Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Data Mining

Lab - 3

NAME: AYUSH J. MARADIA

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

In [1]:
```
import pandas as pd
```

In [2]:
```
df=pd.read_csv("titanic.csv")
```

In [3]:
```
df.head(5)
```

Out[3]:

	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

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

In [4]:
```
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["SibSp"])
print(df["Parch"])
print(df["Age"])
```

```
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
```

```
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
0             A/5 21171
1              PC 17599
2      STON/O2. 3101282
3                113803
4                373450
             ...
886              211536
887              112053
888          W./C. 6607
889              111369
890              370376
Name: Ticket, Length: 891, dtype: object
0       NaN
1       C85
2       NaN
3      C123
4       NaN
       ...
886     NaN
887     B42
888     NaN
889    C148
890     NaN
Name: Cabin, Length: 891, dtype: object
0      S
1      C
2      S
3      S
4      S
      ..
886    S
887    S
888    S
889    C
890    Q
Name: Embarked, Length: 891, dtype: object
Ordinal:
0      3
1      1
2      3
3      1
4      3
      ..
886    2
887    1
888    3
889    1
890    3
Name: Pclass, Length: 891, dtype: int64
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
Binary:
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
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
Numeric:
0       7.2500
1      71.2833
2       7.9250
3      53.1000
4       8.0500
        ...
886    13.0000
887    30.0000
888    23.4500
889    30.0000
890     7.7500
Name: Fare, Length: 891, dtype: float64
0      1
1      1
2      0
3      1
4      0
      ..
886    0
887    0
888    1
889    0
890    0
Name: SibSp, Length: 891, dtype: int64
0      0
1      0
2      0
3      0
4      0
      ..
886    0
887    0
888    2
889    0
890    0
Name: Parch, Length: 891, dtype: int64
0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
886    27.0
887    19.0
888     NaN
889    26.0
890    32.0
Name: Age, Length: 891, dtype: float64
```

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

In [5]:
```
print("Symetric",df["Sex"])
print("Asymetric",df["Survived"])
```

```
Symetric 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
Asymetric 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
```

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

In [59]:
```
columns = ['PassengerId','Survived','Pclass','Age','SibSp','Parch','Fare']
for i in columns:
    print(i)
    print(f"{" "*6}Mean: {df[i].mean():.2f}")
    print(f"{" "*6}Standard Deviation: {df[i].std():.2f}")
    print(f"{" "*6}Minimum: {df[i].min():.2f}")
    print(f"{" "*6}Maximum: {df[i].max():.2f}")
```

```
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 [60]:
```
df['Pclass'].value_counts()
```

Out[60]:
```
Pclass
3    491
1    216
2    184
Name: count, dtype: int64
```

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 [61]:
```
df.describe(include = 'all')
```

Out[61]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Emb
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	

In [62]:
```
df['Fare'].describe()
```

Out[62]:
```
count    891.000000
mean      32.204208
std       49.693429
min        0.000000
25%        7.910400
50%       14.454200
75%       31.000000
max      512.329200
Name: Fare, dtype: float64
```

In [63]:
```
df.describe(include = 'object')
```

Out[63]:

	Name	Sex	Ticket	Cabin	Embarked
count	891	891	891	204	889
unique	891	2	681	147	3
top	Braund, Mr. Owen Harris	male	347082	B96 B98	S
freq	1	577	7	4	644

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

In [67]:
```
df.corr(numeric_only = True)
```

Out[67]:

	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	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500
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

In [68]:
```
df.cov(numeric_only = True)
```

Out[68]:

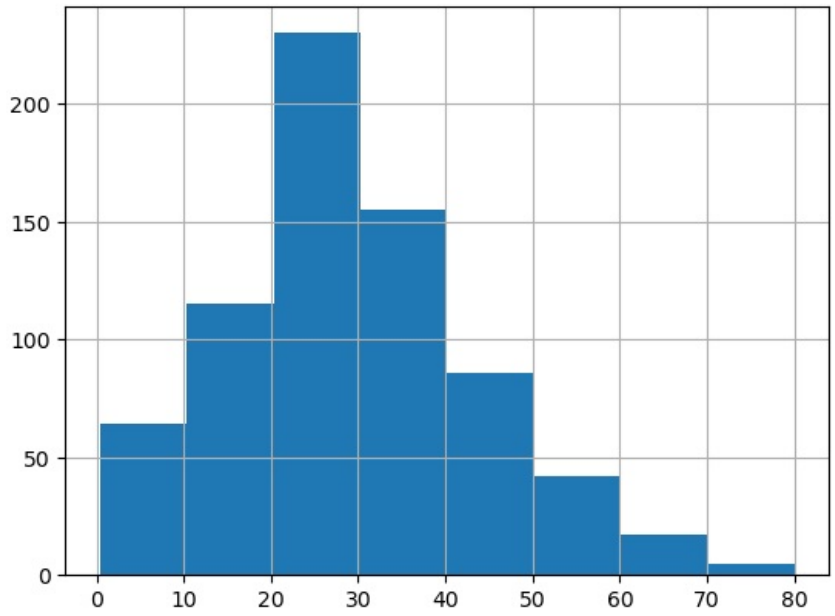
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	66231.000000	-0.626966	-7.561798	138.696504	-16.325843	-0.342697	161.883369
Survived	-0.626966	0.236772	-0.137703	-0.551296	-0.018954	0.032017	6.221787
Pclass	-7.561798	-0.137703	0.699015	-4.496004	0.076599	0.012429	-22.830196
Age	138.696504	-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

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

In [6]:
```
import matplotlib.pyplot as plt
```

In [7]:
```
df['Age'].hist(bins=8)
```

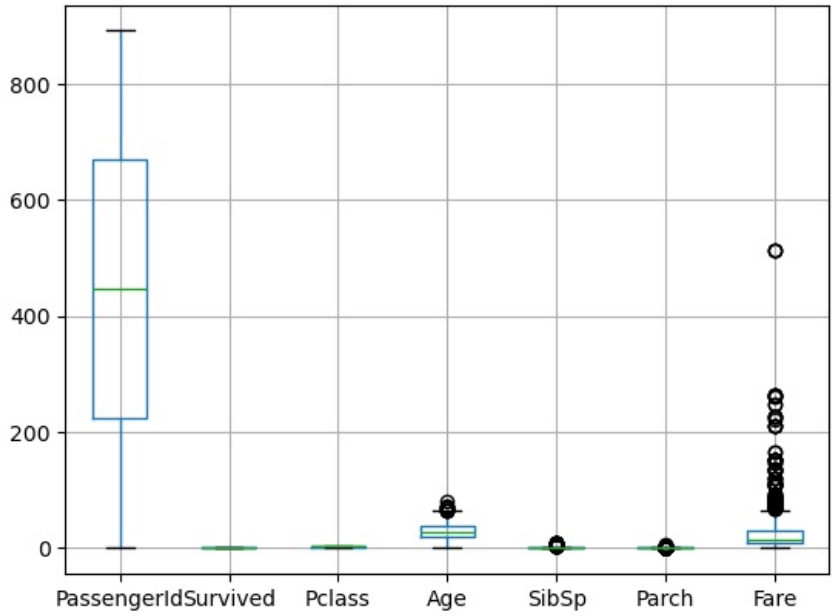
Out[7]: <Axes: >



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

In [8]:
```
df.boxplot()
```

Out[8]: <Axes: >



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

joint distribution.

In [9]:
```
df.plot.scatter(x='Age', y='Fare')
```

Out[9]:
```
<Axes: xlabel='Age', ylabel='Fare'>
```