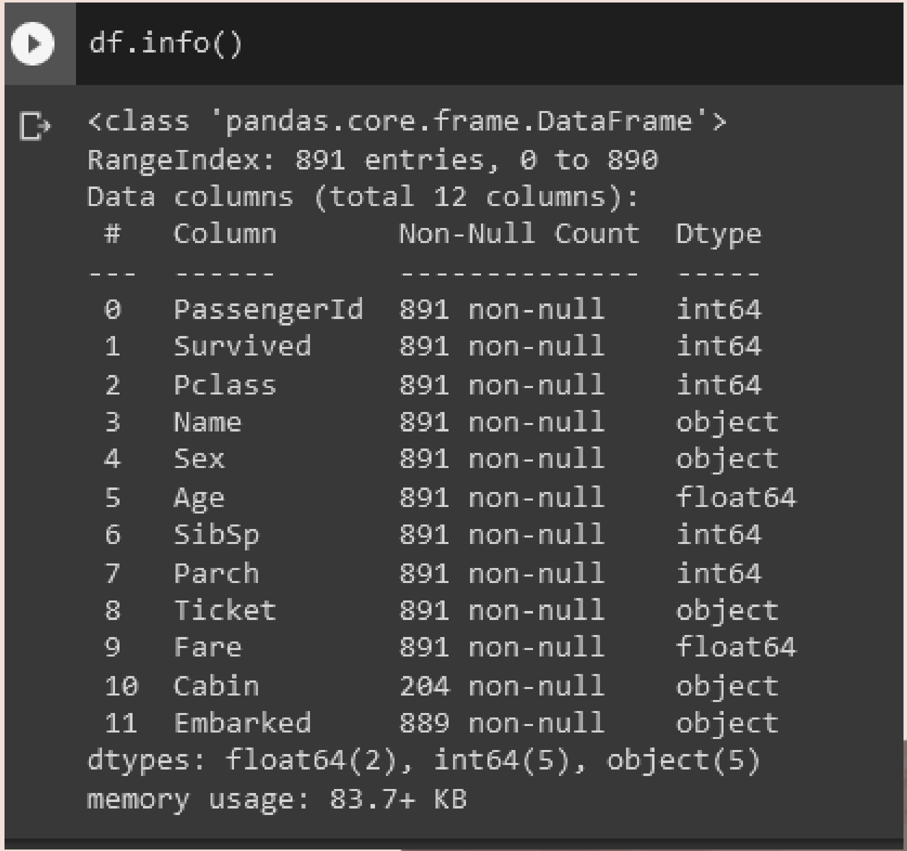
Column Age

- Column age has a skewness distribution so use the median
- If the column age distribution is normal, then use the mean



```
val = df.Age.median()
df['Age'] = df.Age.fillna(val)
```

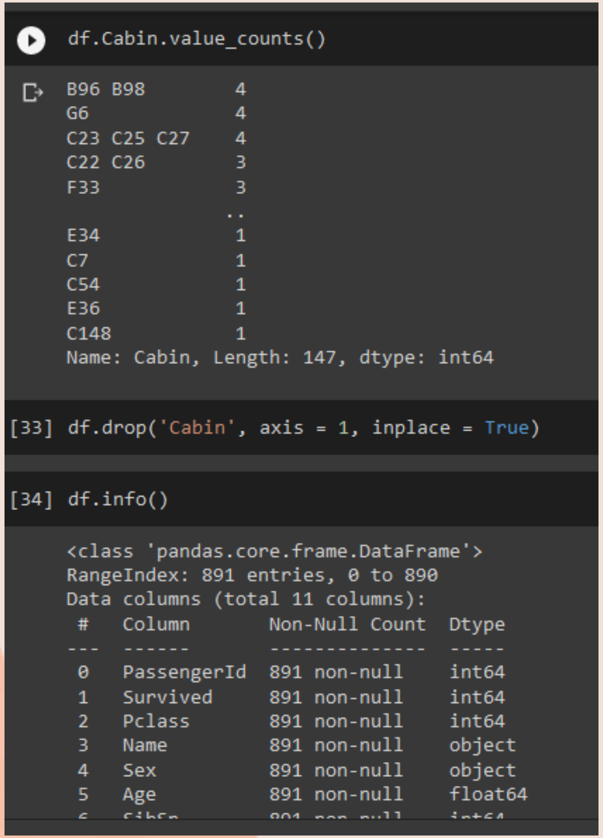
Column Age



```
df.info()
```

```
<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          891 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
```

Column Cabin



```
df.Cabin.value_counts()
```

```
B96 B98        4
G6             4
C23 C25 C27    4
C22 C26        3
F33            3
              ..
E34            1
C7             1
C54            1
E36            1
C148           1
Name: Cabin, Length: 147, dtype: int64
```

```
[33] df.drop('Cabin', axis = 1, inplace = True)
```

```
[34] df.info()
```

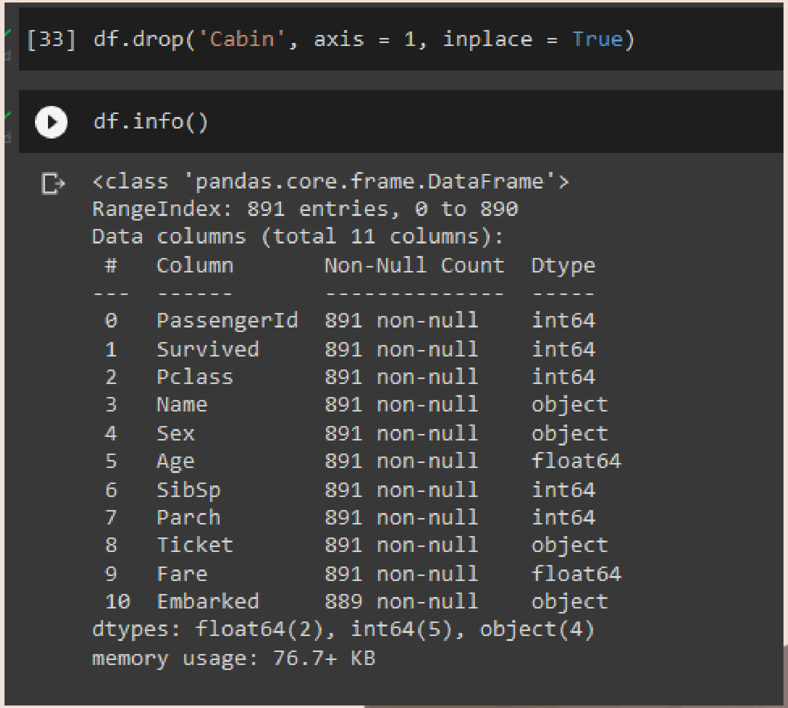
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    float64
```

There is null data in Cabin because the total number of data entries is 891 and Cabin is 204

The data cabin has many unique values so it's deleted

Column Cabin

- :Delete cabin data and display as follows:

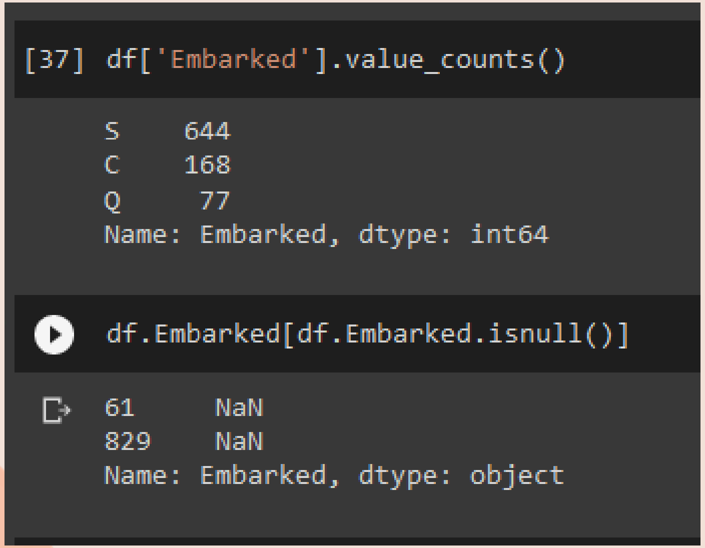


```
[33] df.drop('Cabin', axis = 1, inplace = True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 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          891 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  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(4)
memory usage: 76.7+ KB
```

Column Embarked

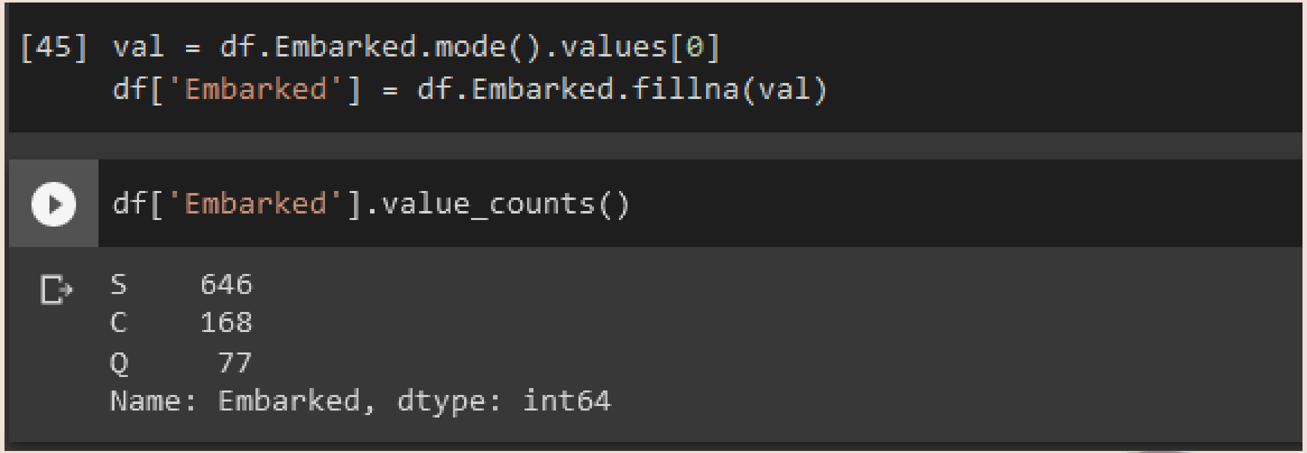


```
[37] df['Embarked'].value_counts()

     S    644
     C    168
     Q     77
     Name: Embarked, dtype: int64

     df.Embarked[df.Embarked.isnull()]

     61     NaN
     829    NaN
     Name: Embarked, dtype: object
```

- There is null data on Embarked because the total number of data entries is 891 and Embarked is 889
- Checking null data location
- Data column embarked in the form of categoric data

Column Embarked

- Perform imputation using the mode because the data is categoric. The mode of the Embarked column proportions is S because it occurs frequently.



```
[45] val = df.Embarked.mode().values[0]
     df['Embarked'] = df.Embarked.fillna(val)
```

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

```
S    646
C    168
Q     77
Name: Embarked, dtype: int64
```

13 April 2022

Data Cleansing

Column SibSp and Column Parch

Column SibSp

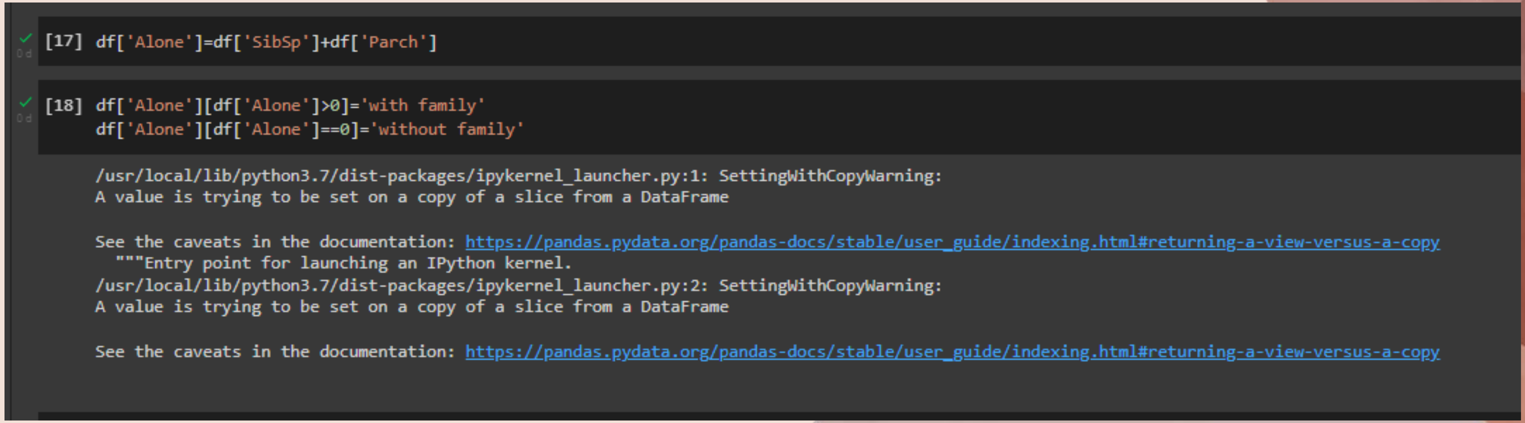
Column SibSp (Sibling Spouse) is a column that states the number of siblings or the number of partners carried by the passenger

Column Parch

Column Parch (Parent Children(means column which states the number of parents or the number of children carried by Passenger

Column SibSp and Column Parch

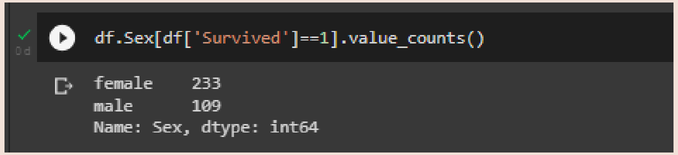
We will perform data manipulation. Manipulating here is not to change the value of the data but to make it easier for this data to be read by the machine. We will create a new column that displays whether they are alone or with family.



```
[17] df['Alone']=df['SibSp']+df['Parch']
```

```
[18] df['Alone'][df['Alone']>0]='with family'
     df['Alone'][df['Alone']==0]='without family'

     /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
     A value is trying to be set on a copy of a slice from a DataFrame

     See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
       """Entry point for launching an IPython kernel.
     /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
     A value is trying to be set on a copy of a slice from a DataFrame

     See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

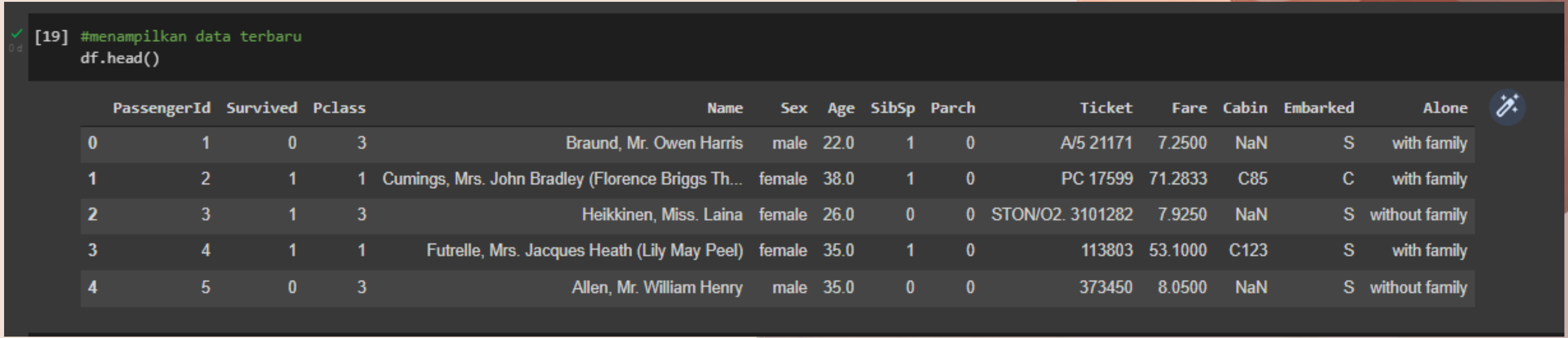
Column Survived and Column Sex



```
df.Sex[df['Survived']==1].value_counts()

female    233
male      109
Name: Sex, dtype: int64
```

Relation Between Column Survived and Column Sex

After created a new column. Now, we will display the newest data.



