Sales Prediction

In [ ]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.linear_model import LinearRegression
```

In [2]:

```
sd=pd.read_csv("advertising.csv")
sd.head(20)
```

Out[2]:

	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	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
5	8.7	48.9	75.0	7.2
6	57.5	32.8	23.5	11.8
7	120.2	19.6	11.6	13.2
8	8.6	2.1	1.0	4.8
9	199.8	2.6	21.2	15.6
10	66.1	5.8	24.2	12.6
11	214.7	24.0	4.0	17.4
12	23.8	35.1	65.9	9.2
13	97.5	7.6	7.2	13.7
14	204.1	32.9	46.0	19.0
15	195.4	47.7	52.9	22.4
16	67.8	36.6	114.0	12.5
17	281.4	39.6	55.8	24.4
18	69.2	20.5	18.3	11.3
19	147.3	23.9	19.1	14.6

In [3]:

```
sd.isnull().sum
```

Out[3]:

```
<bound method NDFrame._add_numeric_operations.<locals>.sum of         TV  Radio  N
ewspaper  Sales
0    False  False      False  False
1    False  False      False  False
2    False  False      False  False
3    False  False      False  False
4    False  False      False  False
..     ...    ...        ...    ...
195  False  False      False  False
196  False  False      False  False
197  False  False      False  False
198  False  False      False  False
199  False  False      False  False

[200 rows x 4 columns]>
```

In [4]:

```python
sd.describe
```

Out[4]:

```
<bound method NDFrame.describe of         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   12.0
3    151.5   41.3       58.5   16.5
4    180.8   10.8       58.4   17.9
..     ...    ...        ...    ...
195   38.2    3.7       13.8    7.6
196   94.2    4.9        8.1   14.0
197  177.0    9.3        6.4   14.8
198  283.6   42.0       66.2   25.5
199  232.1    8.6        8.7   18.4

[200 rows x 4 columns]>
```

In [5]:

```python
sd.dtypes
```

Out[5]:

```
TV           float64
Radio        float64
Newspaper    float64
Sales        float64
dtype: object
```

In [6]:

```python
sd.duplicated().sum()
```

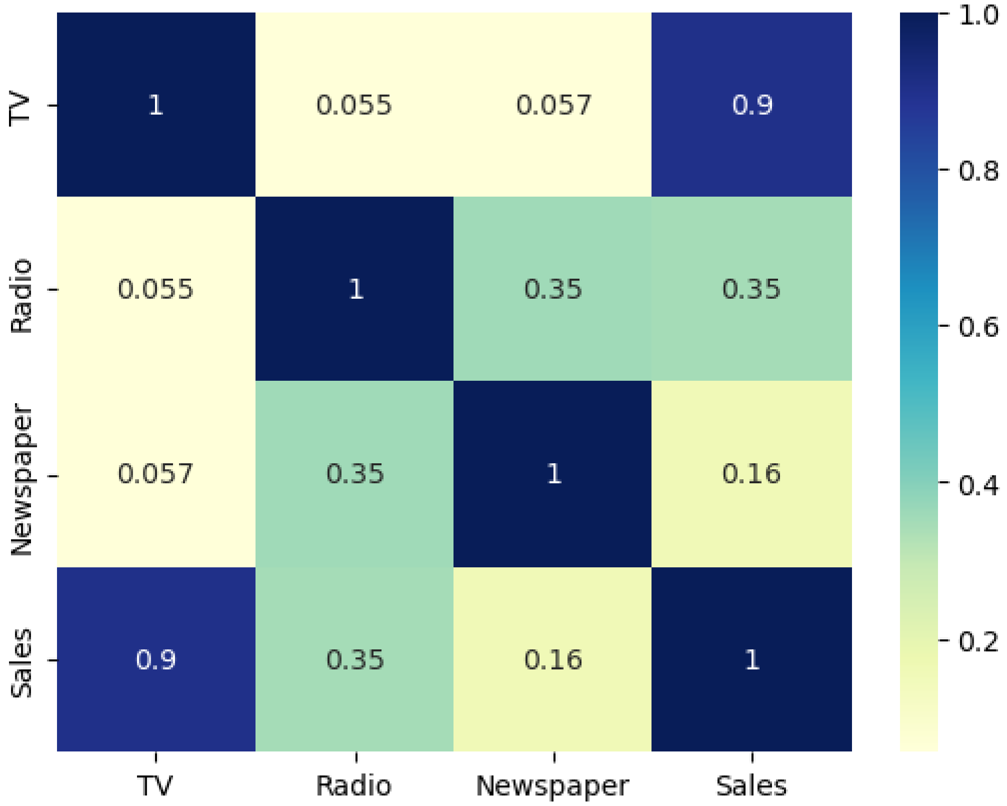
Out[6]:

