Code Written By: Ankit Kotnala

Import Necessary Libraries

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Reading CSV File

In [2]:
```
# Reading the dataset
df=pd.read_csv("Advertising.csv")
```

In [3]:
```
# return top five coloumns.
df.head()
```

Out[3]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9

In [4]:
```
# Return bottom 5 coloumns.
df.tail()
```

Out[4]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
195	196	38.2	3.7	13.8	7.6
196	197	94.2	4.9	8.1	9.7
197	198	177.0	9.3	6.4	12.8
198	199	283.6	42.0	66.2	25.5
199	200	232.1	8.6	8.7	13.4

In [5]:
```
#returns tuple of shape (Rows, columns) of dataframe
df.shape
```

Out[5]:
```
(200, 5)
```

In [6]:
```
# prints statistical analysis about the dataframe
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Unnamed: 0  200 non-null    int64
```

```
 1   TV           200 non-null    float64
 2   Radio        200 non-null    float64
 3   Newspaper    200 non-null    float64
 4   Sales        200 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 7.9 KB
```

In [7]:
```
#returns numerical description of the data in the dataframe
df.describe()
```

Out[7]:

	Unnamed: 0	TV	Radio	Newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000	200.000000
mean	100.500000	147.042500	23.264000	30.554000	14.022500
std	57.879185	85.854236	14.846809	21.778621	5.217457
min	1.000000	0.700000	0.000000	0.300000	1.600000
25%	50.750000	74.375000	9.975000	12.750000	10.375000
50%	100.500000	149.750000	22.900000	25.750000	12.900000
75%	150.250000	218.825000	36.525000	45.100000	17.400000
max	200.000000	296.400000	49.600000	114.000000	27.000000

OBSERVATION:

Avg expense spend is highest on tv.

Avg expense spend is lowest on radio.

Max sale is 27 and min is 1.6

Droping the Column

In [8]:
```
#dropping the column 'Unnamed: 0'
df=df.drop(columns=["Unnamed: 0"])
```

In [9]:
```
#Return dataframe
df
```

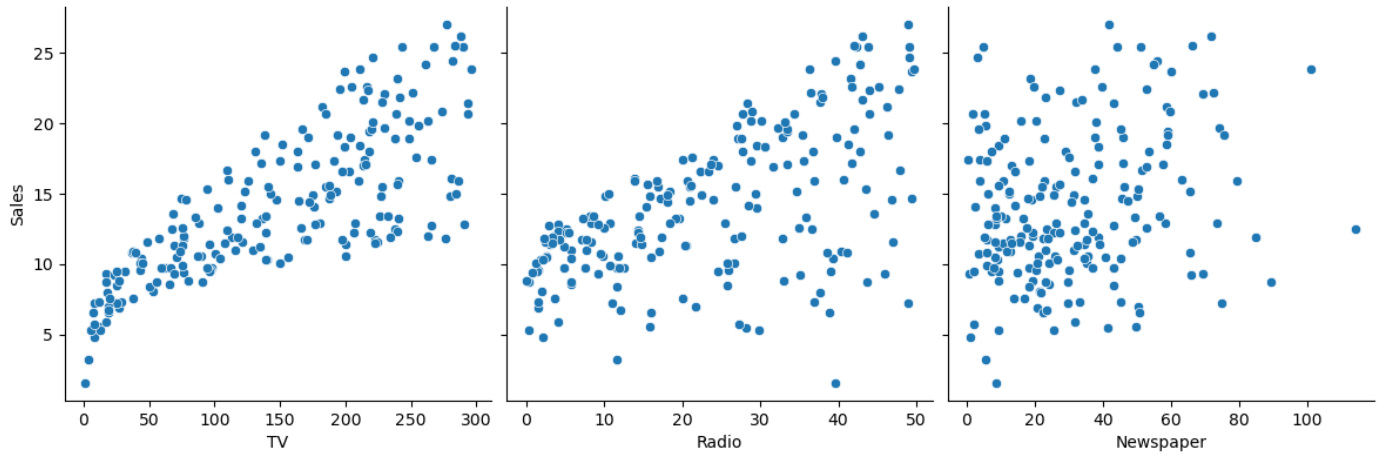
Out[9]:

