Machine Learning (Lab 2)

Practical 2: Explain the following concepts of Machine learning using python.

In [20]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn  as sns
%matplotlib inline
```

1) Find out Mean, Median, Mode and standard deviation.

In [4]:
```
df = pd.read_csv("pima-indians-diabetes.csv", names=['preg', 'plas', 'pres','skin', 'test', 'mass', 'pedi', 'age', 'class'])
```

In [6]:
```
df
```

Out[6]:

	preg	plas	pres	skin	test	mass	pedi	age	class
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

First five column

In [7]:
```
df.head()
```

Out[7]:

	preg	plas	pres	skin	test	mass	pedi	age	class
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Last five column

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

Out[8]:

	preg	plas	pres	skin	test	mass	pedi	age	class
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

Mean

In [9]:
```
df.mean()
```

Out[9]:
```
preg      3.845052
plas    120.894531
pres     69.105469
skin     20.536458
test     79.799479
mass     31.992578
pedi      0.471876
age      33.240885
class     0.348958
dtype: float64
```

Median

In [10]:
```
df.median()
```

Out[10]:
```
preg      3.0000
plas    117.0000
pres     72.0000
skin     23.0000
test     30.5000
mass     32.0000
pedi      0.3725
```

```
age      29.0000
class     0.0000
dtype: float64
```

Mode

In [11]:
```
df.mode()
```

Out[11]:

	preg	plas	pres	skin	test	mass	pedi	age	class
0	1.0	99	70.0	0.0	0.0	32.0	0.254	22.0	0.0
1	NaN	100	NaN	NaN	NaN	NaN	0.258	NaN	NaN

Standard Deviation

In [12]:
```
df.std()
```

Out[12]:
```
preg      3.369578
plas     31.972618
pres     19.355807
skin     15.952218
test    115.244002
mass      7.884160
pedi      0.331329
age      11.760232
class     0.476951
dtype: float64
```

Informarion

In [15]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   preg    768 non-null    int64  
 1   plas    768 non-null    int64  
 2   pres    768 non-null    int64  
 3   skin    768 non-null    int64  
 4   test    768 non-null    int64  
 5   mass    768 non-null    float64
 6   pedi    768 non-null    float64
 7   age     768 non-null    int64  
 8   class   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

Description

In [17]:

```
df.describe()
```

Out[17]:

	preg	plas	pres	skin	test	mass	pedi	age
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000

2)Plotting of Numeric data using Box Plot and Histogram

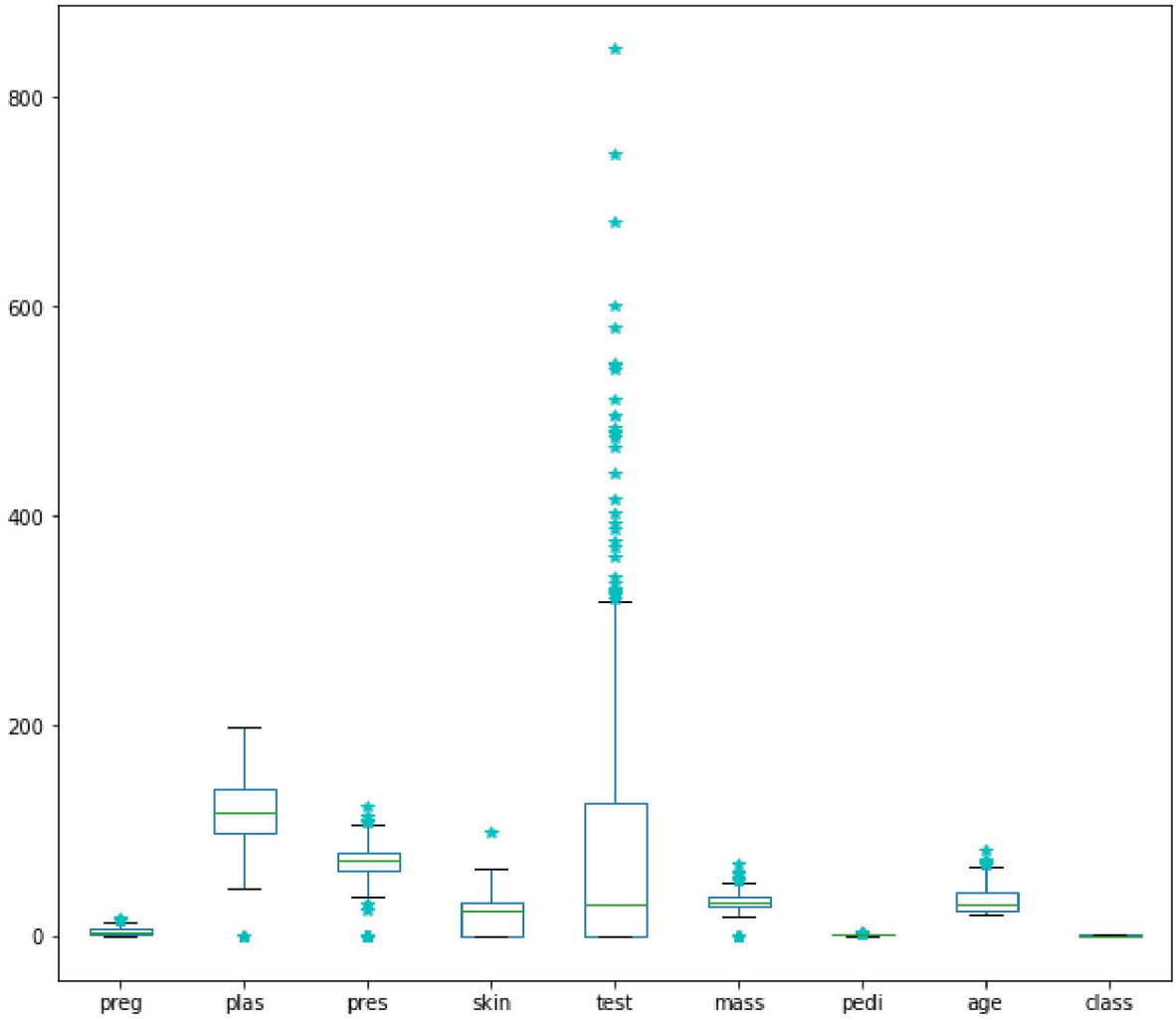
Box plot

In [32]:

```
df.plot(kind='box', figsize=(10,9), sym='c*')
```

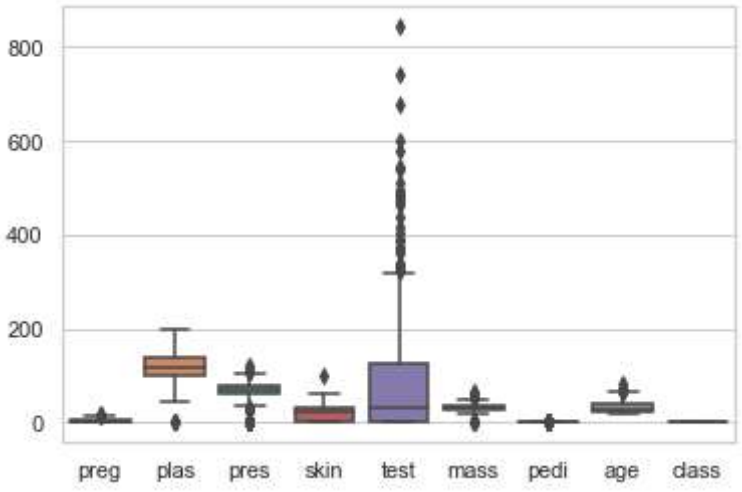
Out[32]:

```
<AxesSubplot:>
```



In [41]:

```
sns.set_theme(style="whitegrid")
ax = sns.boxplot( data=df)
```

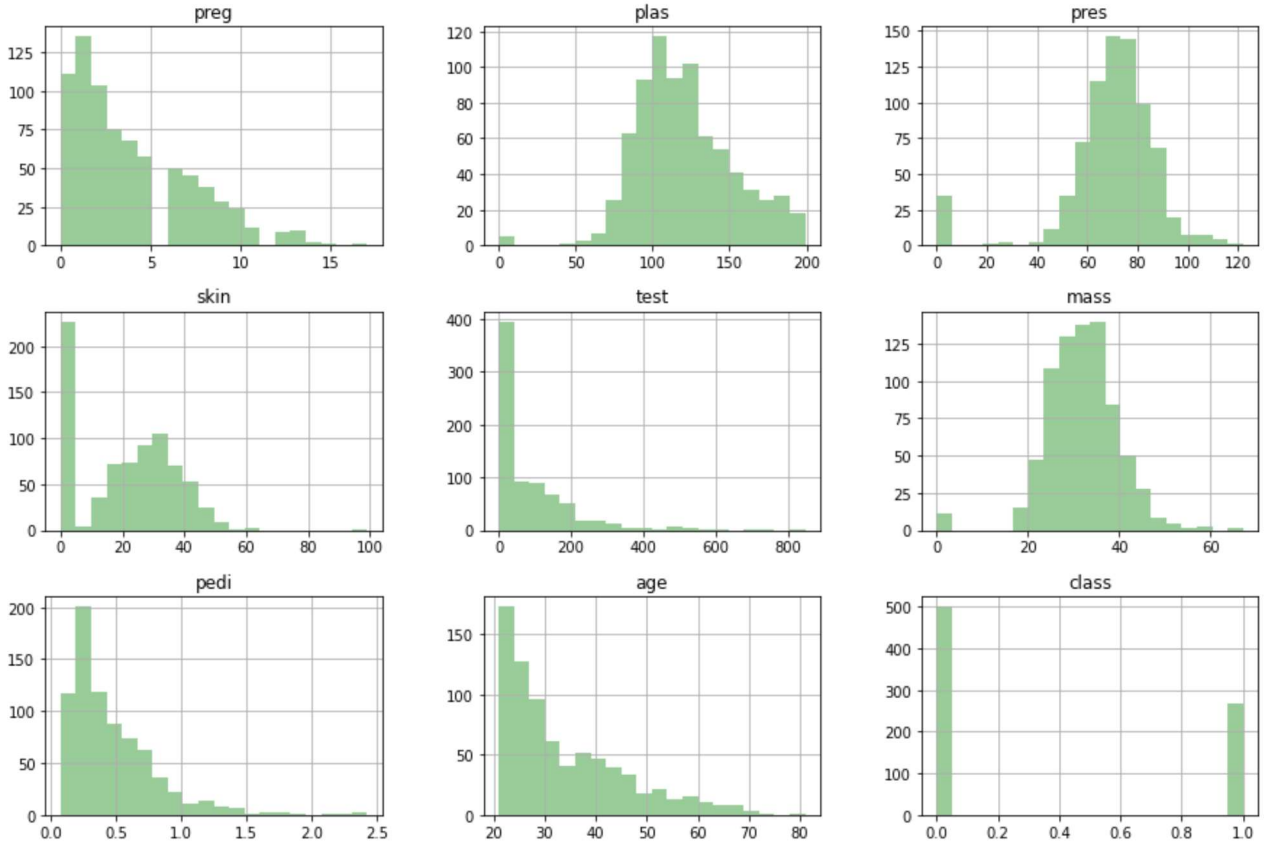


Histogram

In [31]:

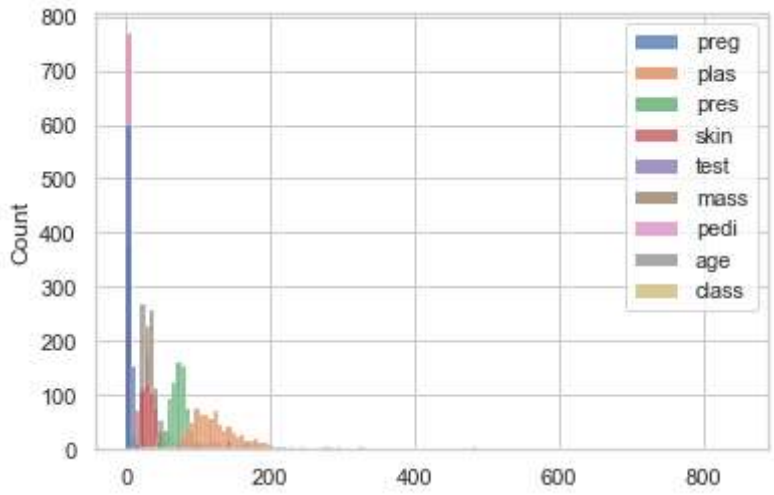
```
df.hist(bins=20, figsize=(15,10),facecolor='g',alpha=.4)
```

Out[31]:
```
array([[<AxesSubplot:title={'center':'preg'}>,
        <AxesSubplot:title={'center':'plas'}>,
        <AxesSubplot:title={'center':'pres'}>],
       [<AxesSubplot:title={'center':'skin'}>,
        <AxesSubplot:title={'center':'test'}>,
        <AxesSubplot:title={'center':'mass'}>],
       [<AxesSubplot:title={'center':'pedi'}>,
        <AxesSubplot:title={'center':'age'}>,
        <AxesSubplot:title={'center':'class'}>]], dtype=object)
```



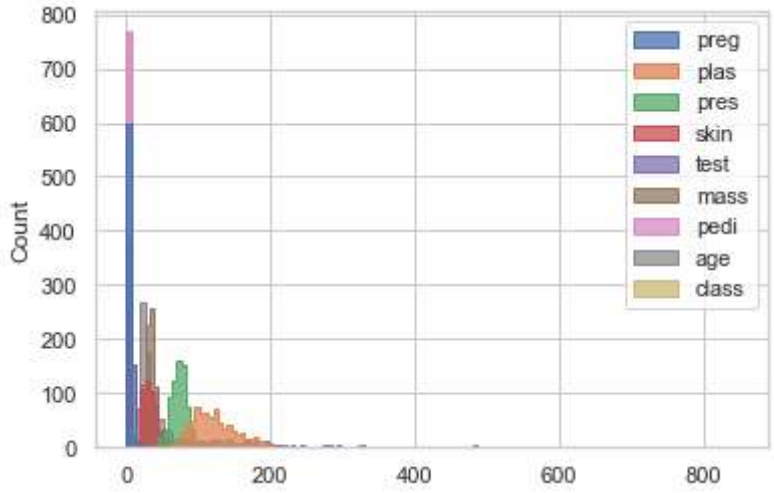
In [42]:
```
sns.histplot(data=df)
```

Out[42]:
```
<AxesSubplot:ylabel='Count'>
```



In [44]:
```
sns.histplot(data=df,element="step")
```

Out[44]: <AxesSubplot:ylabel='Count'>



In [48]:
```
sns.set_theme(style="ticks")
sns.pairplot(df,hue='class')
```

Out[48]: <seaborn.axisgrid.PairGrid at 0x203a3b35048>

3) Exploring relationship between variables by Scatter Plot(use “home1.csv”) and Cross tabulation

In [47]:

```
home1 = pd.read_csv("home1.csv", names=['rooms','rent','sqft'])
home1.head()
```

Out[47]:

	rooms	rent	sqft
0	2	2000	1000
1	3	3000	1200
2	4	4000	1500
3	5	5000	1700
4	1	1000	500

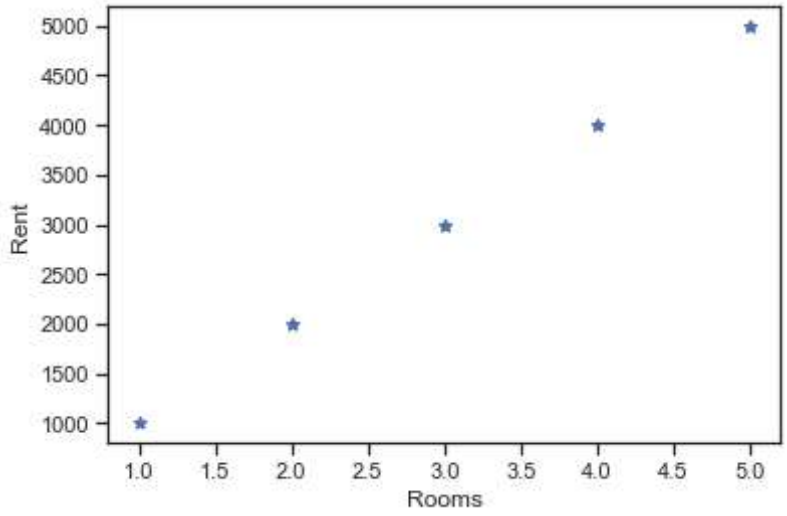
In [50]:
```
home2 = pd.read_csv("home2.csv")
home2.head()
```

Out[50]:

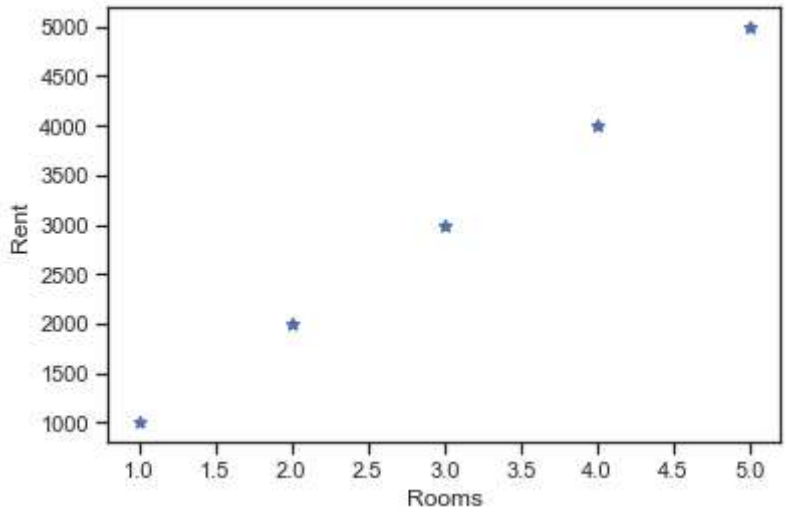
	rooms	rent	sqft
0	2	2000	1000
1	3	3000	1200
2	4	4000	1500
3	5	5000	1700
4	1	1000	500

Scatter plot

In [54]:
```
plt.scatter(home1.rooms, home1.rent, marker='*')
plt.xlabel("Rooms")
plt.ylabel("Rent")
plt.show()
```

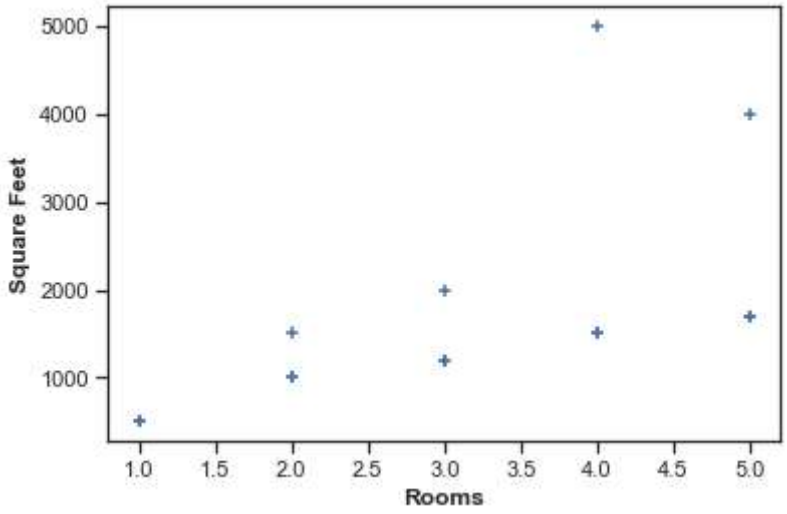


In [56]:
```
plt.scatter(home2.rooms, home2.rent, marker='*')
plt.xlabel("Rooms")
plt.ylabel("Rent")
plt.show()
```



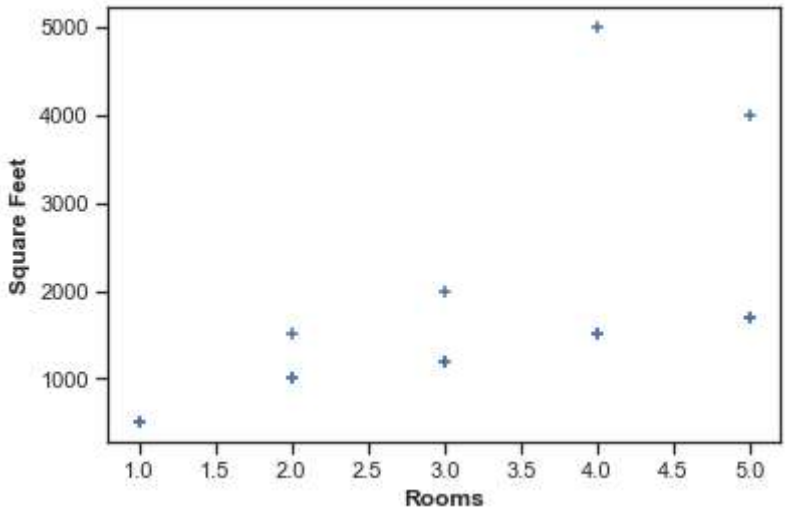
In [59]:

```
plt.scatter(home1.rooms, home1.sqft, marker='+')
plt.xlabel("Rooms", fontweight='bold')
plt.ylabel("Square Feet", fontweight='bold')
plt.show()
```



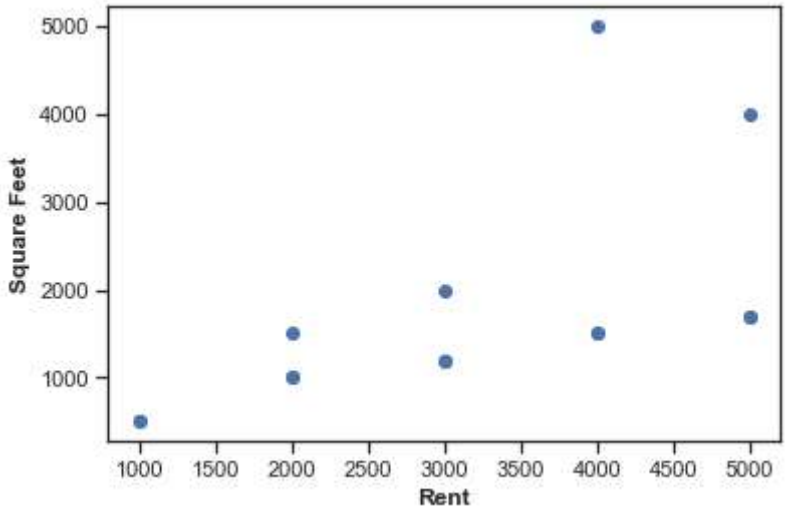
In [60]:

```
plt.scatter(home2.rooms, home2.sqft, marker='+')
plt.xlabel("Rooms", fontweight='bold')
plt.ylabel("Square Feet", fontweight='bold')
plt.show()
```



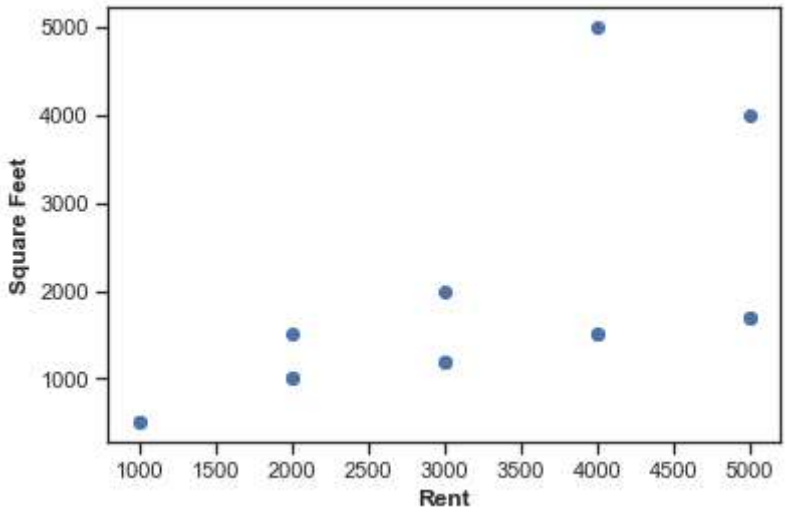
In [63]:

```
plt.scatter(home1.rent, home1.sqft, marker='o')
plt.xlabel("Rent", fontweight='bold')
plt.ylabel("Square Feet", fontweight='bold')
plt.show()
```



In [64]:

```
plt.scatter(home2.rent, home2.sqft, marker='o')
plt.xlabel("Rent", fontweight='bold')
plt.ylabel("Square Feet", fontweight='bold')
plt.show()
```



Cross Tablulation

In [72]:
```
pd.crosstab(home1.rooms, home1.rent)
```

Out[72]:

rooms	1000	2000	3000	4000	5000
1	2	0	0	0	0
2	0	3	0	0	0
3	0	0	3	0	0
4	0	0	0	3	0
5	0	0	0	0	3

In [73]:
```
pd.crosstab(home2.rooms, home2.rent)
```

Out[73]:

rooms	1000	2000	3000	4000	5000
1	2	0	0	0	0
2	0	3	0	0	0
3	0	0	3	0	0
4	0	0	0	3	0
5	0	0	0	0	3

In [68]:
```
pd.crosstab(home1.rooms, home1.sqft)
```

Out[68]:

rooms	500	1000	1200	1500	1700	2000	4000	5000

sqft	500	1000	1200	1500	1700	2000	4000	5000
rooms								
1	2	0	0	0	0	0	0	0
2	0	2	0	1	0	0	0	0
3	0	0	2	0	0	1	0	0
4	0	0	0	2	0	0	0	1
5	0	0	0	0	2	0	1	0

In [69]:
```
pd.crosstab(home2.rooms, home2.sqft)
```

Out[69]:

sqft	500	1000	1200	1500	1700	2000	4000	5000
rooms								
1	2	0	0	0	0	0	0	0
2	0	2	0	1	0	0	0	0
3	0	0	2	0	0	1	0	0
4	0	0	0	2	0	0	0	1
5	0	0	0	0	2	0	1	0

In [70]:
```
pd.crosstab(home1.rent, home1.sqft)
```

Out[70]:

sqft	500	1000	1200	1500	1700	2000	4000	5000
rent								
1000	2	0	0	0	0	0	0	0
2000	0	2	0	1	0	0	0	0
3000	0	0	2	0	0	1	0	0
4000	0	0	0	2	0	0	0	1
5000	0	0	0	0	2	0	1	0

In [71]:
```
pd.crosstab(home2.rent, home2.sqft)
```

Out[71]:

sqft	500	1000	1200	1500	1700	2000	4000	5000
rent								
1000	2	0	0	0	0	0	0	0
2000	0	2	0	1	0	0	0	0
3000	0	0	2	0	0	1	0	0
4000	0	0	0	2	0	0	0	1

sqft rent	500	1000	1200	1500	1700	2000	4000	5000
5000	0	0	0	0	2	0	1	0