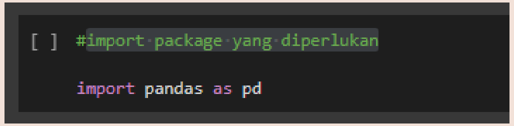
```
[19] #menampilkan data terbaru
     df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Alone
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	with family
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	with family
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	without family
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	with family
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	without family

13 April 2022

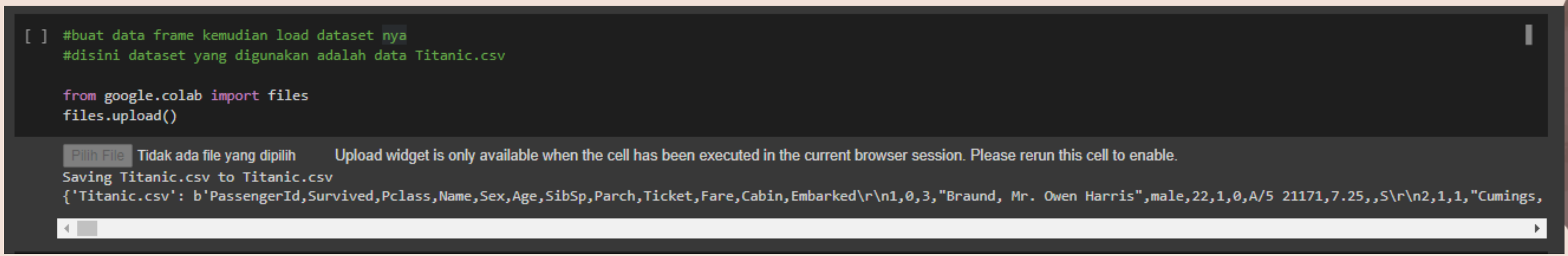
Explanatory Data Analysis

import the required packages



```
[ ] #import package yang diperlukan

    import pandas as pd
```

- ❑ Create a data frame then load the dataset
- ❑ Here the dataset used is data Titanic.csv

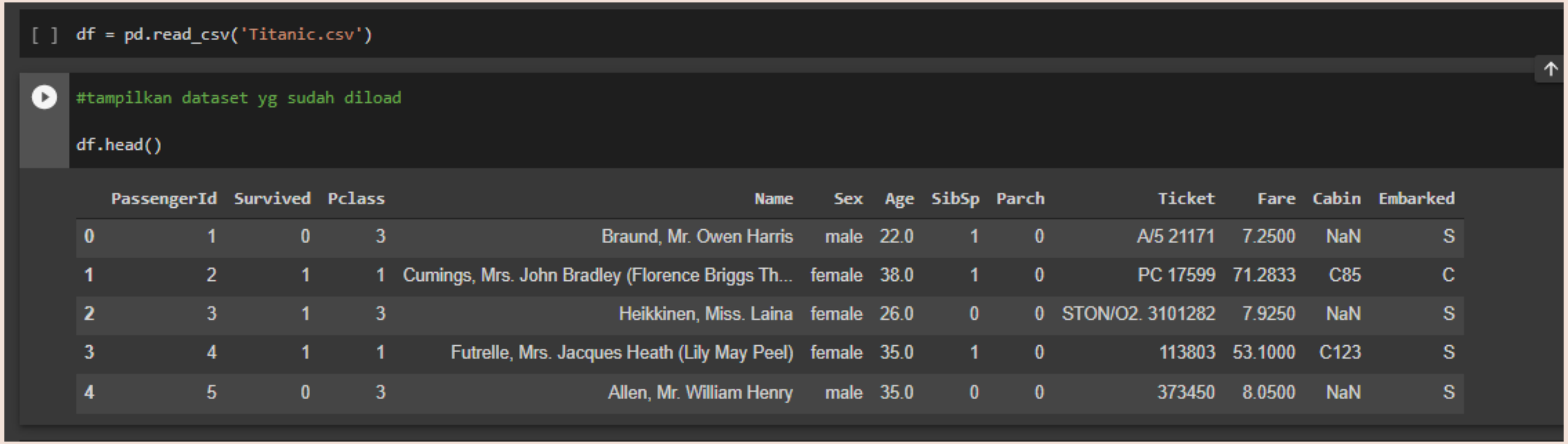


```
[ ] #buat data frame kemudian load dataset nya
    #disini dataset yang digunakan adalah data Titanic.csv

    from google.colab import files
    files.upload()
```

Pilih File Tidak ada file yang dipilih   Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Titanic.csv to Titanic.csv
{'Titanic.csv': b'PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked\r\n1,0,3,"Braund, Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,S\r\n2,1,1,"Cumings,
```

Displays the database that has been loaded

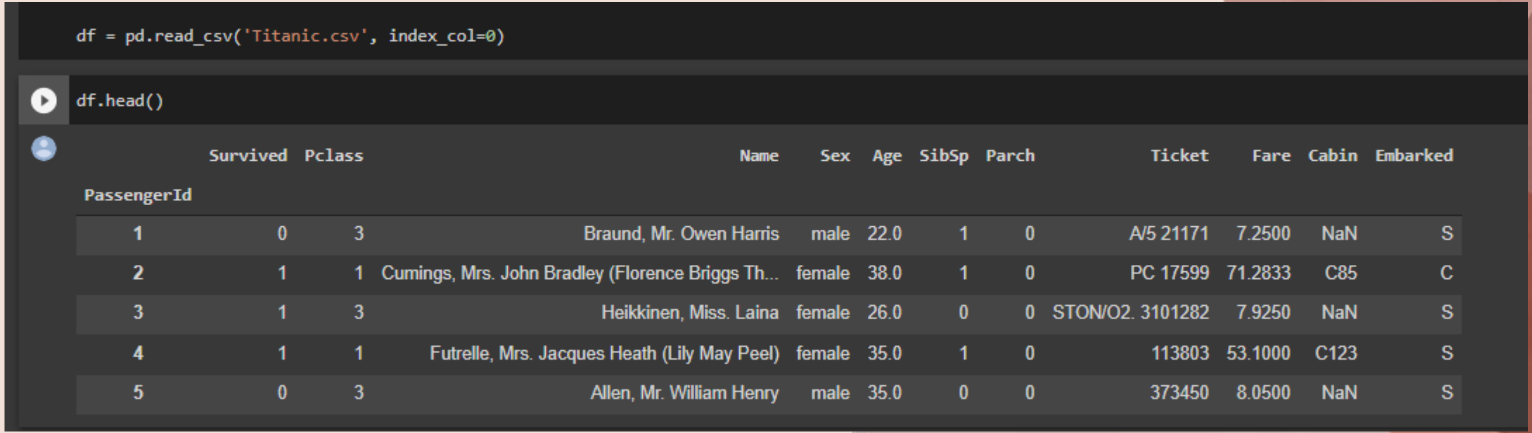


```
[ ] df = pd.read_csv('Titanic.csv')
```

```
#tampilkan dataset yg sudah diload

df.head()
```

```
   PassengerId  Survived  Pclass                                               Name     Sex   Age  SibSp  Parch            Ticket     Fare Cabin Embarked
0            1         0       3                            Braund, Mr. Owen Harris    male  22.0      1      0         A/5 21171   7.2500   NaN        S
1            2         1       1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1      0          PC 17599  71.2833   C85        C
2            3         1       3                             Heikkinen, Miss. Laina  female  26.0      0      0  STON/O2. 3101282   7.9250   NaN        S
3            4         1       1       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1      0            113803  53.1000  C123        S
4            5         0       3                           Allen, Mr. William Henry    male  35.0      0      0            373450   8.0500   NaN        S
```

- ❑ pandas default index starts from zero
- ❑ while the dataset index of the PassengerId column starts from one
- ❑ then we will use the index dataset from column PassengerId

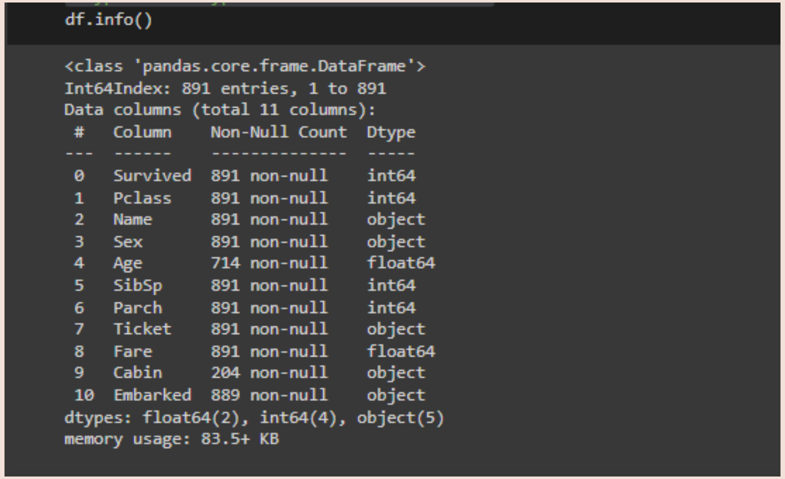


```
df = pd.read_csv('Titanic.csv', index_col=0)
```

```
df.head()
```

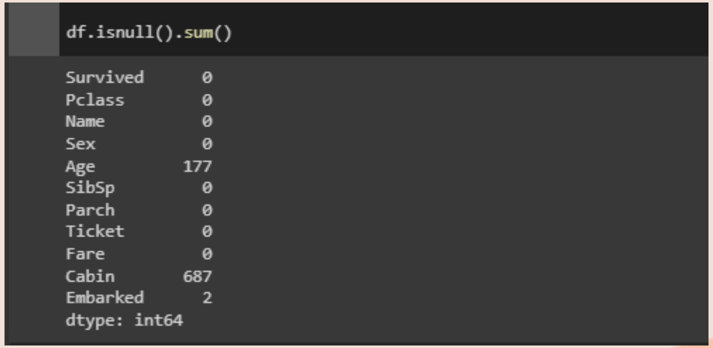
```
             Survived  Pclass                                               Name     Sex   Age  SibSp  Parch            Ticket     Fare  Cabin  Embarked
PassengerId
          1         0       3                            Braund, Mr. Owen Harris    male  22.0      1      0         A/5 21171   7.2500    NaN         S
          2         1       1  Cumings, Mrs. John Bradley (Florence Briggs Th...  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   7.9250    NaN         S
          4         1       1       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1      0            113803  53.1000   C123         S
          5         0       3                           Allen, Mr. William Henry    male  35.0      0      0            373450   8.0500    NaN         S
```

- ❑ Checked data condition
- ❑ Int64Index displays the total number of data entered
- ❑ Non null count is the number of data entered that is not null
- ❑ Dtype is the data type of the column



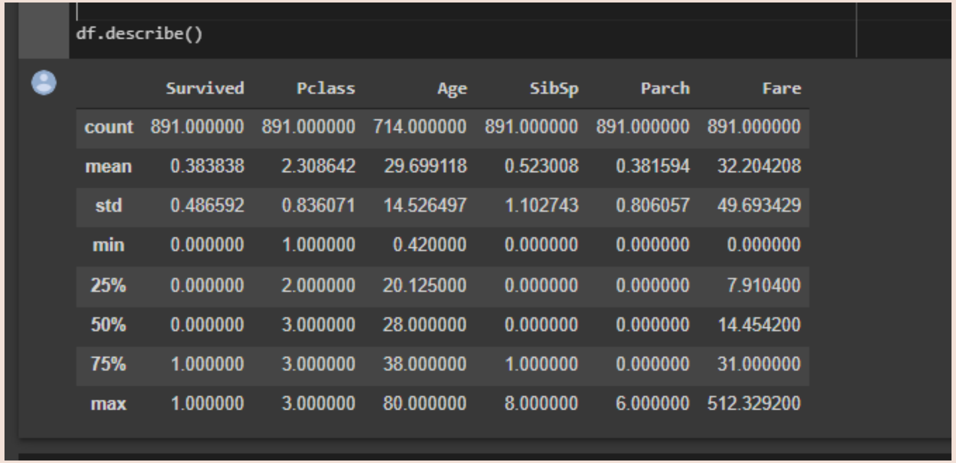
```
df.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Name      891 non-null    object
 3   Sex       891 non-null    object
 4   Age       714 non-null    float64
 5   SibSp     891 non-null    int64
 6   Parch     891 non-null    int64
 7   Ticket    891 non-null    object
 8   Fare      891 non-null    float64
 9   Cabin     204 non-null    object
 10  Embarked  889 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 83.5+ KB
```

- ❑ display the number of NaN from the dataset



```
df.isnull().sum()

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

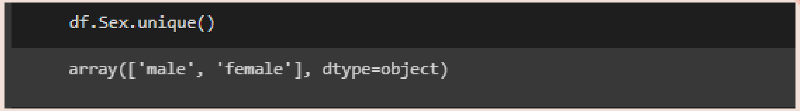
Display calculations from column datasets of type integer or float



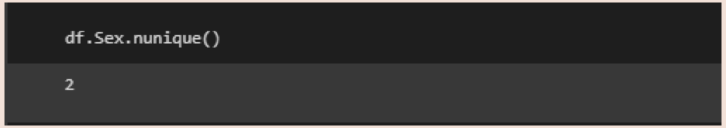
```
df.describe()
```

```
       Survived      Pclass         Age       SibSp       Parch        Fare
count  891.000000  891.000000  714.000000  891.000000  891.000000  891.000000
mean     0.383838    2.308642   29.699118    0.523008    0.381594   32.204208
std      0.486592    0.836071   14.526497    1.102743    0.806057   49.693429
min      0.000000    1.000000    0.420000    0.000000    0.000000    0.000000
25%      0.000000    2.000000   20.125000    0.000000    0.000000    7.910400
50%      0.000000    3.000000   28.000000    0.000000    0.000000   14.454200
75%      1.000000    3.000000   38.000000    1.000000    0.000000   31.000000
max      1.000000    3.000000   80.000000    8.000000    6.000000  512.329200
```

- Show any unique values in that column
- For example from column Sex



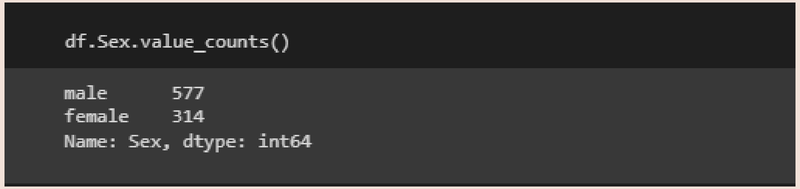
```
df.Sex.unique()

array(['male', 'female'], dtype=object)
```



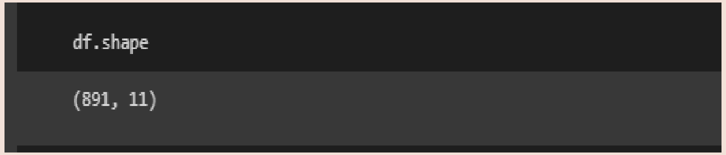
```
df.Sex.nunique()

2
```

- Display the number of unique values in that column

- Display the proportion of its unique data for the categoric data type



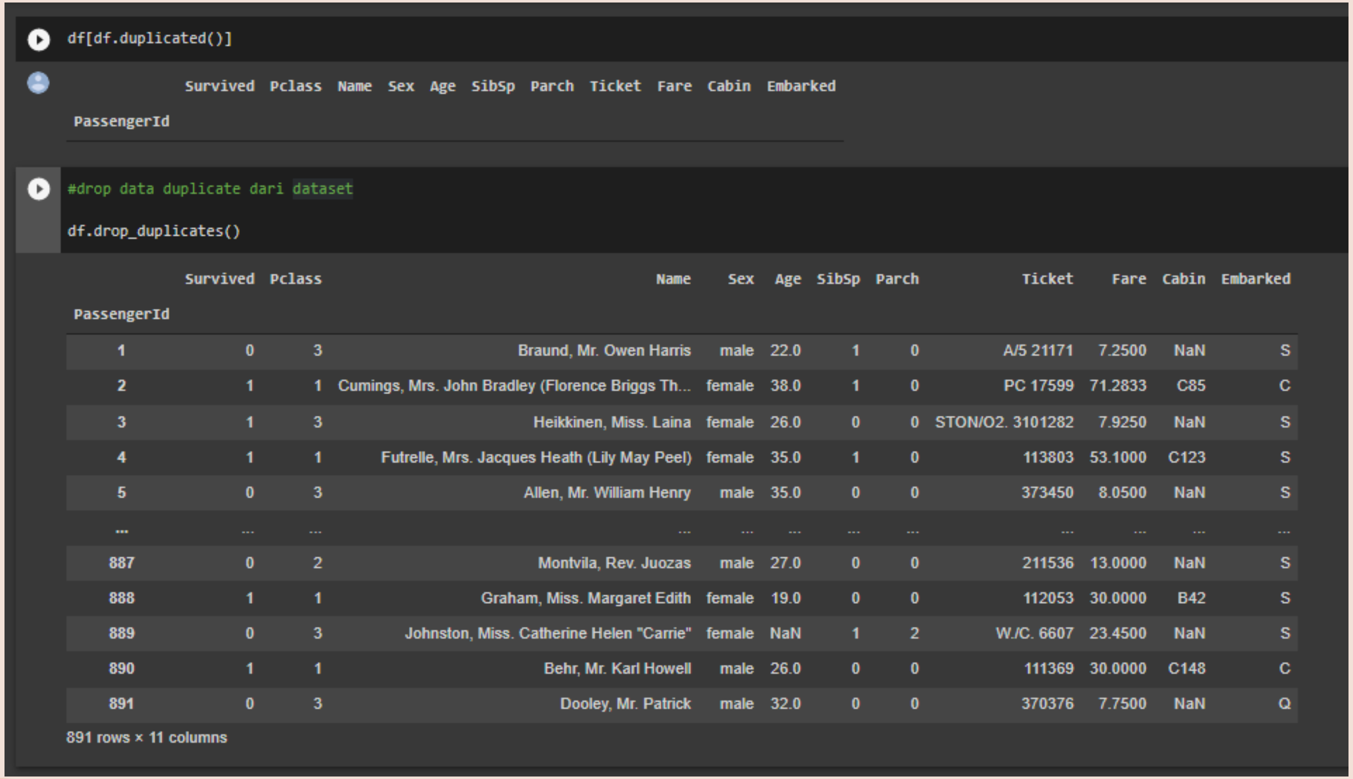
```
df.Sex.value_counts()

male      577
female    314
Name: Sex, dtype: int64
```



```
df.shape

(891, 11)
```

- Display the number of rows and the number of columns of the dataset

Drop duplicate data from dataset



```
df[df.duplicated()]
```

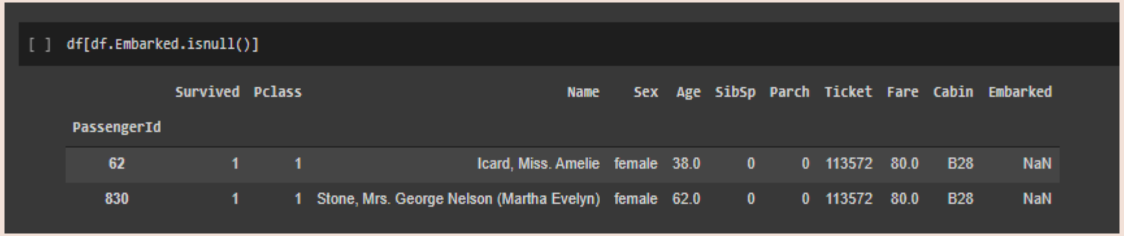
```
           Survived  Pclass  Name  Sex  Age  SibSp  Parch  Ticket  Fare  Cabin  Embarked
PassengerId
```

```
#drop data duplicate dari dataset

df.drop_duplicates()
```

```
             Survived  Pclass                                               Name     Sex   Age  SibSp  Parch            Ticket     Fare  Cabin  Embarked
PassengerId
          1         0       3                            Braund, Mr. Owen Harris    male  22.0      1      0         A/5 21171   7.2500    NaN         S
          2         1       1  Cumings, Mrs. John Bradley (Florence Briggs Th...  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   7.9250    NaN         S
          4         1       1       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1      0            113803  53.1000   C123         S
          5         0       3                           Allen, Mr. William Henry    male  35.0      0      0            373450   8.0500    NaN         S
        ...       ...     ...                                                ...     ...   ...    ...    ...               ...      ...    ...       ...
        887         0       2                              Montvila, Rev. Juozas    male  27.0      0      0            211536  13.0000    NaN         S
        888         1       1                       Graham, Miss. Margaret Edith  female  19.0      0      0            112053  30.0000    B42         S
        889         0       3           Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1      2        W./C. 6607  23.4500    NaN         S
        890         1       1                              Behr, Mr. Karl Howell    male  26.0      0      0            111369  30.0000   C148         C
        891         0       3                                Dooley, Mr. Patrick    male  32.0      0      0            370376   7.7500    NaN         Q
891 rows × 11 columns
```

Column Embarked

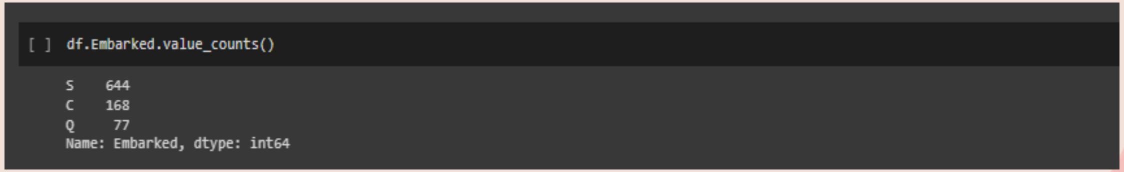
There is null data in the Column “Embarked”. We can know this from the total number of data entries is 891, while in the Column “Embarked” there is 889. Therefore, we need to check where the data zero is



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

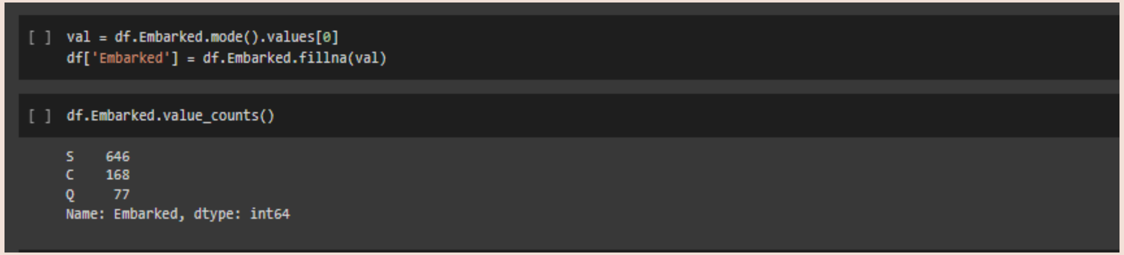
	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
PassengerId											
62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	B28	NaN
830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	B28	NaN

Then we can show the proportion of the Column “Embarked” to see if any data is categorical



```
[ ] df.Embarked.value_counts()

S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

Column Embarked



```
[ ]  val = df.Embarked.mode().values[0]
     df['Embarked'] = df.Embarked.fillna(val)
```

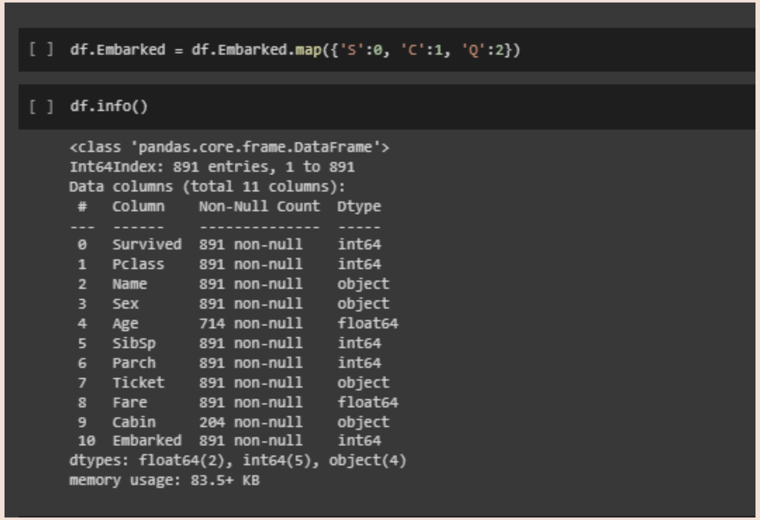
```
[ ]  df.Embarked.value_counts()
```

```
     S    646
     C    168
     Q     77
     Name: Embarked, dtype: int64
```

The imputation uses mode because the result we get from checking the data type of the Column “Embarked” is categoric data. And also from the proportion of Column “Embarked” data is data that appereas frequently, therefore we will use S as the mode

From the imputation, we get the proportion had changed. Before the data from S was 644, now the number of “Embarked” data is 646

Column Embarked



```
[ ]  df.Embarked = df.Embarked.map({'S':0, 'C':1, 'Q':2})
```

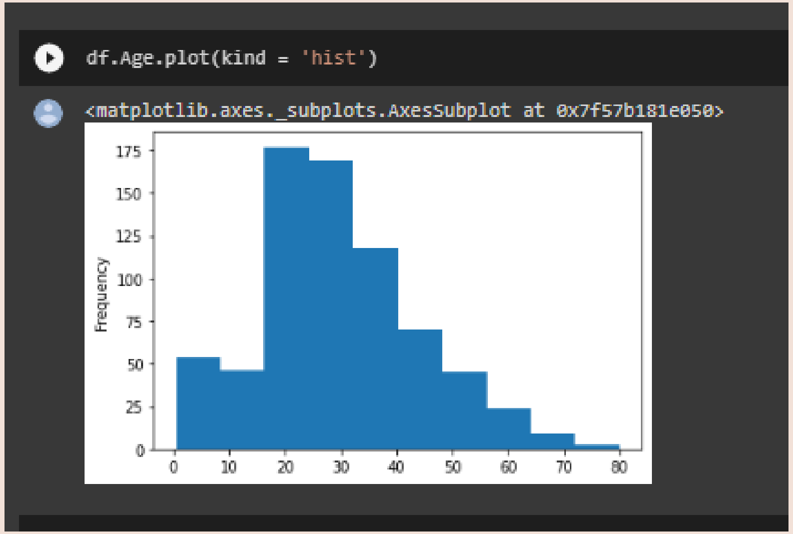
```
[ ]  df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Name      891 non-null    object
 3   Sex       891 non-null    object
 4   Age       714 non-null    float64
 5   SibSp     891 non-null    int64
 6   Parch     891 non-null    int64
 7   Ticket    891 non-null    object
 8   Fare      891 non-null    float64
 9   Cabin     204 non-null    object
 10  Embarked  891 non-null    int64
dtypes: float64(2), int64(5), object(4)
memory usage: 83.5+ KB
```

We have to change the Column “Embarked” data type to “Numerice” type so that we can facilitate the analysis process.

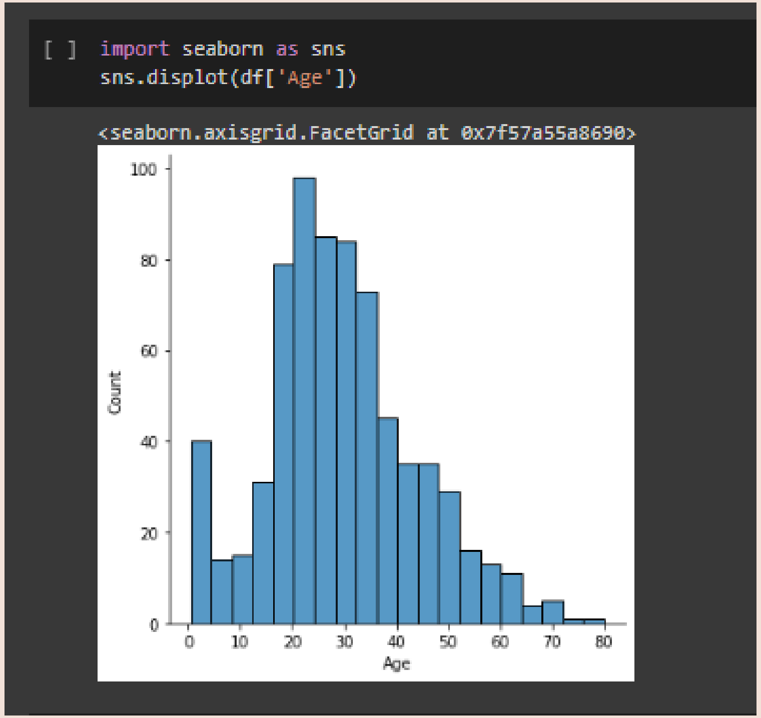
Column Age

There is null data in the Column “Age” because the total number of data entries is 891, while the Column “Age” returns 714



```
df.Age.plot(kind = 'hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f57b181e050>
```

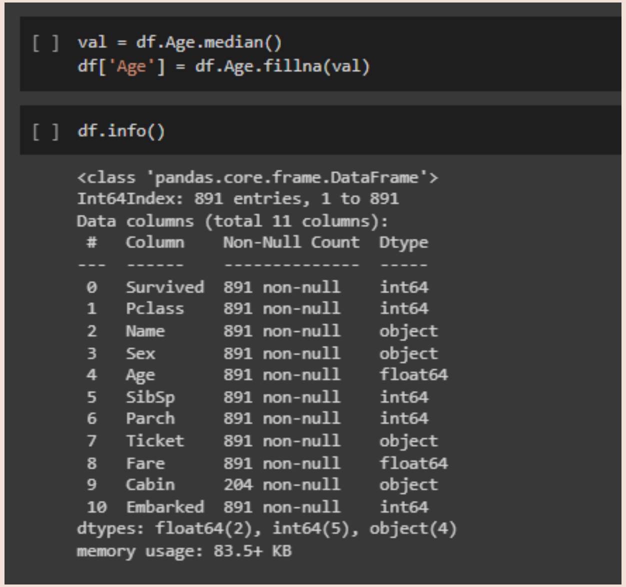


```
[ ] import seaborn as sns
    sns.displot(df['Age'])
```

```
<seaborn.axisgrid.FacetGrid at 0x7f57a55a8690>
```

Column Age

We can use Median to calculated the Column "Age". This is because the Column "Age" has a skewness distribution

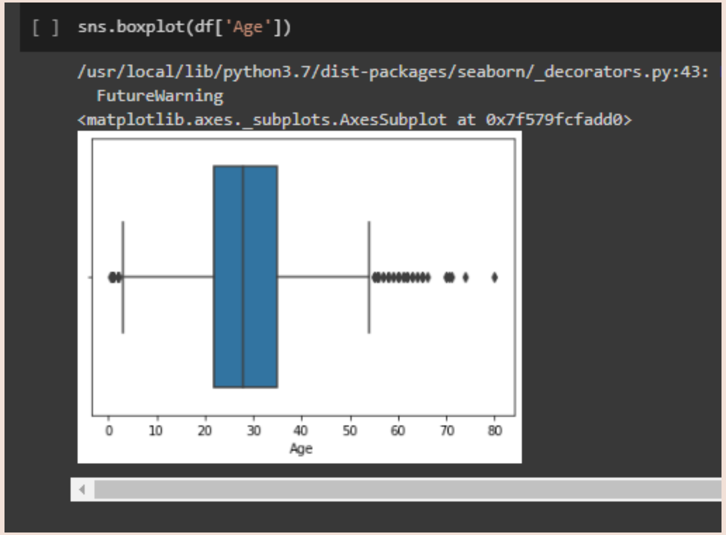


```
[ ] val = df.Age.median()
    df['Age'] = df.Age.fillna(val)
```

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Name      891 non-null    object
 3   Sex       891 non-null    object
 4   Age       891 non-null    float64
 5   SibSp     891 non-null    int64
 6   Parch     891 non-null    int64
 7   Ticket    891 non-null    object
 8   Fare      891 non-null    float64
 9   Cabin     204 non-null    object
 10  Embarked  891 non-null    int64
dtypes: float64(2), int64(5), object(4)
memory usage: 83.5+ KB
```

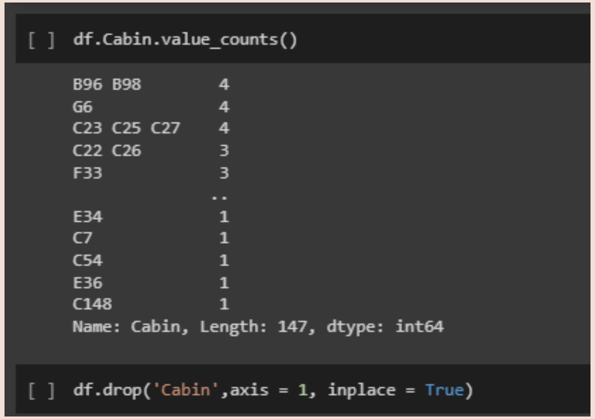
This the visualization



```
[ ] sns.boxplot(df['Age'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f579fcfadd0>
```

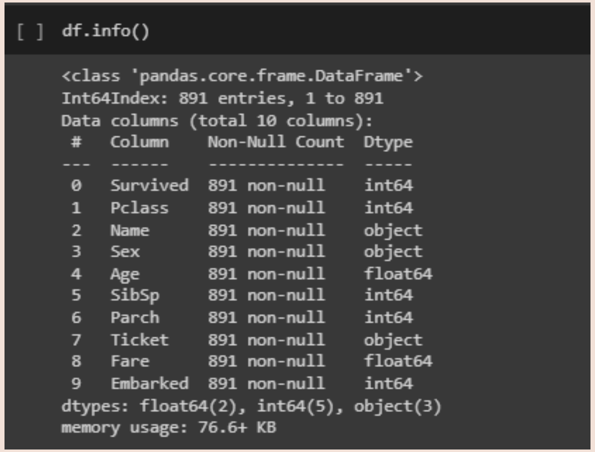
Column Cabin



```
[ ] df.Cabin.value_counts()

    B96 B98        4
    G6             4
    C23 C25 C27    4
    C22 C26        3
    F33            3
                  ..
    E34            1
    C7             1
    C54            1
    E36            1
    C148           1
    Name: Cabin, Length: 147, dtype: int64

[ ] df.drop('Cabin',axis = 1, inplace = True)
```

There is null data in the Column "Cabin". We can know this from the total number of data enteries 891, while in Column "Cabin" there are 204

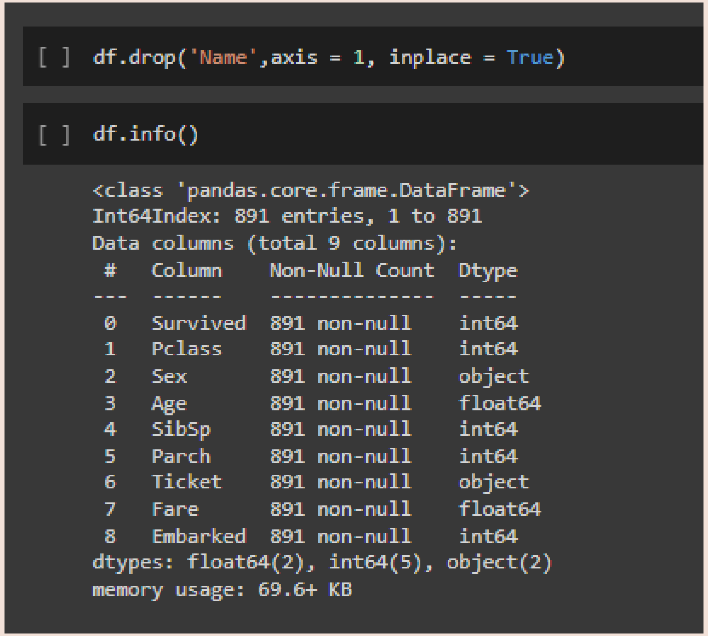
Because Column "Ticket" has too many unique data and the information their give us is not give us many informative to find out "Survived Data". Then we can delete the Column "Ticket"



```
[ ] df.info()

    <class 'pandas.core.frame.DataFrame'>
    Int64Index: 891 entries, 1 to 891
    Data columns (total 10 columns):
     #   Column    Non-Null Count  Dtype
    ---  ------    --------------  -----
     0   Survived  891 non-null    int64
     1   Pclass    891 non-null    int64
     2   Name      891 non-null    object
     3   Sex       891 non-null    object
     4   Age       891 non-null    float64
     5   SibSp     891 non-null    int64
     6   Parch     891 non-null    int64
     7   Ticket    891 non-null    object
     8   Fare      891 non-null    float64
     9   Embarked  891 non-null    int64
    dtypes: float64(2), int64(5), object(3)
    memory usage: 76.6+ KB
```

Column Name

Because Column “Name” has too many unique data and the information their give us is not give us many informative to find out “Survived Data”. Then we can delete the Column “Name”

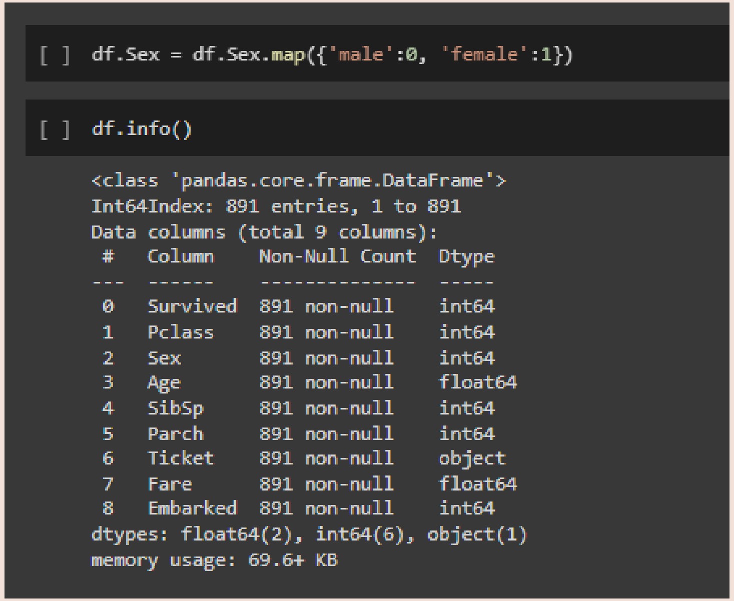


```
[ ] df.drop('Name',axis = 1, inplace = True)
```

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Sex       891 non-null    object
 3   Age       891 non-null    float64
 4   SibSp     891 non-null    int64
 5   Parch     891 non-null    int64
 6   Ticket    891 non-null    object
 7   Fare      891 non-null    float64
 8   Embarked  891 non-null    int64
dtypes: float64(2), int64(5), object(2)
memory usage: 69.6+ KB
```

Column Sex



```
[ ] df.Sex = df.Sex.map({'male':0, 'female':1})
```

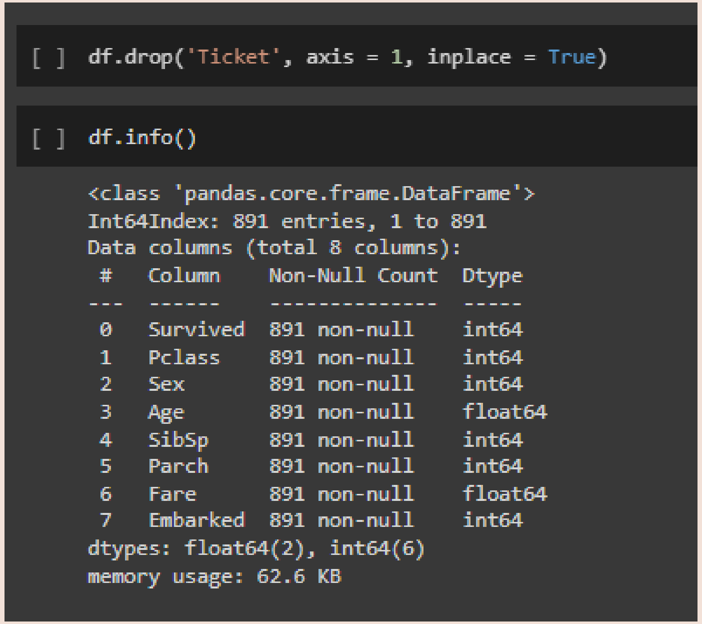
```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Sex       891 non-null    int64
 3   Age       891 non-null    float64
 4   SibSp     891 non-null    int64
 5   Parch     891 non-null    int64
 6   Ticket    891 non-null    object
 7   Fare      891 non-null    float64
 8   Embarked  891 non-null    int64
dtypes: float64(2), int64(6), object(1)
memory usage: 69.6+ KB
```

We have to change the Column “Sex” data type to “Numerice” type so that we can facilitate the analysis process

Column Ticket

Because Column “Ticket” has too many unique data and the information their give us is not give us many informative to find out “Survived Data”. Then we can delete the Column “Ticket”

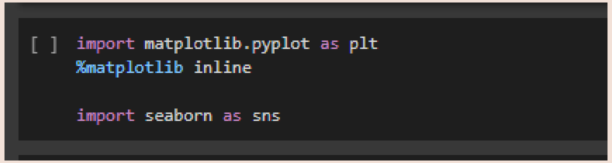


```
[ ]  df.drop('Ticket', axis = 1, inplace = True)
```

```
[ ]  df.info()
```

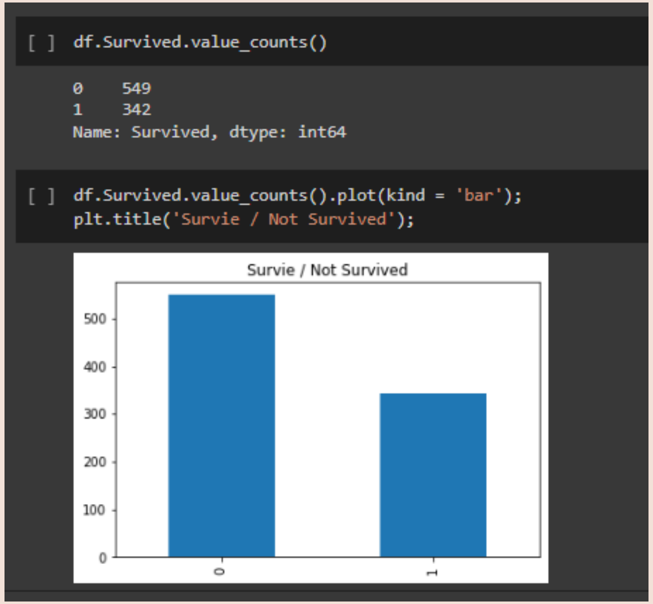
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  891 non-null    int64  
 1   Pclass    891 non-null    int64  
 2   Sex       891 non-null    int64  
 3   Age       891 non-null    float64
 4   SibSp     891 non-null    int64  
 5   Parch     891 non-null    int64  
 6   Fare      891 non-null    float64
 7   Embarked  891 non-null    int64  
dtypes: float64(2), int64(6)
memory usage: 62.6 KB
```

Survived Data Visualization



```
[ ]  import matplotlib.pyplot as plt
     %matplotlib inline

     import seaborn as sns
```

import the required packages

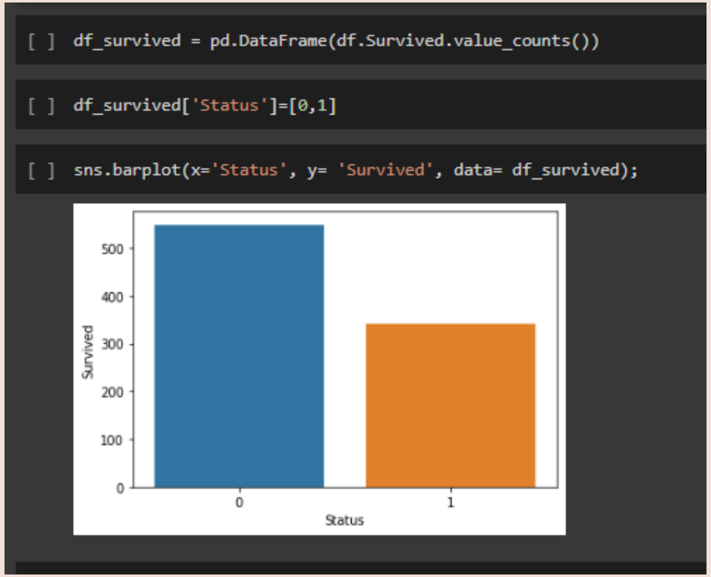


```
[ ]  df.Survived.value_counts()

     0    549
     1    342
     Name: Survived, dtype: int64

[ ]  df.Survived.value_counts().plot(kind = 'bar');
     plt.title('Survie / Not Survived');
```

Display the propostion of Survived Data for visualization them

Survived Data Visualization

Create data frame from the Column Survived

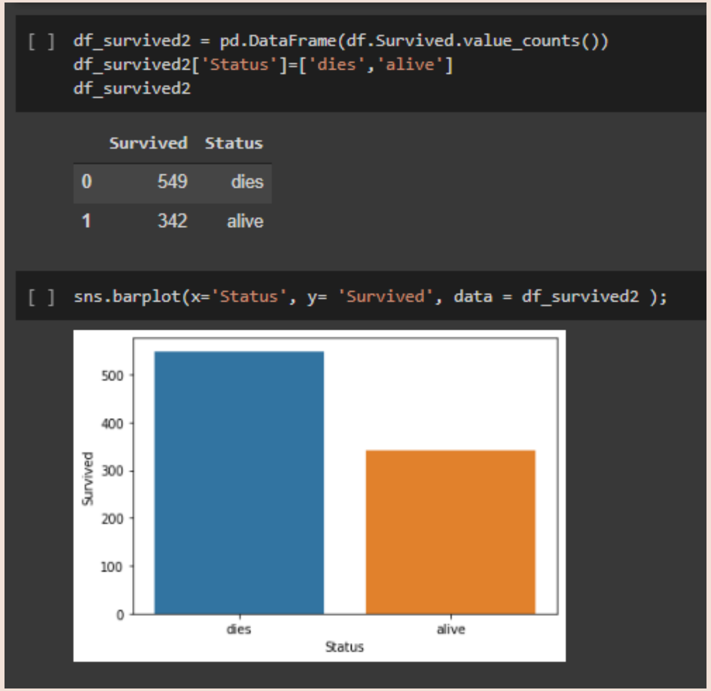


```
[ ]  df_survived = pd.DataFrame(df.Survived.value_counts())
```

```
[ ]  df_survived['Status']=[0,1]
```

```
[ ]  sns.barplot(x='Status', y= 'Survived', data= df_survived);
```

Change the status description



```
[ ]  df_survived2 = pd.DataFrame(df.Survived.value_counts())
     df_survived2['Status']=['dies','alive']
     df_survived2
```

	Survived	Status
0	549	dies
1	342	alive

```
[ ]  sns.barplot(x='Status', y= 'Survived', data = df_survived2 );
```

Thank You