```
0
```

Feature engineering

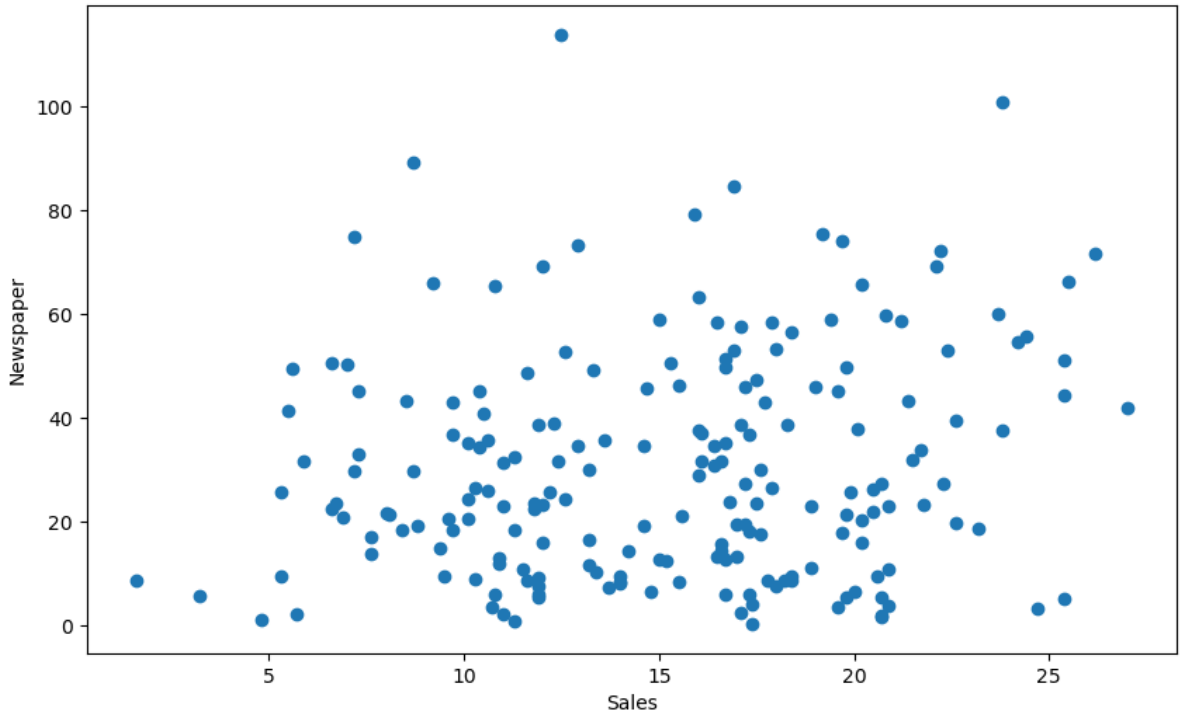
In [7]:

```python
sns.heatmap(sd.corr(),annot=True,cmap="YlGnBu")
plt.show()
```



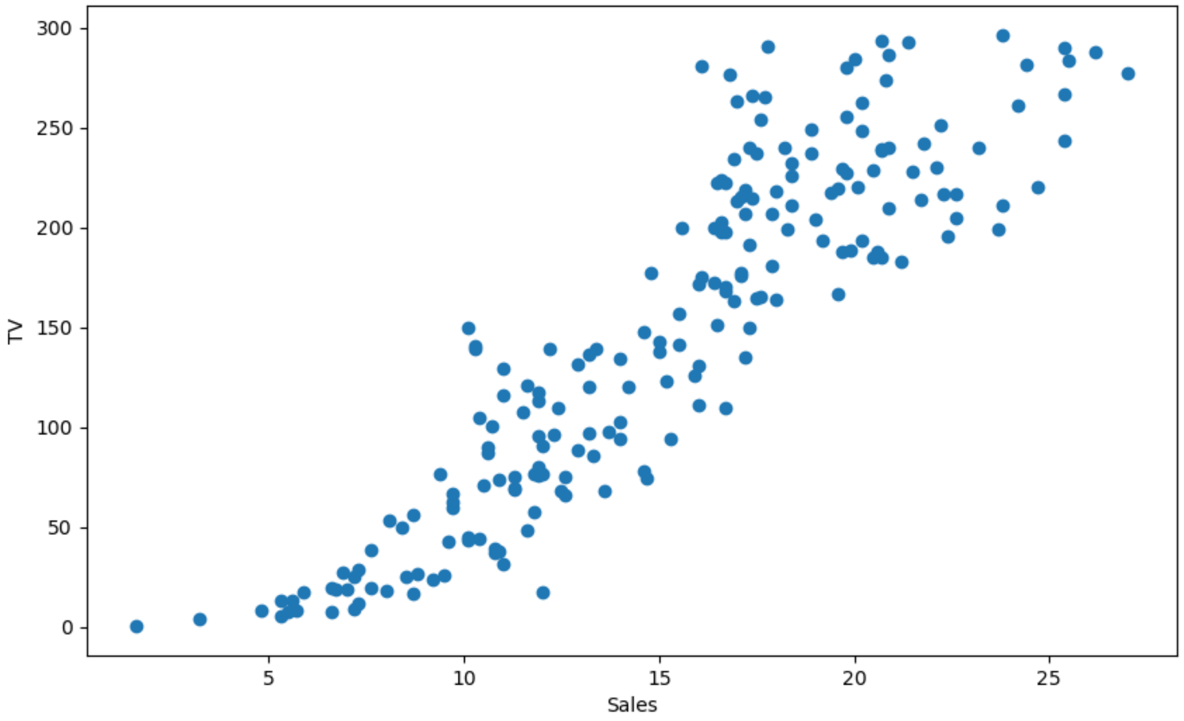
In [8]:

```
fig, ax = plt.subplots(figsize=(10,6))
ax.scatter(sd['Sales'], sd['Newspaper'])
ax.set_xlabel('Sales')
ax.set_ylabel('Newspaper')
plt.show()
```



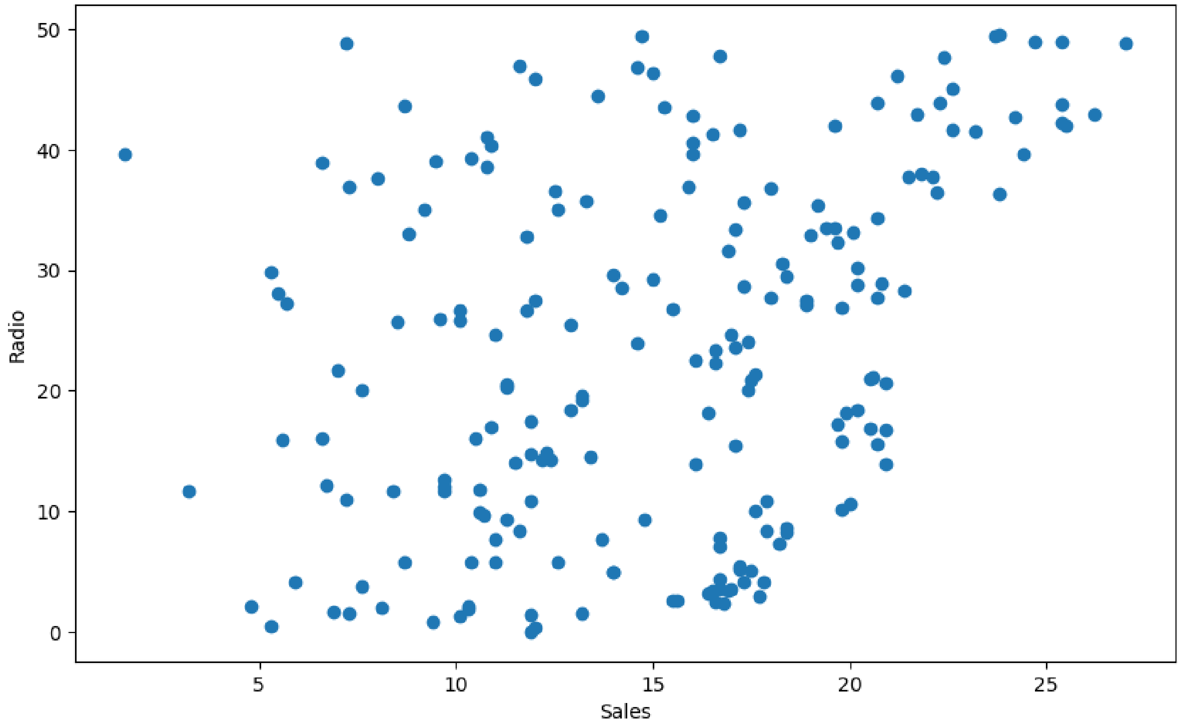
In [9]:

```
fig, ax = plt.subplots(figsize=(10,6))
ax.scatter(sd['Sales'], sd['TV'])
ax.set_xlabel('Sales')
ax.set_ylabel('TV')
plt.show()
```



In [10]:

```
fig, ax = plt.subplots(figsize=(10,6))
ax.scatter(sd['Sales'], sd['Radio'])
ax.set_xlabel('Sales')
ax.set_ylabel('Radio')
plt.show()
```



In [30]:

```
from sklearn.linear_model import LinearRegression

# Assuming 'sd' is your DataFrame
x1 = sd[['TV']]
y1 = sd['Sales']

# Create a LinearRegression model instance
model = LinearRegression()

# Fit the model to your data
```

```
model.fit(x1, y1)

# Get the model parameters (intercept and coefficient)
intercept = model.intercept_
coefficient = model.coef_

# Print the results
print("Value of B:", intercept)
print("Value of W:", coefficient)
```

```
Value of B: 6.9748214882298925
Value of W: [0.05546477]
```

In [ ]:

```

```

x = sm.add_constant(x1) results = sm.OLS(y1,x1).fit()

results.summary()

plt.scatter(x1,y1)

yhat = 0.0017*x1 + 0.275

fig = plt.plot(x1,yhat, lw=4, c='orange', label = 'regression line')

plt.xlabel('sales', fontsize = 20)

plt.ylabel('TV', fontsize = 20)

plt.show()

plt.scatter(x1,y1)

yhat = 0.05*x1 + 6.9

fig = plt.plot(x1,yhat, lw=4, c='orange', label = 'regression line')

plt.xlabel('sales', fontsize = 20)

plt.ylabel('TV', fontsize = 20)

plt.show()

In [31]:

```
y_pred=model.predict(x1)
print(y_pred)
```

```
[19.73726517  9.44300377  7.92881554 15.37773421 17.00285199  7.45736499
 10.16404579 13.6416869   7.45181851 18.05668263 10.64104282 18.88310771
  8.29488303 12.38263661 18.29518114 17.81263764 10.73533293 22.5826079
 10.8129836  15.14478218 19.08832736 20.142158    7.70695646 19.63742859
 10.43027669 21.55650964 14.90073719 20.29191288 20.77445638 10.89063428
 23.22045276 13.23679407 12.36599718 21.70626452 12.28280002 23.09843026
 21.77836873 11.11803984  9.3653531  19.62078916 18.20643751 16.79208586
 23.2592781  18.4504825   8.36698723 16.6867028  11.9500114  20.28081992
 19.57641734 10.68541463 18.05668263 12.54348444 18.97739782 17.10268858
 21.54541669 18.00676433  7.37971431 14.52912323 18.6667951  18.66124863
  9.94218671 21.46776601 20.24754106 12.67105342 14.2462529  10.80189065
  8.72196176 14.70106401 20.142158   18.99958373 18.01785729 13.06485329
  8.46127734 14.15196279 18.81100351  7.91217611  8.50010268 13.65832633
  7.27433125 13.40873486 11.21232995 20.27527345 11.1513187  10.76861179
 18.81654998 17.69061514 11.20678348 13.11477158 11.87236072 13.06485329
 14.42374016  8.56111392 19.04950202 20.8909324  12.93173784 16.03221851
 17.93466013 17.23025755 23.04296549 14.47365846 19.31018644 23.41457946
 22.51605017 17.39665186 20.18652981 14.62341334  8.36144075 11.98883674
  7.70140998 21.14052387 19.49876666 20.38065651 16.71998166 18.60023738
 11.31216654 11.14022575 14.69551754 11.21232995 13.94674314  8.05083804
 14.81199356  8.01755917 19.39893007 13.80253473 19.70398631 11.81134947
  7.4074467  11.42309608 19.19371042 10.28052181  7.01364683 21.68407862
  7.44072556 19.16597804  9.02147152  9.6537699   8.39471961 22.15552917
  9.35980662 17.23025755 11.04593564 17.71834753 19.20480338 12.77643648
 12.31053241 14.75652879 20.29191288 20.46385367  9.08248277  9.45409673
 22.54378256 13.68605872 17.93466013 16.47593667 17.39110538  7.20222705
 12.18296344 15.2834441   7.6237593  14.27953176 16.54249439 11.72815232
 17.42438424 16.04331146 13.47529259 19.98131016  7.96764088 18.44493602
 18.92193305 22.74345573  9.74806001 16.09877623  8.06193099 16.31508884
 19.31018644 22.33301643 20.75227047 16.41492542 22.32192348 16.15978748
 15.66060454 19.09387384 10.09194159 22.92648948 21.05178023 18.34509943
 14.71215697 17.57413912 22.83774584  8.0120127   9.16567992 11.16241166
  7.92881554 16.2263452  15.27789763  9.09357572 12.19960287 16.79208586
 22.70463039 19.84819471]
```

In [29]:

```
results = pd.DataFrame()
```

In [33]:

```
results = pd.DataFrame({'Coefficient': model.coef_, 'Intercept': model.intercept_})
results.to_csv('linear_regression_results.csv', index=False)
```

In [5]:

```
x=input("Enter value of TV promotion:")
x1=float(x)
r=0.05546477*x1+6.9748214882298925
print("Sale will be:",r)
```

```
Enter value of TV promotion:34
Sale will be 8.860623668229891
```

In [ ]: