

Data Mining

Lab - 3

Name : Vyas Bhagyesh Y.

Div : 5-B

Batch : 2

Roll-No : 423

Enrollment No : 23010101662

1) First, you need to read the titanic dataset from local disk and display first five records

In [16]:

```
import pandas as pd
from pandas.api.types import is_numeric_dtype
import matplotlib as plt
```

In [4]:

```
df=pd.read_csv('titanic.csv')
```

In [5]:

```
df
```

Out[5]:

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

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
4	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...
886	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows x 12 columns

2) Identify Nominal, Ordinal, Binary and Numeric attributes from data sets and display all values.

In [11]:

```
df.columns
```

Out[11]:

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'], dtype='object')
```

In [10]:

```
df['PassengerId'] #nominal,numeric
```

Out[10]:

```
0      1
1      2
2      3
3      4
4      5
...
886    887
887    888
888    889
889    890
890    891
Name: PassengerId, Length: 891, dtype: int64
```

In [8]:

```
df.Sex
```

Out[8]:

```
0      male
1    female
2    female
3    female
4      male
...
886    male
887    female
888    female
889    male
890    male
Name: Sex, Length: 891, dtype: object
```

```
In [12]:
```

```
df.Survived #binary
```

```
Out[12]:
```

```
0      0
1      1
2      1
3      1
4      0
```

```
..
886    0
887    1
888    0
889    1
890    0
```

```
Name: Survived, Length: 891, dtype: int64
```

```
In [ ]:
```

```
df.Pclass #ordinal
```

```
In [ ]:
```

```
df.Name
```

```
In [ ]:
```

```
df.Embarked
```

```
In [15]:
```

```
print("Nominal : ")
print(df['Name'])
print(df.PassengerId)
print(df.Ticket)
print(df.Cabin)
print(df.Embarked)
```

```
print("Ordinal : ")
print(df.Pclass)
print(df.PassengerId)
```

```
print("Binary : ")
print(df.Sex)
print(df.Survived)
```

```
print("Numeric : ")
print(df.Fare)
print(df.PassengerId)
print(df.Age)
print(df.)
```

```
Nominal :
```

```
0      Braund, Mr. Owen Harris
1  Cumings, Mrs. John Bradley (Florence Briggs Th...
2      Heikkinen, Miss. Laina
3  Futrelle, Mrs. Jacques Heath (Lily May Peel)
4      Allen, Mr. William Henry
```

```
...
886      Montvila, Rev. Juozas
887      Graham, Miss. Margaret Edith
888  Johnston, Miss. Catherine Helen "Carrie"
889      Behr, Mr. Karl Howell
890      Dooley, Mr. Patrick
```

```
Name: Name, Length: 891, dtype: object
```

3) Identify symmetric and asymmetric binary attributes from data sets and display all values.

In []:

```
print("Asymmetric : ")
print(df.Survived)
print("Symmetric : ")
print(df.Sex)
```

4) For each quantitative attribute, calculate its average, standard deviation, minimum, mode, range and maximum values.

In [23]:

```
for col in df.columns:
    if is_numeric_dtype(df[col].dtype):
        print(f"{col} : ")
        print("\tMean = ", df[col].mean())
        print("\tStandard Deviation = ", df[col].std())
        print("\tMaximum = ", df[col].max())
        print("\tMinimum = ", df[col].min())
        #print("\tAverage = ", df[col].average())
        if col!="PassengerId":
            print("\tMode = ", df[col].mode()[0])
            print("\tRange = ", df[col].max()-df[col].min())
```

```
PassengerId :
Mean = 446.0
Standard Deviation = 257.3538420152301
Maximum = 891
Minimum = 1
Range = 890
Survived :
Mean = 0.3838383838383838
Standard Deviation = 0.4865924542648575
Maximum = 1
Minimum = 0
Mode = 0
Range = 1
Pclass :
Mean = 2.308641975308642
Standard Deviation = 0.836071240977049
Maximum = 3
Minimum = 1
Mode = 3
Range = 2
Age :
Mean = 29.69911764705882
Standard Deviation = 14.526497332334042
Maximum = 80.0
Minimum = 0.42
Mode = 24.0
Range = 79.58
SibSp :
Mean = 0.5230078563411896
Standard Deviation = 1.1027434322934317
Maximum = 8
Minimum = 0
Mode = 0
Range = 8
Parch :
Mean = 0.38159371492704824
Standard Deviation = 0.8060572211299483
Maximum = 6
Minimum = 0
Mode = 0
Range = 6
Fare :
Mean = 32.204207968574636
Standard Deviation = 49.6934285971809
Maximum = 512.3292
Minimum = 0.0
Mode = 0.05
```

```
Mode = 0.00  
Range = 512.3292
```

In [5]:

```
PassengerId:  
  Mean = 446.00  
  Standard deviation = 257.35  
  Minimum = 1.00  
  Maximum = 891.00  
Survived:  
  Mean = 0.38  
  Standard deviation = 0.49  
  Minimum = 0.00  
  Maximum = 1.00  
Pclass:  
  Mean = 2.31  
  Standard deviation = 0.84  
  Minimum = 1.00  
  Maximum = 3.00  
Age:  
  Mean = 29.70  
  Standard deviation = 14.53  
  Minimum = 0.42  
  Maximum = 80.00  
SibSp:  
  Mean = 0.52  
  Standard deviation = 1.10  
  Minimum = 0.00  
  Maximum = 8.00  
Parch:  
  Mean = 0.38  
  Standard deviation = 0.81  
  Minimum = 0.00  
  Maximum = 6.00  
Fare:  
  Mean = 32.20  
  Standard deviation = 49.69  
  Minimum = 0.00  
  Maximum = 512.33
```

6) For the qualitative attribute (class), count the frequency for each of its distinct values.

In [32]:

```
df.Pclass.value_counts()
```

Out[32]:

```
Pclass  
3      491  
1      216  
2      184  
Name: count, dtype: int64
```

In [31]:

```
df.Embarked.value_counts()
```

Out[31]:

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

7) It is also possible to display the summary for all the attributes simultaneously in a table

7) it is also possible to display the summary for all the attributes simultaneously in a table using the describe() function. If an attribute is quantitative, it will display its mean, standard deviation and various quantiles (including minimum, median, and maximum) values. If an attribute is qualitative, it will display its number of unique values and the top (most frequent) values.

In [34]:

```
df.describe(include='all')
```

Out[34]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
count	891.000000	891.000000	891.000000	891	891	714.000000	891.000000	891.000000	891	891.000000	204
unique	NaN	NaN	NaN	891	2	NaN	NaN	NaN	681	NaN	147
top	NaN	NaN	NaN	Braund, Mr. Owen Harris	male	NaN	NaN	NaN	347082	NaN	B96 B98
freq	NaN	NaN	NaN	1	577	NaN	NaN	NaN	7	NaN	4
mean	446.000000	0.383838	2.308642	NaN	NaN	29.699118	0.523008	0.381594	NaN	32.204208	NaN
std	257.353842	0.486592	0.836071	NaN	NaN	14.526497	1.102743	0.806057	NaN	49.693429	NaN
min	1.000000	0.000000	1.000000	NaN	NaN	0.420000	0.000000	0.000000	NaN	0.000000	NaN
25%	223.500000	0.000000	2.000000	NaN	NaN	20.125000	0.000000	0.000000	NaN	7.910400	NaN
50%	446.000000	0.000000	3.000000	NaN	NaN	28.000000	0.000000	0.000000	NaN	14.454200	NaN
75%	668.500000	1.000000	3.000000	NaN	NaN	38.000000	1.000000	0.000000	NaN	31.000000	NaN
max	891.000000	1.000000	3.000000	NaN	NaN	80.000000	8.000000	6.000000	NaN	512.329200	NaN

8) For multivariate statistics, you can compute the covariance and correlation between pairs of attributes.

In [45]:

```
df.cov(numeric_only=True)
```

Out[45]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	66231.000000	-0.626966	-7.561798	138.696504	-16.325843	-0.342697	161.883369
Survived	-0.626966	0.236772	-0.137703	-0.551296	-0.018954	0.032017	6.221787
Pclass	-7.561798	-0.137703	0.699015	-4.496004	0.076599	0.012429	-22.830196
Age	138.696504	-0.551296	-4.496004	211.019125	-4.163334	-2.344191	73.849030
SibSp	-16.325843	-0.018954	0.076599	-4.163334	1.216043	0.368739	8.748734
Parch	-0.342697	0.032017	0.012429	-2.344191	0.368739	0.649728	8.661052
Fare	161.883369	6.221787	-22.830196	73.849030	8.748734	8.661052	2469.436846

In [44]:

```
df.corr(numeric_only=True)
```

Out[44]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307

Pclass	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
Age	0.036847	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.096067
SibSp	-0.057527	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.159651
Parch	-0.001652	0.081629	0.018443	-0.189119	0.414838	1.000000	0.216225
Fare	0.012658	0.257307	-0.549500	0.096067	0.159651	0.216225	1.000000

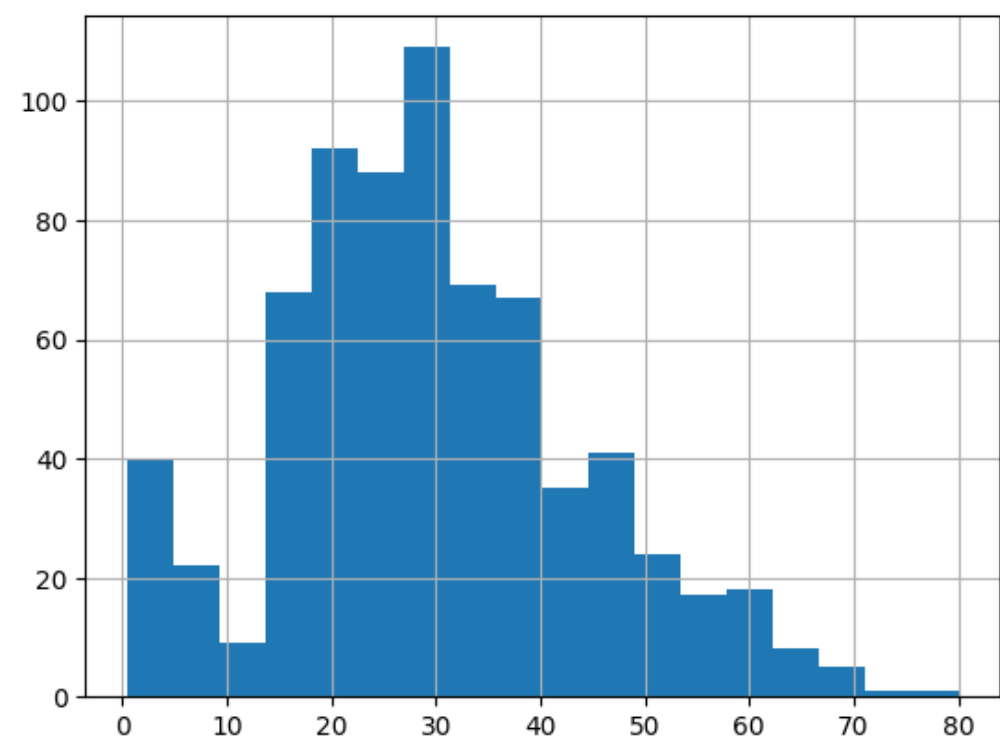
9) Display the histogram for Age attribute by discretizing it into 8 separate bins and counting the frequency for each bin.

In [52]:

```
df.Age.hist(bins=8)
```

Out[52]:

<Axes: >



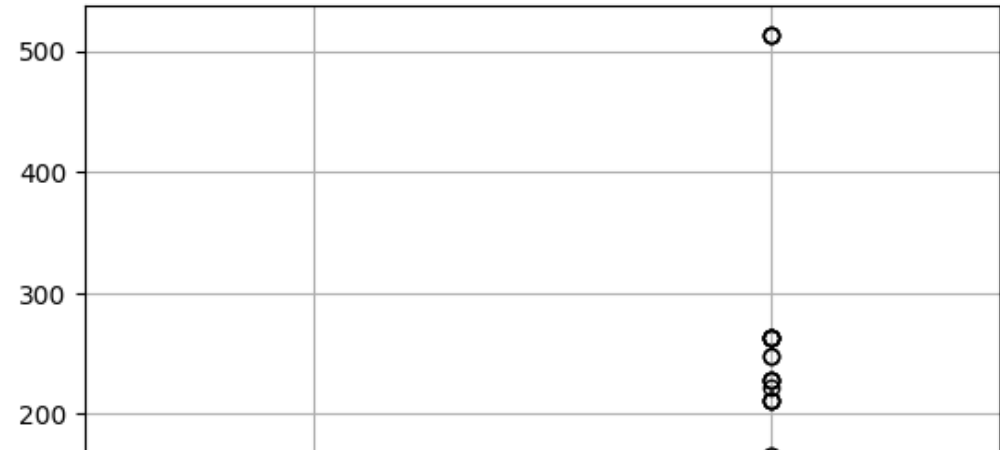
10) A boxplot can also be used to show the distribution of values for each attribute.

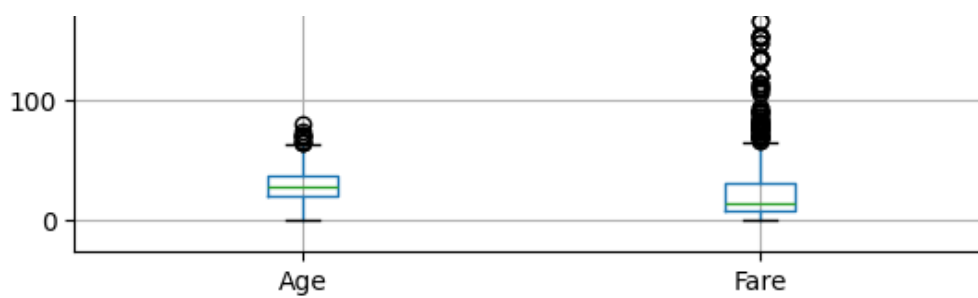
In [62]:

```
df.boxplot(column=['Age', 'Fare'])
```

Out[62]:

<Axes: >





11) Display scatter plot for any 5 pair of attributes , we can use a scatter plot to visualize their joint distribution.

In [67]:

```
df.plot.scatter(x='Age',y='Fare')
```

Out[67]:

<Axes: xlabel='Age', ylabel='Fare'>