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	9.7

197	177.0	9.3	6.4	12.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	13.4

200 rows × 4 columns

In [10]:
```
sns.pairplot(df, x_vars=['TV', 'Radio', 'Newspaper'], y_vars='Sales', kind='scatter', he
plt.show()
```

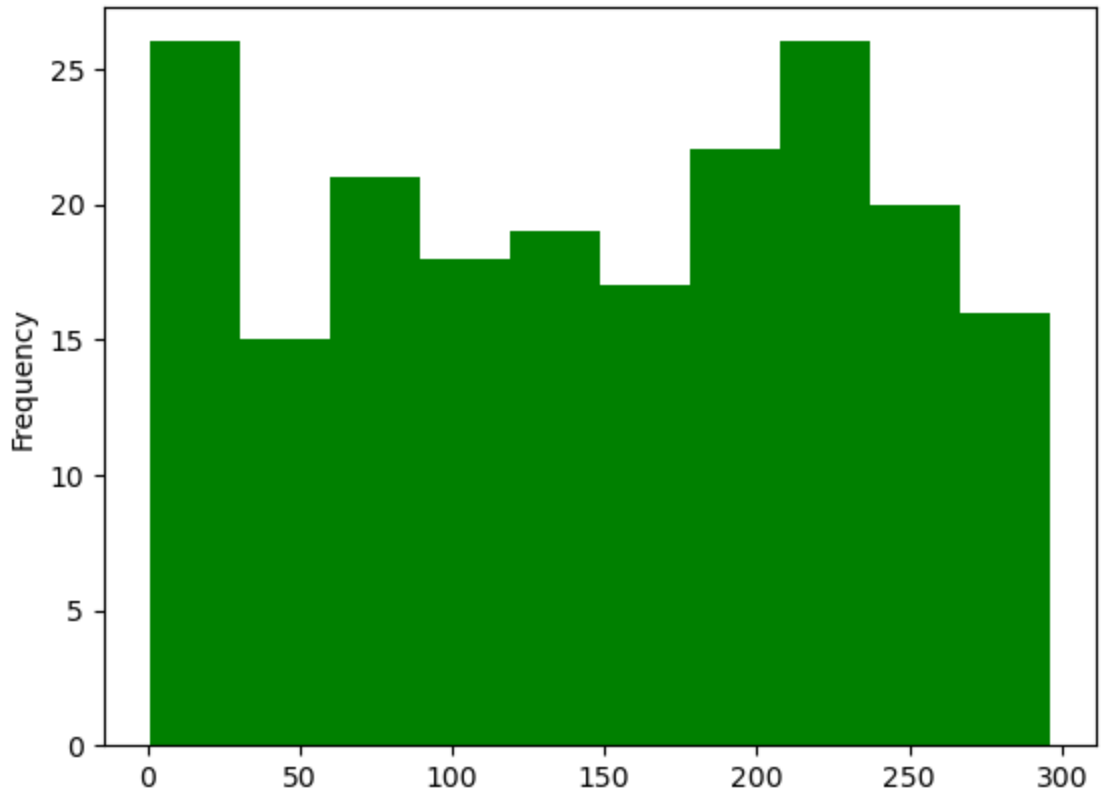


observation:

when advertising cost increases in tv ads the sale will increase as well while the newspaper and radio it's bit unpredictable.

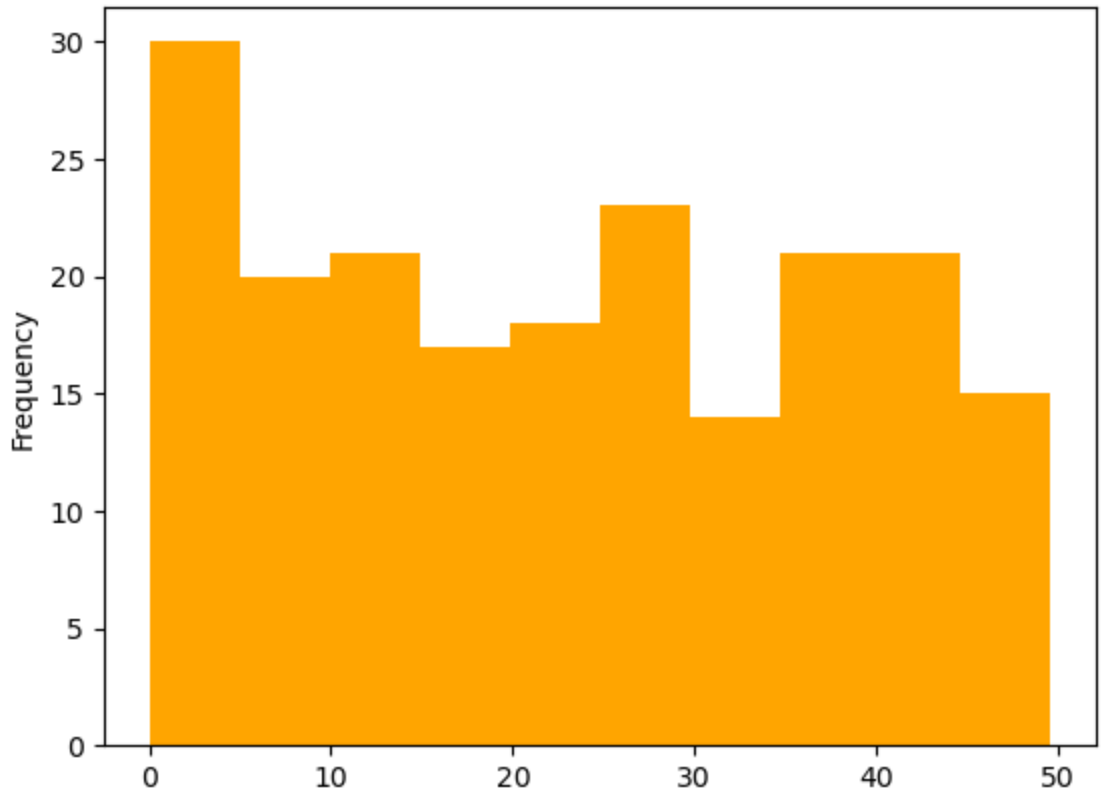
In [11]:
```
df['TV'].plot.hist(bins=10,color="green",xlabel="tv")
```

Out[11]:
```
<Axes: ylabel='Frequency'>
```



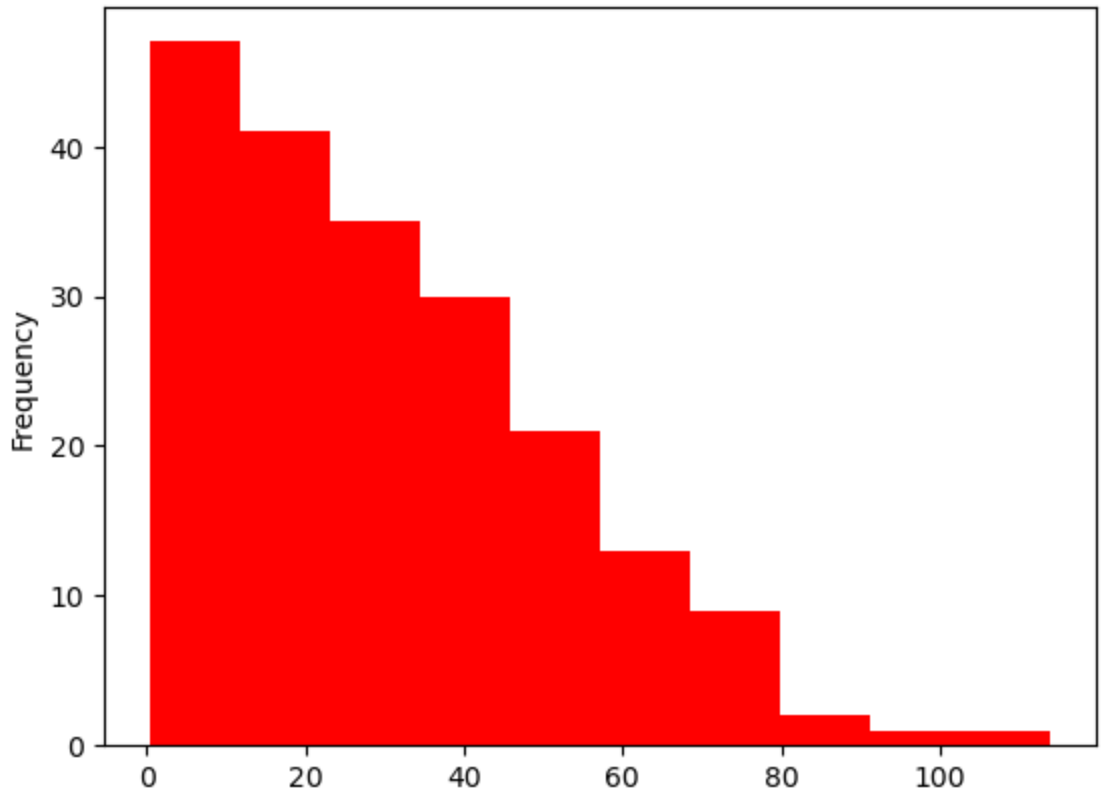
In [12]:
```
df['Radio'].plot.hist(bins=10,color="orange",xlabel="Radio")
```

Out[12]:
```
<Axes: ylabel='Frequency'>
```



In [13]:
```
df['Newspaper'].plot.hist(bins=10,color="Red",xlabel="Newspaper")
```

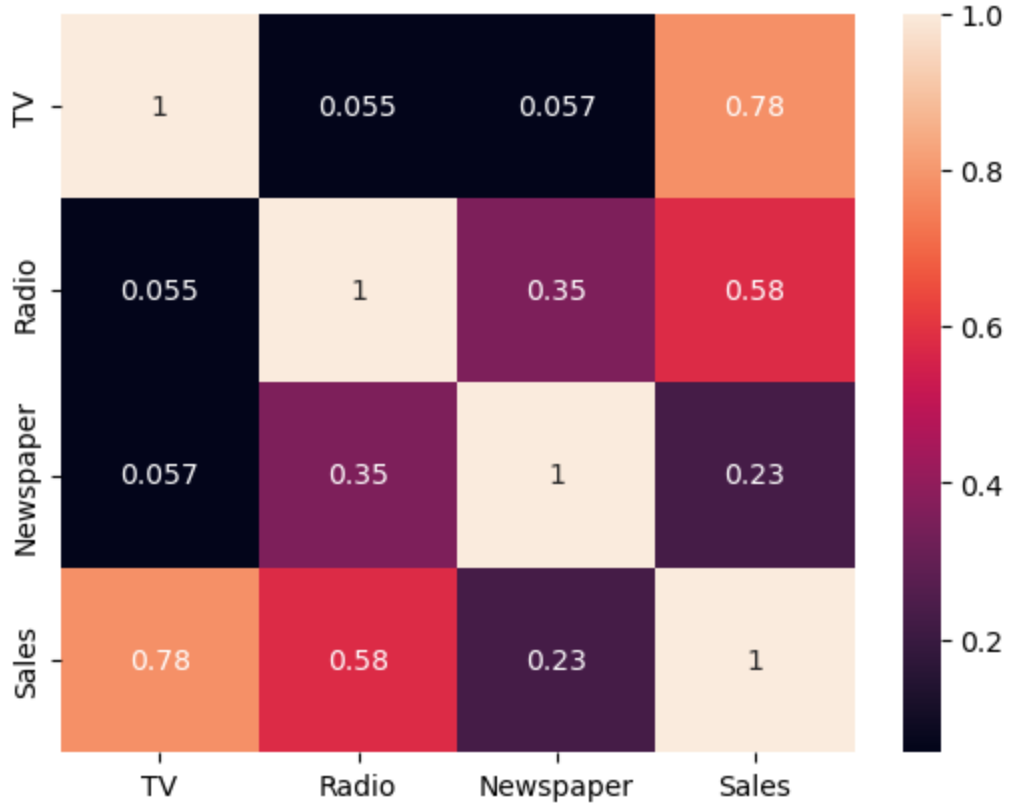
Out[13]:
```
<Axes: ylabel='Frequency'>
```



observation:

The majority sales in the result of low advertising cost in newspaper. hence, tv ads are dominating.

In [14]:
```
sns.heatmap(df.corr(),annot= True)
plt.show()
```



observation:

Sales is highly coordinated with tv.

In [15]:
```
x=df.iloc[:, 0:-1]
```

In [16]:
```
x
```

Out[16]:

	TV	Radio	Newspaper
0	230.1	37.8	69.2
1	44.5	39.3	45.1
2	17.2	45.9	69.3
3	151.5	41.3	58.5
4	180.8	10.8	58.4
...	...	...	...
195	38.2	3.7	13.8
196	94.2	4.9	8.1
197	177.0	9.3	6.4
198	283.6	42.0	66.2
199	232.1	8.6	8.7

200 rows × 3 columns

In [17]:
```
y=df.iloc[:,-1]
```

In [18]:
```
y
```

Out[18]:
```
0      22.1
1      10.4
2       9.3
3      18.5
4      12.9
       ...
195     7.6
196     9.7
197    12.8
198    25.5
199    13.4
Name: Sales, Length: 200, dtype: float64
```

Train Test Split

In [19]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=43)
```

In [20]:
```
x_train
```

Out[20]:

	TV	Radio	Newspaper
116	139.2	14.3	25.6
138	43.0	25.9	20.5
155	4.1	11.6	5.7
82	75.3	20.3	32.5
160	172.5	18.1	30.7
...	...	...	...
58	210.8	49.6	37.7
21	237.4	5.1	23.5
49	66.9	11.7	36.8
64	131.1	42.8	28.9
68	237.4	27.5	11.0

160 rows × 3 columns

In [21]:
```
x_test
```

Out[21]:

	TV	Radio	Newspaper
56	7.3	28.1	41.4
37	74.7	49.4	45.7
67	139.3	14.5	10.2
79	116.0	7.7	23.1

80	76.4	26.7	22.3
188	286.0	13.9	3.7
183	287.6	43.0	71.8
10	66.1	5.8	24.2
128	220.3	49.0	3.2
62	239.3	15.5	27.3
65	69.0	9.3	0.9
17	281.4	39.6	55.8
133	219.8	33.5	45.1
195	38.2	3.7	13.8
146	240.1	7.3	8.7
38	43.1	26.7	35.1
173	168.4	7.1	12.8
149	44.7	25.8	20.6
93	250.9	36.5	72.3
29	70.6	16.0	40.8
0	230.1	37.8	69.2
2	17.2	45.9	69.3
122	224.0	2.4	15.6
180	156.6	2.6	8.3
95	163.3	31.6	52.9
121	18.8	21.7	50.4
185	205.0	45.1	19.6
39	228.0	37.7	32.0
66	31.5	24.6	2.2
19	147.3	23.9	19.1
11	214.7	24.0	4.0
45	175.1	22.5	31.5
41	177.0	33.4	38.7
92	217.7	33.5	59.0
168	215.4	23.6	57.6
1	44.5	39.3	45.1
57	136.2	19.2	16.6
189	18.7	12.1	23.4
151	121.0	8.4	48.7
167	206.8	5.2	19.4

In [22]:
```
y_train
```

Out[22]:
```
116    12.2
138     9.6
155     3.2
82     11.3
160    14.4
       ...
58     23.8
21     12.5
49      9.7
64     18.0
68     18.9
Name: Sales, Length: 160, dtype: float64
```

In [23]:
```
y_test
```

Out[23]:
```
56      5.5
37     14.7
67     13.4
79     11.0
80     11.8
188    15.9
183    26.2
10      8.6
128    24.7
62     15.7
65      9.3
17     24.4
133    19.6
195     7.6
146    13.2
38     10.1
173    11.7
149    10.1
93     22.2
29     10.5
0      22.1
2       9.3
122    11.6
180    10.5
95     16.9
121     7.0
185    22.6
39     21.5
66      9.5
19     14.6
11     17.4
45     14.9
41     17.1
92     19.4
168    17.1
1      10.4
57     13.2
189     6.7
151    11.6
167    12.2
Name: Sales, dtype: float64
```

In [24]:
```
x_train=x_train.astype(int)
y_train=y_train.astype(int)
x_test=x_test.astype(int)
y_test=y_test.astype(int)
```

In [25]:
```
from sklearn.preprocessing import StandardScaler
Sc=StandardScaler()
```

```
x_train_scaled=Sc.fit_transform(x_train)
x_test_scaled=Sc.fit_transform(x_test)
```

Applying Linear Regression

In [26]:
```
from sklearn.linear_model import LinearRegression
```

In [27]:
```
lr=LinearRegression()
```

In [28]:
```
lr.fit(x_train_scaled,y_train)
```

Out[28]:
```
▾ LinearRegression
LinearRegression()
```

In [29]:
```
y_pred=lr.predict(x_test_scaled)
print(y_pred)
```

```
[ 8.07208561 15.39694276 11.20723017  8.7231933  10.80138195 17.66036561
 23.785389    6.05064595 22.03934034 15.92229543  7.02529552 22.70917874
 18.68464752  4.38385522 14.34701329  9.2970207  11.08559493  9.15049517
 20.67810902  8.46853177 19.98026988 11.97879363 12.59404776  9.52303143
 15.73547183  7.12996739 20.52852873 19.92139498  8.37303394 13.40290607
 16.65207302 14.45505255 16.79013098 18.58218129 16.44742571 11.99347795
 12.08933433  5.31167661  9.13267146 12.38991849]
```

Evaluate the performance of a Linear Regerssion Model

In [30]:
```
from sklearn.metrics import r2_score
```

In [31]:
```
r2_score(y_test,y_pred)
```

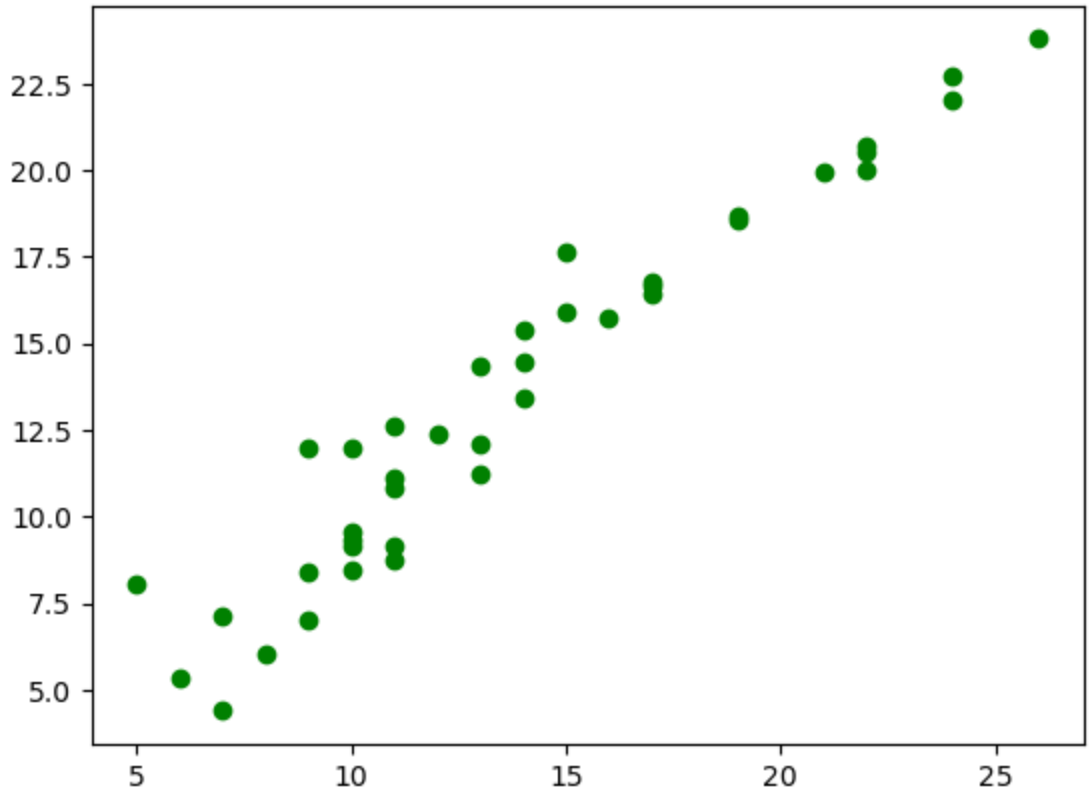
Out[31]:
```
0.9222988021105912
```

Analyzing Data By Scatter Plot

In [32]:
```
import matplotlib.pyplot as plt
```

In [33]:
```
plt.scatter(y_test,y_pred,c='g')
```

Out[33]:
```
<matplotlib.collections.PathCollection at 0x2e514c27090>
```



In [ ]: