

Data Mining

Lab - 3

Enrollment no.:- 22010101478

Name:- Jay Ramani

Batch:- A-3

Roll no.:- 156

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')
df
```

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
...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

In [4]:

```
df.head()
```

Out[4]:

	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 [22]:

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

```
print("Nominal", nominal)
print("Ordinal", ordinal)
print("Binary", binary)
print("Numeric", numeric)
```

Nominal ['PassengerId', 'Cabin', 'Embarked', 'Name', 'Ticket']
Ordinal ['Pclass']
Binary ['Survived', 'Sex']
Numeric ['Fare', 'Age', 'SibSp', 'Parch']

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

```
In [20]: print("Symmetric")
df['Sex']
```

Symmetric

```
Out[20]: 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 [21]: print('Asymmetric')
df['Survived']
```

Asymmetric

```
Out[21]: 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 [41]: from pandas.api.types import is_numeric_dtype

for col in df.columns:
    if is_numeric_dtype(df[col].dtype):
        print(col, ":")
        print("\tMean = ", df[col].mean())
        print("\tStandard deviation = ", df[col].std())
        print("\tMinimum = ", df[col].min())
        print("\tMaximum = ", df[col].max())
        print("\tMode = ", df[col].mode()[0])
        print("\tRange = ", df[col].max() - df[col].min())
```

PassengerId :
Mean = 446.0
Standard deviation = 257.3538420152301
Minimum = 1
Maximum = 891
Mode = 1
Range = 890

Survived :
Mean = 0.3838383838383838
Standard deviation = 0.4865924542648575
Minimum = 0
Maximum = 1
Mode = 0
Range = 1

Pclass :
Mean = 2.308641975308642
Standard deviation = 0.836071240977049
Minimum = 1
Maximum = 3
Mode = 3
Range = 2

Age :
Mean = 29.69911764705882
Standard deviation = 14.526497332334042
Minimum = 0.42
Maximum = 80.0
Mode = 24.0
Range = 79.58

SibSp :
Mean = 0.5230078563411896
Standard deviation = 1.1027434322934317
Minimum = 0
Maximum = 8
Mode = 0
Range = 8

Parch :
Mean = 0.38159371492704824
Standard deviation = 0.8060572211299483
Minimum = 0
Maximum = 6
Mode = 0
Range = 6

Fare :
Mean = 32.204207968574636
Standard deviation = 49.6934285971809
Minimum = 0.0
Maximum = 512.3292
Mode = 8.05
Range = 512.3292

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

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')
```

	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 [46]: 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
```

```
In [47]: df[['PassengerId', 'Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare']].cov()
```

Out[47]:

	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 [48]: df[['PassengerId', 'Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare']].corr()
```

Out[48]:

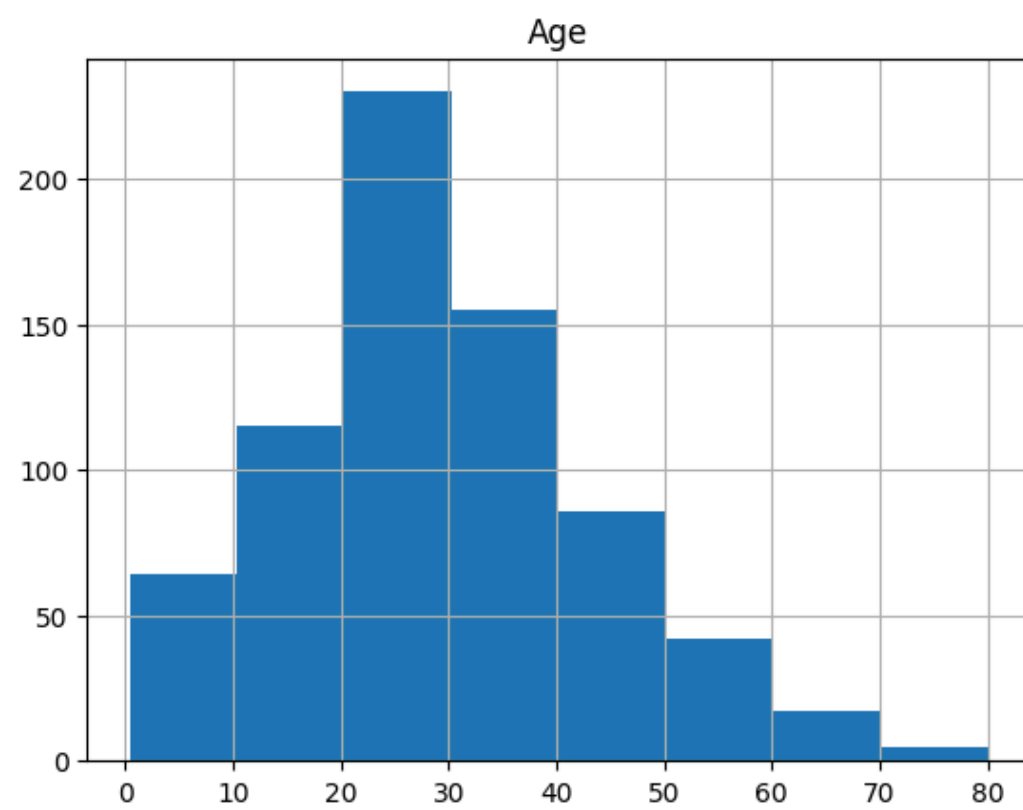
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500
Age	0.036847	-0.077221	-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 [49]: import matplotlib.pyplot as plt
```

```
In [50]: df.hist(column='Age', bins=8)
```

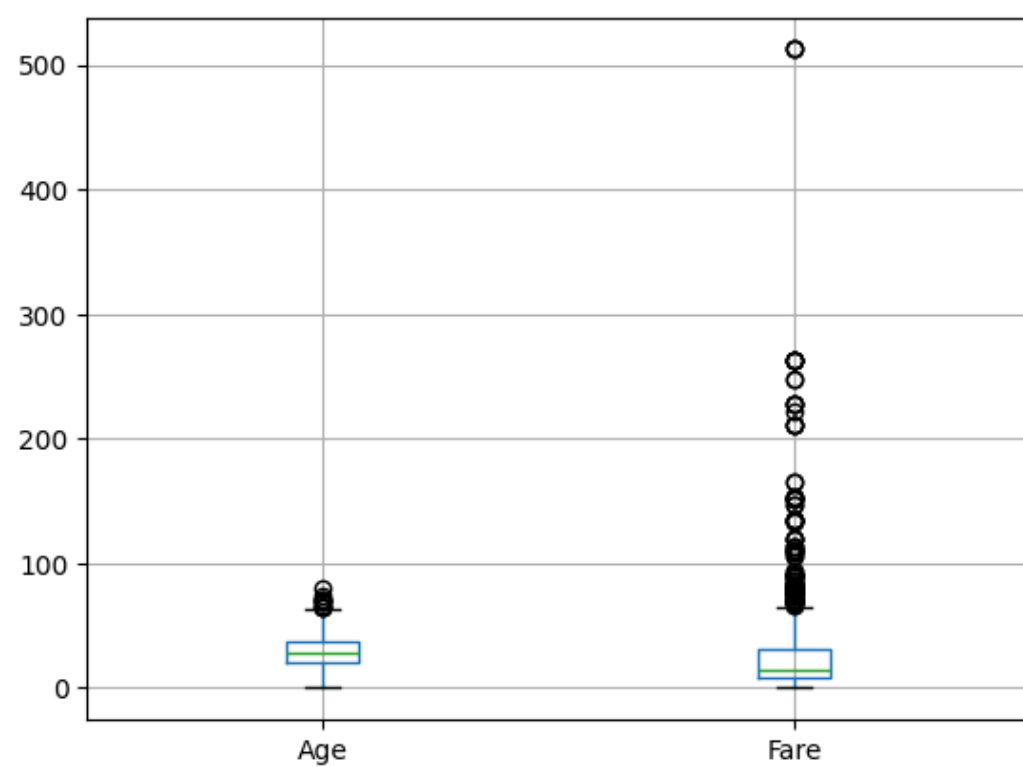
```
Out[50]: array([[<Axes: title={'center': 'Age'}>]], dtype=object)
```



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

```
In [51]: df.boxplot(column=['Age', 'Fare'])
```

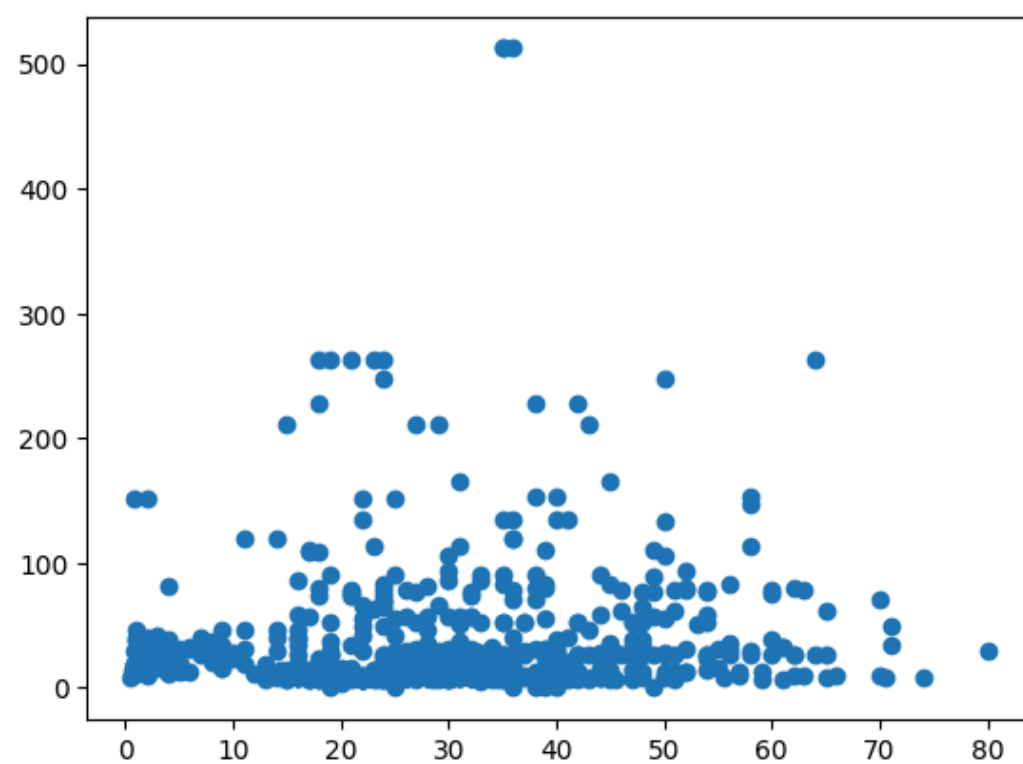
Out[51]: <Axes: >



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

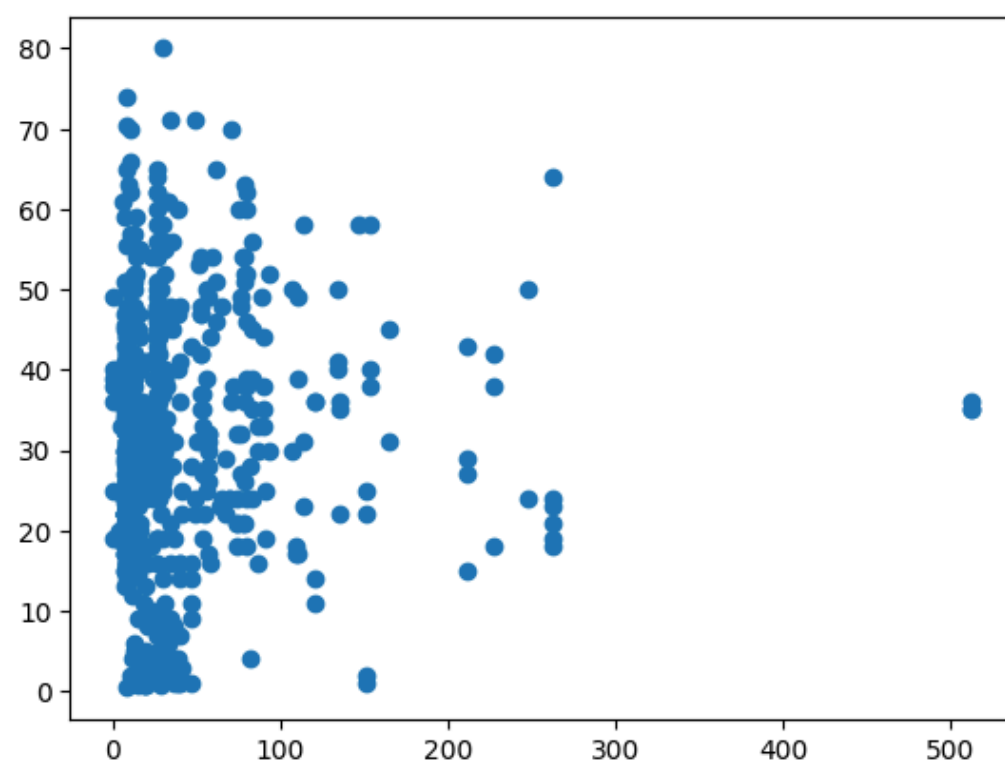
```
In [54]: plt.scatter(df['Age'], df['Fare'])
```

Out[54]: <matplotlib.collections.PathCollection at 0x2623d7f0810>



```
In [61]: plt.scatter(df['Fare'], df['Age'])
```

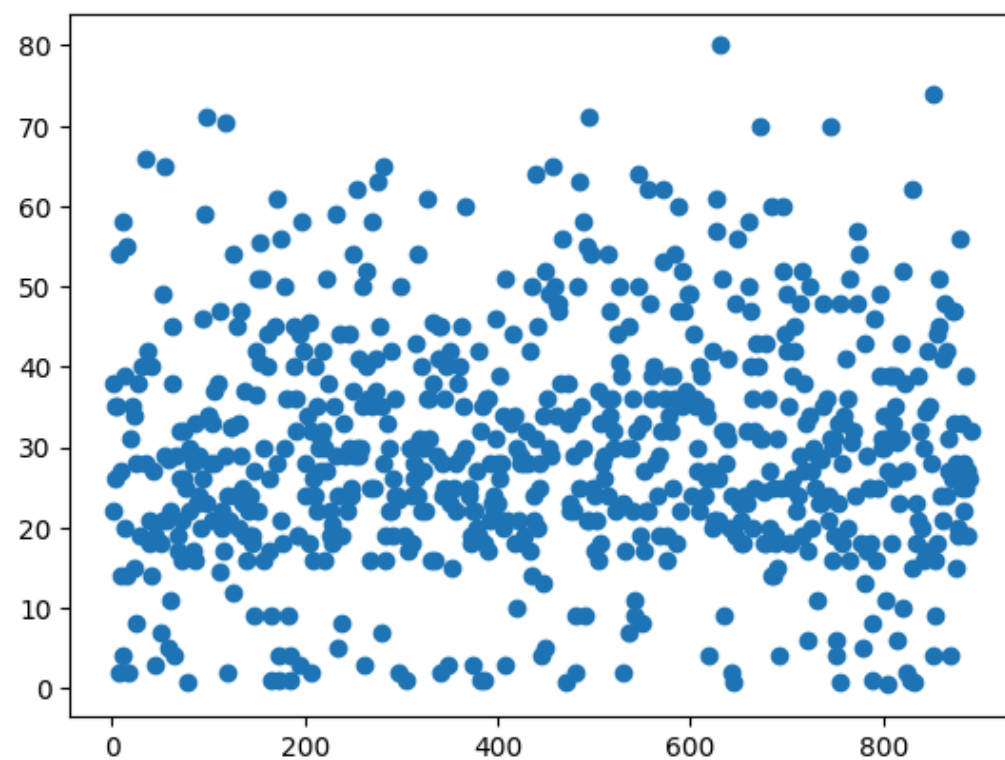
```
Out[61]: <matplotlib.collections.PathCollection at 0x2623f4fb410>
```



```
In [ ]: df[['PassengerId', 'Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare']].corr()
```

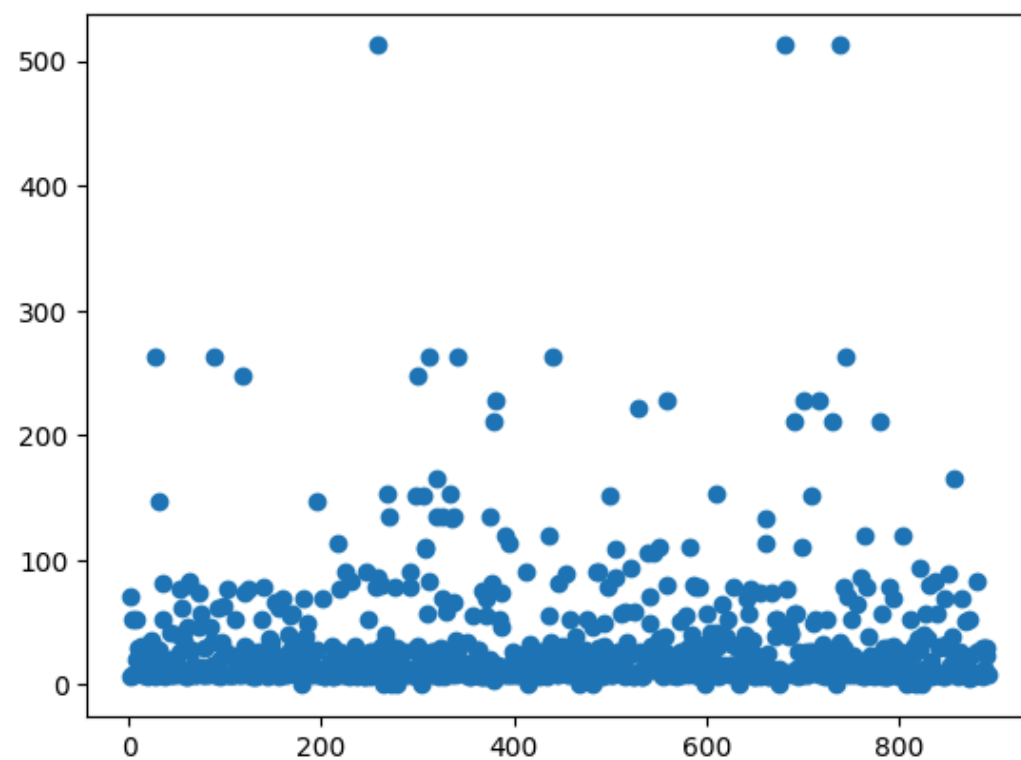
```
In [71]: plt.scatter(df['PassengerId'], df['Age'])
```

```
Out[71]: <matplotlib.collections.PathCollection at 0x262409cb510>
```



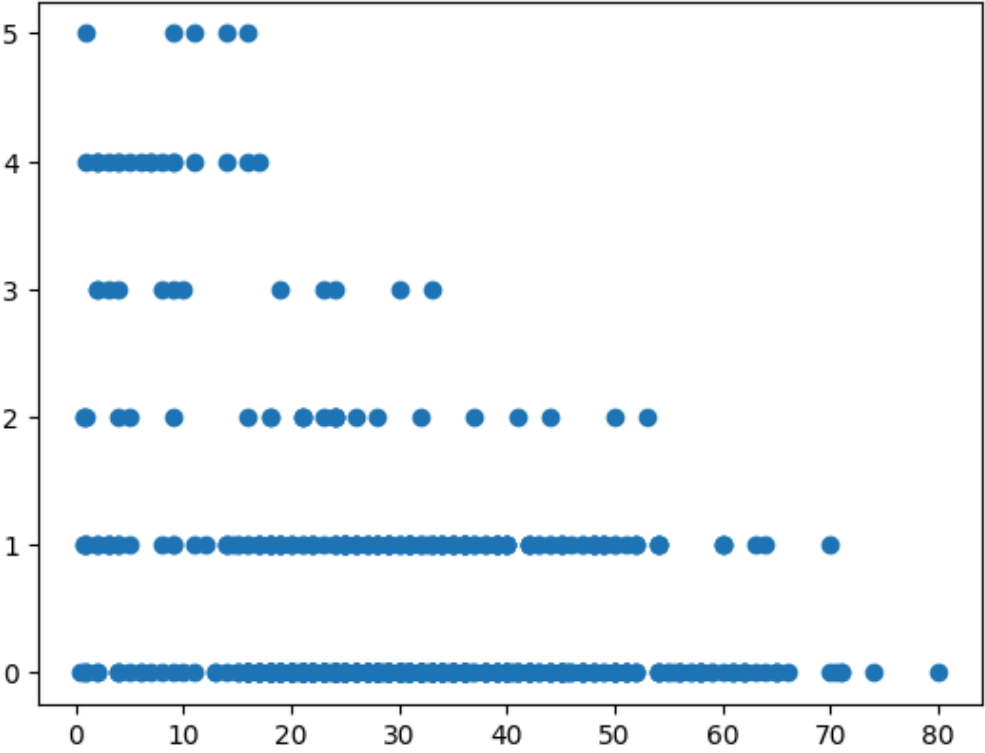
```
In [72]: plt.scatter(df['PassengerId'], df['Fare'])
```

```
Out[72]: <matplotlib.collections.PathCollection at 0x262409cb2d0>
```



```
In [77]: plt.scatter(df['Age'], df['SibSp'])
```

Out[77]: <matplotlib.collections.PathCollection at 0x2624405df10>



In []: