Darshan UNIVERSITY

योग: कर्मसु कौशलम्

Data Mining

Lab - 3

Mohil Parmar

23010101192

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

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

In [3]:
```
df = pd.read_csv('titanic.csv')
```

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

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
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 [13]:
```
numeric=['PassengerId','Age','SibSp','Parch','Fare']
binary=['Survived','Sex']
ordinal=['Pclass']
nominal=['Name','Ticket','Cabin','Embarked']
```

In [23]:
```
df['Embarked'].unique()
```

Out[23]:
```
array(['S', 'C', 'Q', nan], dtype=object)
```

In [31]:
```
df['Pclass'].unique()
```

Out[31]:
```
array([3, 1, 2], dtype=int64)
```

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

In [43]:
```
sym=df['Sex'].unique()
asym=df['Survived'].unique()
print(sym,asym)
```

```
['male' 'female'] [0 1]
```

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

In [107…

```
numeric=['PassengerId','Age','SibSp','Parch','Fare']
for i in numeric:
    print(i,":")

    print("   mean :" ,df[i].mean())
    print("   Standard Deviation :",df[i].std())
    print("   Minimum :", df[i].min())
    print("   Mode :", df[i].mode()[0])
    print("   Range :", (df[i].max()-df[i].min()))
    print("   Maximum :", df[i].max())
    print("=====")
```

```
PassengerId :
   mean : 446.0
   Standard Deviation : 257.3538420152301
   Minimum : 1
   Mode : 1
   Range : 890
   Maximum : 891
=====
Age :
   mean : 29.69911764705882
   Standard Deviation : 14.526497332334044
   Minimum : 0.42
   Mode : 24.0
   Range : 79.58
   Maximum : 80.0
=====
SibSp :
   mean : 0.5230078563411896
   Standard Deviation : 1.1027434322934275
   Minimum : 0
   Mode : 0
   Range : 8
   Maximum : 8
=====
Parch :
   mean : 0.38159371492704824
   Standard Deviation : 0.8060572211299559
   Minimum : 0
   Mode : 0
   Range : 6
   Maximum : 6
=====
Fare :
   mean : 32.204207968574636
   Standard Deviation : 49.693428597180905
   Minimum : 0.0
   Mode : 8.05
   Range : 512.3292
   Maximum : 512.3292
=====
```

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

In [138…

```
Qualitative = ['Name','Ticket','Cabin','Embarked','Pclass','Survived','Sex']
for i in Qualitative:
    print(df[i].value_counts())
    print("=========")
```

```
Name
Braund, Mr. Owen Harris                     1
Boulos, Mr. Hanna                           1
Frolicher-Stehli, Mr. Maxmillian            1
Gilinski, Mr. Eliezer                       1
Murdlin, Mr. Joseph                         1
                                           ..
Kelly, Miss. Anna Katherine "Annie Kate"    1
McCoy, Mr. Bernard                          1
Johnson, Mr. William Cahoone Jr             1
Keane, Miss. Nora A                         1
Dooley, Mr. Patrick                         1
Name: count, Length: 891, dtype: int64
=========
Ticket
347082      7
CA. 2343    7
1601        7
3101295     6
CA 2144     6
           ..
9234        1
19988       1
2693        1
PC 17612    1
370376      1
Name: count, Length: 681, dtype: int64
=========
Cabin
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: count, Length: 147, dtype: int64
```

```
=========
Embarked
S    644
C    168
Q     77
Name: count, dtype: int64
=========
Pclass
3    491
1    216
2    184
Name: count, dtype: int64
=========
Survived
0    549
1    342
Name: count, dtype: int64
=========
Sex
male      577
female    314
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 [152…

```
# df.describe() only numberic data nu description aapse
# df.describe(include=["object"]) string or object nu description aapse
df.describe(include='all')
```

Out[152...

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
count	891.000000	891.000000	891.000000	891	891	714.000000	891.000000	891.000000	891	891.000000	204	889
unique	NaN	NaN	NaN	891	2	NaN	NaN	NaN	681	NaN	147	3
top	NaN	NaN	NaN	Braund, Mr. Owen Harris	male	NaN	NaN	NaN	347082	NaN	B96 B98	S
freq	NaN	NaN	NaN	1	577	NaN	NaN	NaN	7	NaN	4	644
mean	446.000000	0.383838	2.308642	NaN	NaN	29.699118	0.523008	0.381594	NaN	32.204208	NaN	NaN
std	257.353842	0.486592	0.836071	NaN	NaN	14.526497	1.102743	0.806057	NaN	49.693429	NaN	NaN
min	1.000000	0.000000	1.000000	NaN	NaN	0.420000	0.000000	0.000000	NaN	0.000000	NaN	NaN
25%	223.500000	0.000000	2.000000	NaN	NaN	20.125000	0.000000	0.000000	NaN	7.910400	NaN	NaN
50%	446.000000	0.000000	3.000000	NaN	NaN	28.000000	0.000000	0.000000	NaN	14.454200	NaN	NaN
75%	668.500000	1.000000	3.000000	NaN	NaN	38.000000	1.000000	0.000000	NaN	31.000000	NaN	NaN
max	891.000000	1.000000	3.000000	NaN	NaN	80.000000	8.000000	6.000000	NaN	512.329200	NaN	NaN

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

In [178...

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

Out[178...

	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 [176...

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

Out[176...

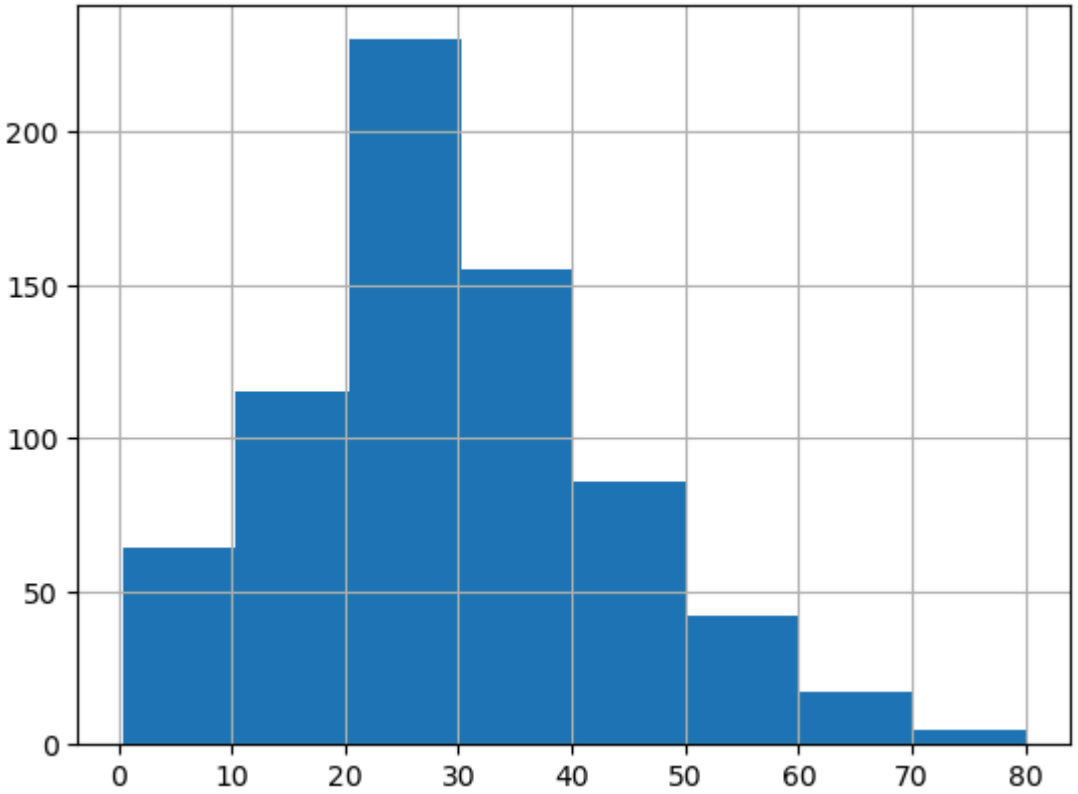
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500
Age	0.036847	-0.077221	-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

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

In [185...

```
import matplotlib.pyplot as plt
plt.hist(df['Age'],bins=8)
```

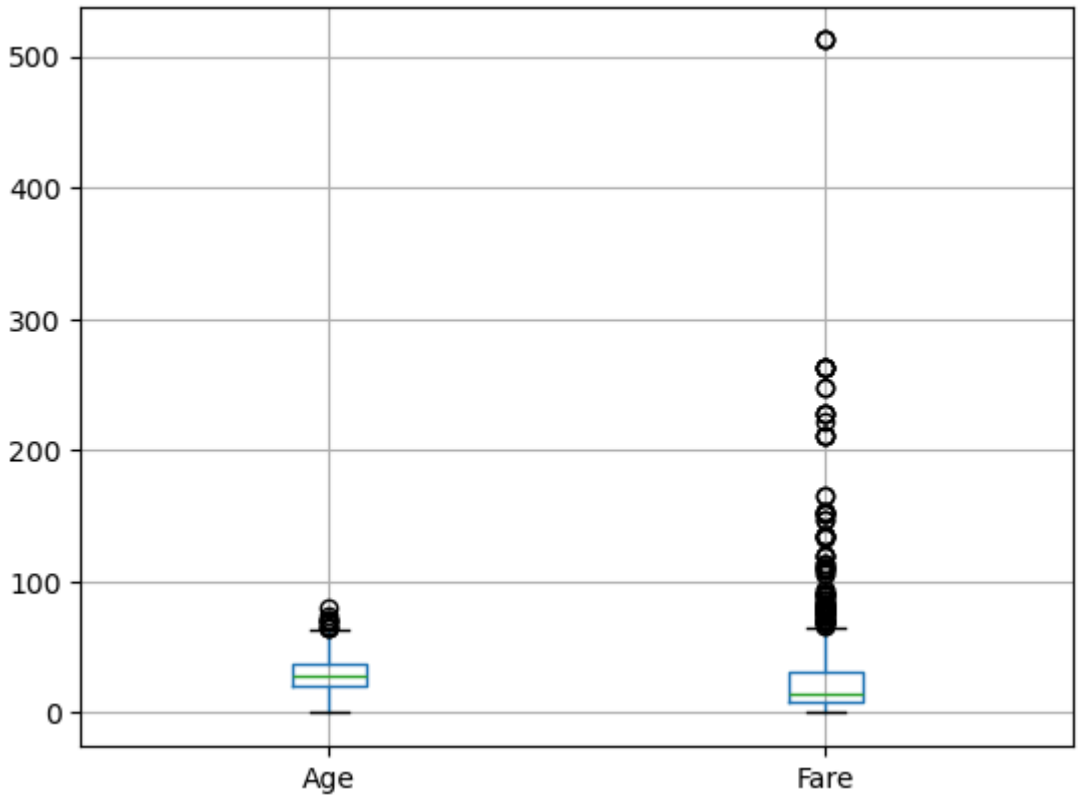
```
plt.grid(True)
plt.show()
```



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

In [197…
```
df[['Age','Fare']].boxplot()
```

Out[197…
```
<Axes: >
```



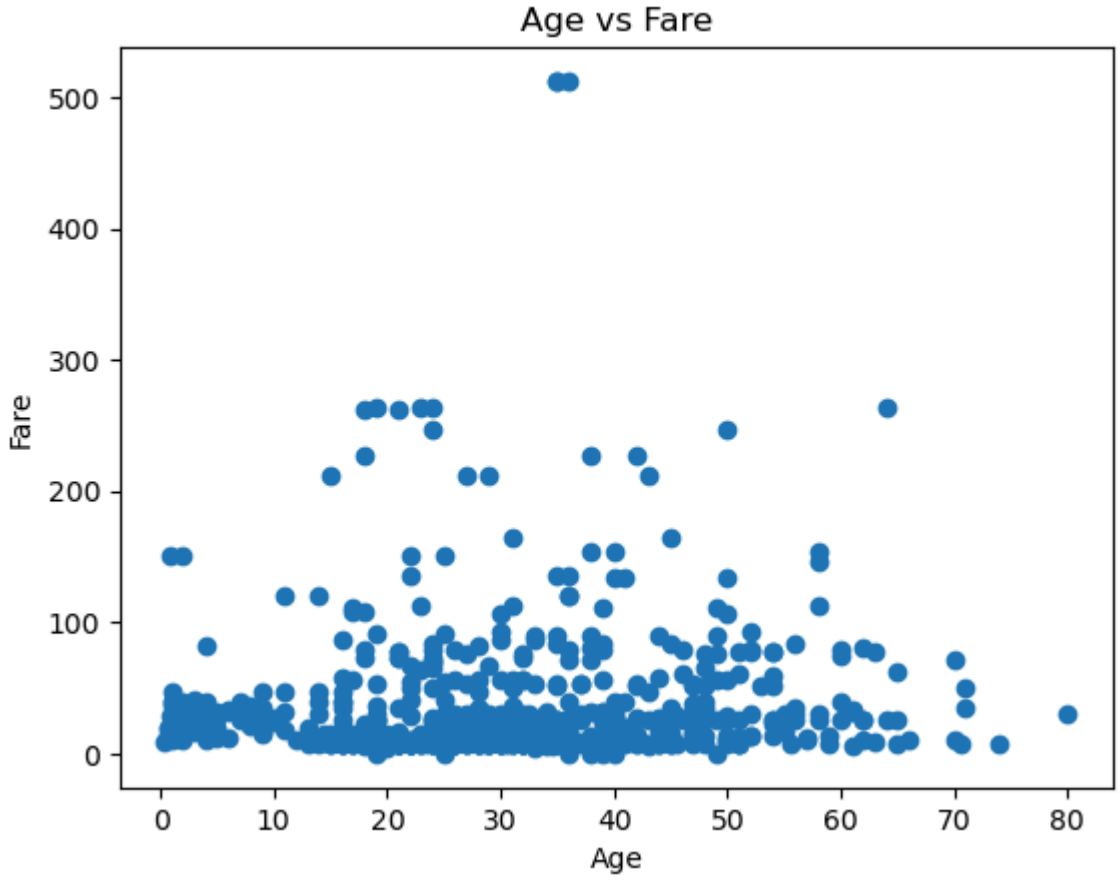
11) Display scatter plot for any 5 pair of attributes , we can use a scatter plot to visualize their joint distribution.

In [201…

```
plt.scatter(df['Age'] , df['Fare'])
plt.xlabel('Age')
plt.ylabel('Fare')
plt.title('Age vs Fare')
```

Out[201…

```
Text(0.5, 1.0, 'Age vs Fare')
```


Age vs Fare
Fare
Age
0
100
200
300
400
500
0
10
20
30
40
50
60
70
80