I

Exploratoy, Cleaning and Visualizing Data From the Titanic.csv Files

PANDAS GROUP
DATA SCIENCE TRACK B

AJIE PRASETYA, DEWA AYU RAI I.M, M. RAMADHONI, VIVI INDAH FITRIANI, FIRDAUSI NURUS SA'IDAH

Data Cleaning

Import some libraries that will be needed for the analysis

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

Create a dataframe then load the dataset. The dataset used here is data Titanic.csv

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

```
Choose Files  No file chosen
Saving Titanic.csv to Titanic.csv
{'Titanic.csv': b'PassengerId,Sur
```

```
df = pd.read_csv("Titanic.csv")
```

Show the dataset that has been loaded

```
[ ] 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

let's check the data condition

```
[ ] 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
```

we will see a statistical summary of the imported dataset using pandas.describe() method.

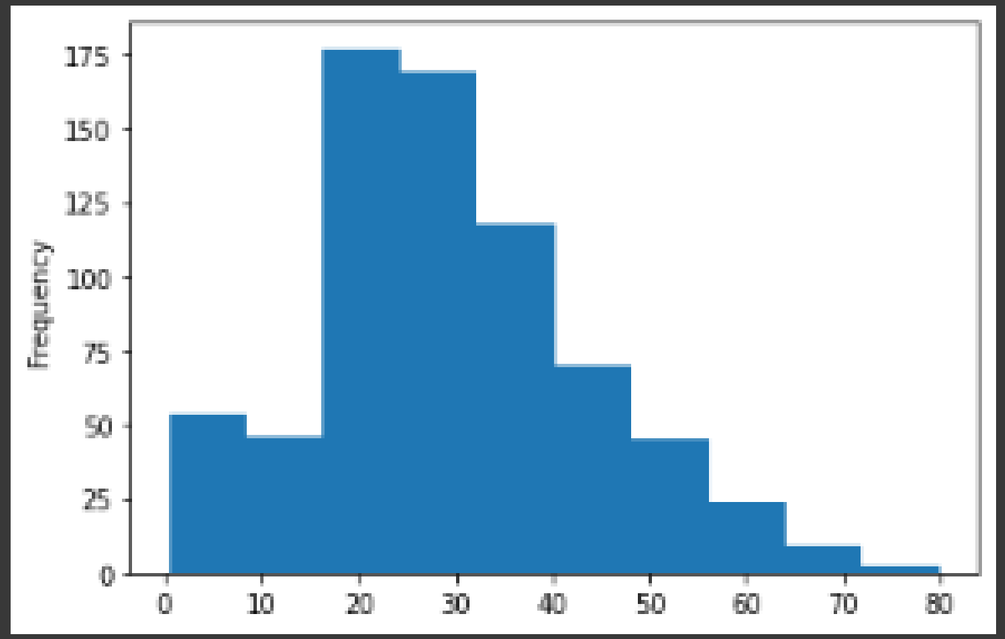
```
[ ] df.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Column Age

The total number of data entries is 891, while the Age column is 714. It means there is null data in column Age. We will do imputation on column Age to determine what methods we will use in the imputation column Age then we will show the histogram column Age

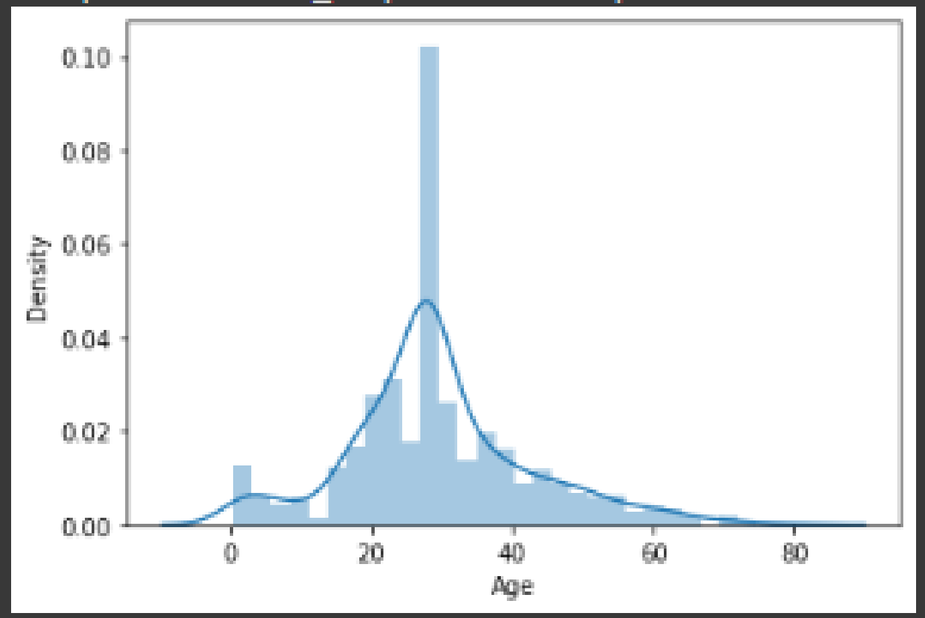
```
[ ] df.Age.plot(kind='hist');
```



Next Step

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

```
/usr/local/lib/python3.7/dist-packages/seaborn/distri
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7fe01f0d7
```




```
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
```

Because column Age has a skewness distribution, then we will imputation on column Age using median

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

Because column Age has a skewness distribution, then we will imputation on column Age using median

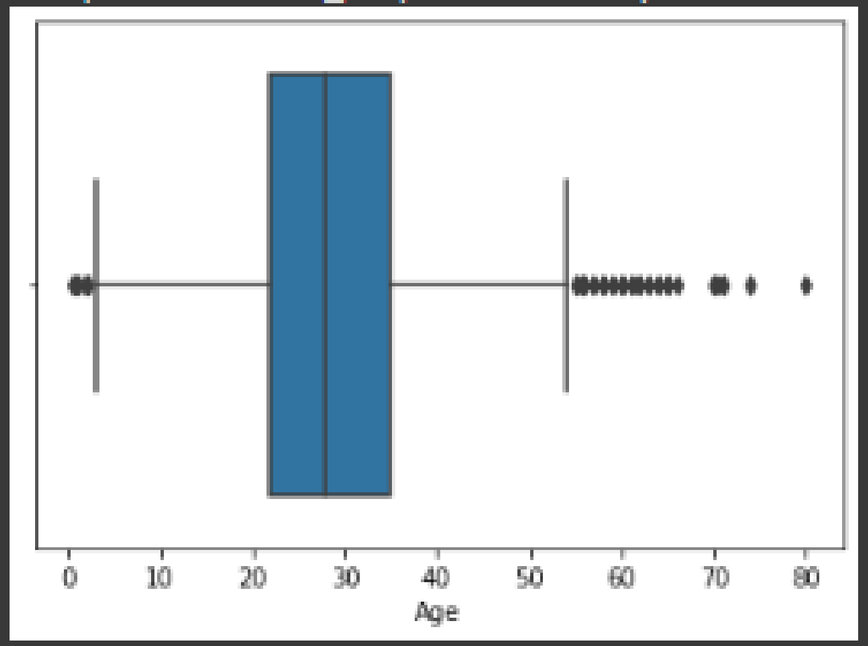
```
[ ] df.info()

    <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
```

Display boxplot visualization of age column

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

```
/usr/local/lib/python3.7/dist-packages/seaborn/_
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fe0
```



Cabin Column

The total number of data entries is 891, while the Cabin column is 204. This means that there is null data in the cabin column.

Show proportion of data 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
```

It can be seen that the Cabin column value has too many unique data and also the Cabin info is not very informative to find out the survived data. Then we will delete the column cabin

```
[ ] 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
```

Embarked Column

The total number of data entry is 891, while the Embarked column is 889, meaning that there is null data in the embarked column. We will check where the null data is

```
[ ] 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
```

Show the proportions of the Embarked column data, it turns out that the Embarked column data is in the form of categorical data.

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

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

When we are going to do imputation on the Embarked columnthen we check the Embarked column data type first, The data column Embarked is in the form of categoric data, so the imputation uses the mode and From the Embarked column proportion, S is the data that appears most often, then S is the mode (mode)

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

After the imputation, it can be seen that the proportion has changed

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

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

Because the Embarked column is still an Object data type, to facilitate the analysis process we will convert the object data type to a numeric type

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

Display dataset info to see if the Embarked column has changed its data type.

```
[ ] 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     891 non-null    int64
dtypes: float64(2), int64(6), object(3)
memory usage: 76.7+ KB
```

It turns out that the Embarked column has now changed its data type to numeric

Sibsp Column and Parch Column

Manipulate to make it easier for the data to be read by machines. created a new column showing whether he was alone or with family.

```
[25] df['alone']=df['SibSp']+df['Parch']
```

```
df['alone'][df['alone']>0]='with family'
df['alone'][df['alone']==0]='withou 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
```

show new data view

```
[27] df.head()
```

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

Relationship Between Column Sex and Column Survived

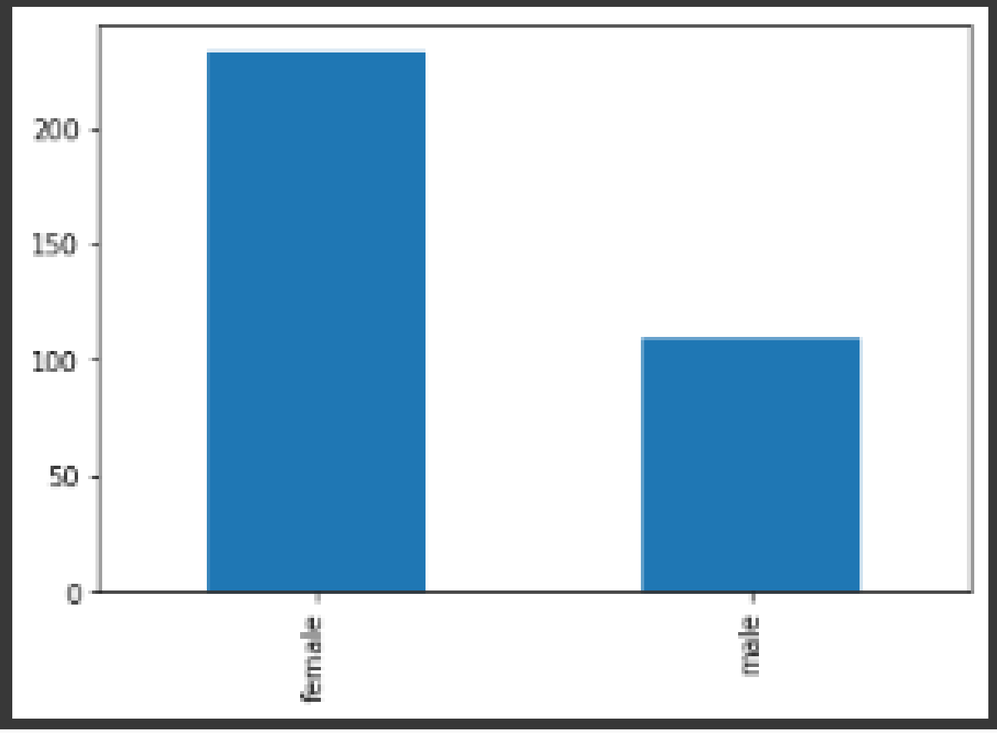
See the proportion of the sex column that survived

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

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

Display the visualization of the sex column that survived

```
[29] df.Sex[df['Survived']==1].value_counts().plot(kind='bar');
```

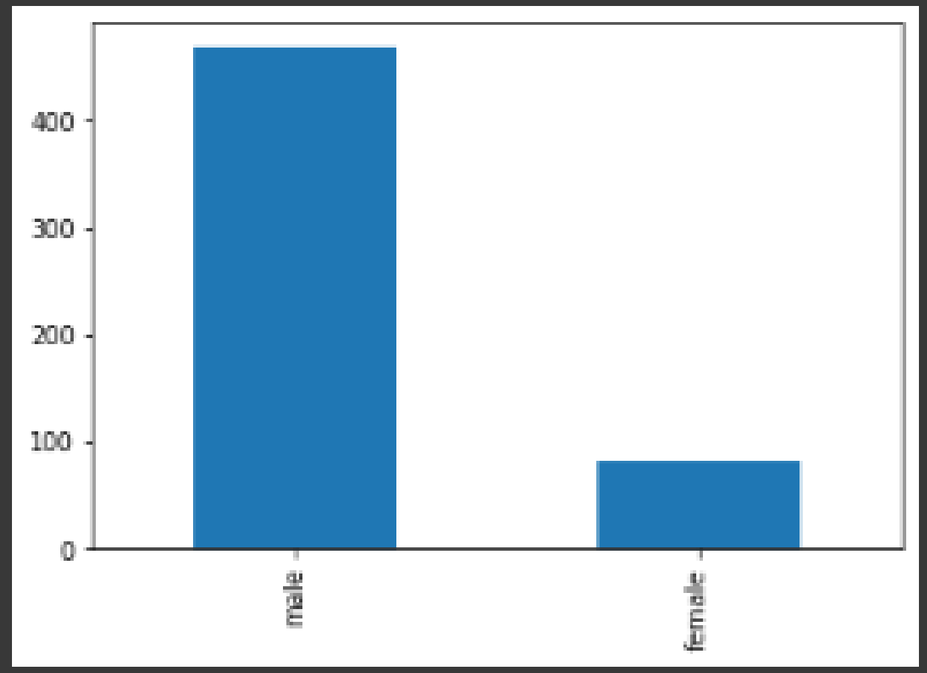


Next Step

```
[30] df.Sex[df['Survived']==0].value_counts()
```

```
male      468
female     81
Name: Sex, dtype: int64
```

```
[31] df.Sex[df['Survived']==0].value_counts().plot(kind='bar');
```



Name

Because the name column is too much unique and also less informative data then here we will delete the column.

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

Now we'll see the change

```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 9 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   Sex          891 non-null    int64
 4   Age          891 non-null    float64
 5   SibSp        891 non-null    int64
 6   Parch        891 non-null    int64
 7   Fare         891 non-null    float64
 8   Embarked     891 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 62.8 KB
```

Sex

Sex column is still a type of data object, so here we will change the data type to a numeric data type to make it easier to use

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

Now we will see the tyoe data that has been changed

```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 9 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   Sex          0 non-null      float64
 4   Age          891 non-null    float64
 5   SibSp        891 non-null    int64
 6   Parch        891 non-null    int64
 7   Fare         891 non-null    float64
 8   Embarked     891 non-null    int64
dtypes: float64(3), int64(6)
memory usage: 62.8 KB
```

Ticket

Because we don't need a ticket column then we can mneghapusnya with the drop function

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

Now we will see the deleted column

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 9 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   Sex          0 non-null      float64
 4   Age          891 non-null    float64
 5   SibSp        891 non-null    int64
 6   Parch        891 non-null    int64
 7   Fare         891 non-null    float64
 8   Embarked     891 non-null    int64
dtypes: float64(3), int64(6)
memory usage: 62.8 KB
```

Visualisasi Data Survived

Import package/library that we will use

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

    import seaborn as sns
```

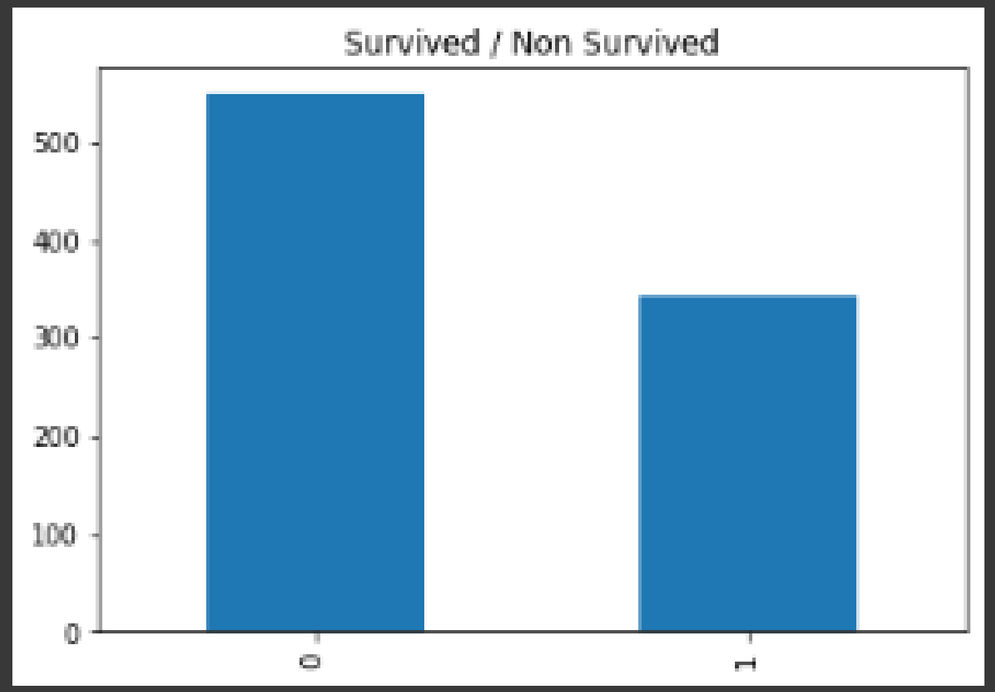
We will display the calculated value of the Survived column

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

    0    549
    1    342
    Name: Survived, dtype: int64
```

Now let's display the visualization results in the form of a bar chart

```
[ ] df.Survived.value_counts().plot(kind='bar');
    plt.title('Survived / Non Survived');
```



Here we will create frame data from the survived column

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

Show info from survived column

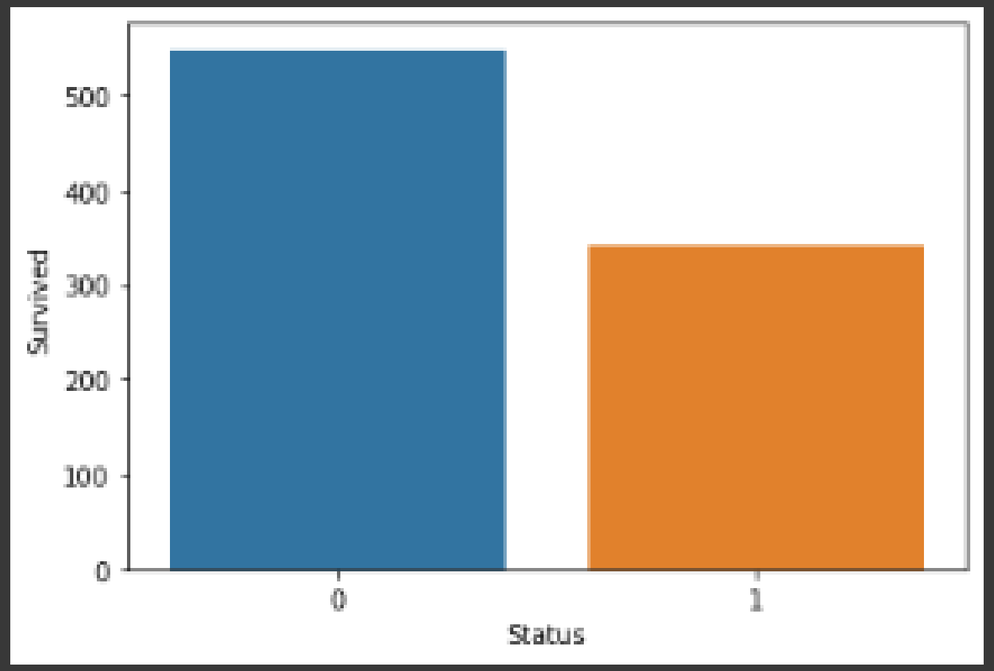
```
[ ] df_survived['Status']=[0,1]
```

```
[ ] df_survived
```

	Survived	Status
0	549	0
1	342	1

Now we will see the changes

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



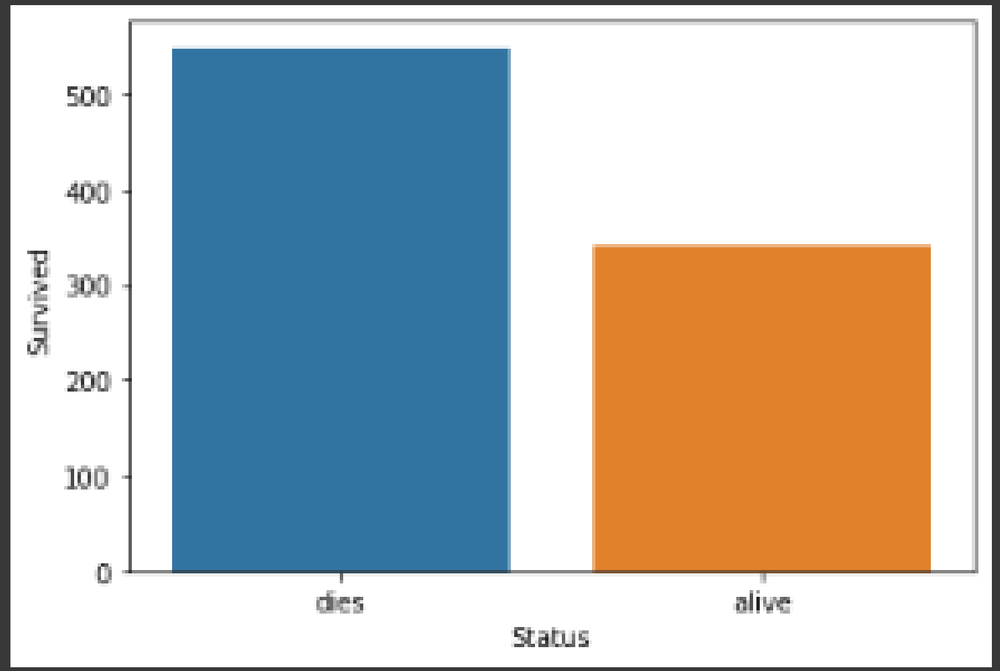
Now we will change the status to dies and alive

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

	Survived	Status
0	549	dies
1	342	alive

Now lets show the visualization result

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



Thank You