In [1]:

```
import opendatasets as od
od.download("https://www.kaggle.com/datasets/ashydv/advertising-dataset")
```

```
Please provide your Kaggle credentials to download this dataset. Learn more:
http://bit.ly/kaggle-creds (http://bit.ly/kaggle-creds)
Your Kaggle username: masapetaalishaanjum
Your Kaggle Key: ·····
Downloading advertising-dataset.zip to .\advertising-dataset

100%|
       | 1.83k/1.83k [00:00<00:00, 234kB/s]
```

In [2]:

```
import pandas as pd
import numpy as np
d=pd.read_csv('advertising.csv')
d
```

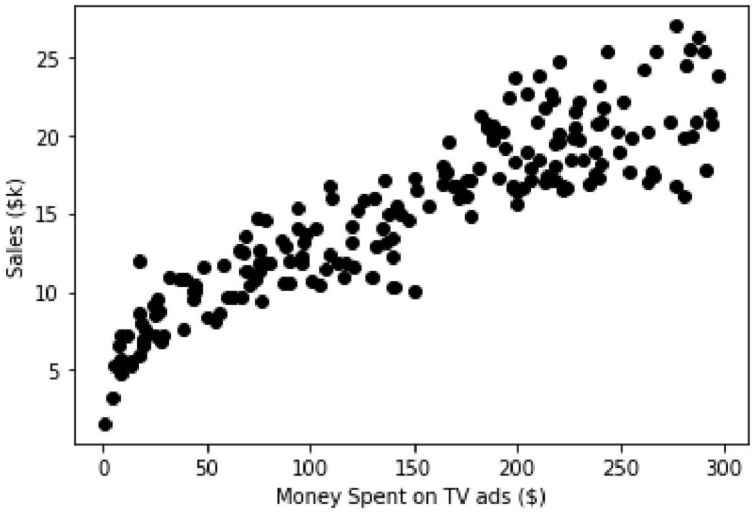
Out[2]:

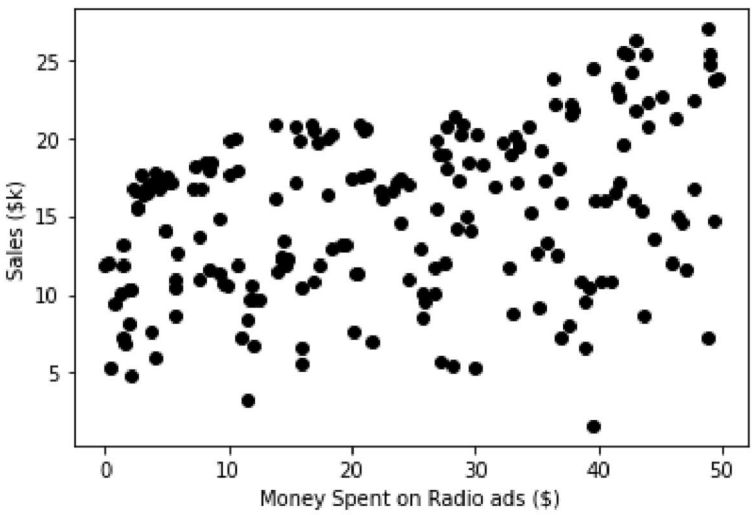
	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

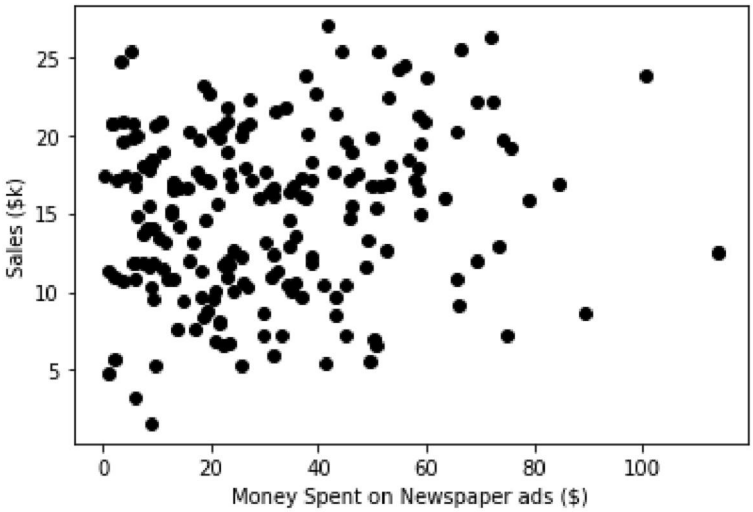
200 rows × 4 columns

In [6]:

```
import matplotlib.pyplot as plt
def scatter_plot(feature, target):
    plt.scatter(d[feature],
                d[target],
                c='black'
                )
    plt.xlabel("Money Spent on {} ads ($)".format(feature))
    plt.ylabel("Sales ($k)")
    plt.show()
scatter_plot("TV", "Sales")
scatter_plot("Radio", "Sales")
scatter_plot("Newspaper", "Sales")
```







In [8]:

```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression

xs = d.drop(["Sales"], axis=1)
y = d["Sales"].values.reshape(-1,1)
linreg = LinearRegression()
MSE = cross_val_score(linreg, xs, y, scoring="neg_mean_squared_error", cv=5)

mean_MSE = np.mean(MSE)
print(mean_MSE)
```

```
-2.858243009991009
```

In [9]:

```
# Ridge Regression
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge
ridge = Ridge()

parameters = {"alpha":[1e-15, 1e-10, 1e-8, 1e-4, 1e-3, 1e-2, 1, 5, 10, 20]}
ridge_regression = GridSearchCV(ridge, parameters, scoring='neg_mean_squared_error', cv=5)
ridge_regression.fit(xs, y)
print(ridge_regression.best_params_)
print(ridge_regression.best_score_)
```

```
{'alpha': 20}
-2.858183819191847
```

In [10]:

```
#Lasso Regression
from sklearn.linear_model import Lasso
lasso = Lasso()

parameters = {"alpha":[1e-15, 1e-10, 1e-8, 1e-4, 1e-3, 1e-2, 1, 5, 10, 20]}
lasso_regression = GridSearchCV(lasso, parameters, scoring='neg_mean_squared_error', cv=5)
lasso_regression.fit(xs, y)

print(lasso_regression.best_params_)
print(lasso_regression.best_score_)
```

```
{'alpha': 1}
-2.851375129372136

C:\Users\ALISHA ANJUM\anaconda3\lib\site-packages\sklearn\linear_model\_coor
dinate_descent.py:530: ConvergenceWarning: Objective did not converge. You m
ight want to increase the number of iterations. Duality gap: 218.07054962659
353, tolerance: 0.43671069375
  model = cd_fast.enet_coordinate_descent(
C:\Users\ALISHA ANJUM\anaconda3\lib\site-packages\sklearn\linear_model\_coor
dinate_descent.py:530: ConvergenceWarning: Objective did not converge. You m
ight want to increase the number of iterations. Duality gap: 182.98789426660
773, tolerance: 0.458666
  model = cd_fast.enet_coordinate_descent(
C:\Users\ALISHA ANJUM\anaconda3\lib\site-packages\sklearn\linear_model\_coor
dinate_descent.py:530: ConvergenceWarning: Objective did not converge. You m
ight want to increase the number of iterations. Duality gap: 171.69369428124
58, tolerance: 0.4210973437500001
  model = cd_fast.enet_coordinate_descent(
C:\Users\ALISHA ANJUM\anaconda3\lib\site-packages\sklearn\linear_model\_coor
dinate_descent.py:530: ConvergenceWarning: Objective did not converge. You m
ight want to increase the number of iterations. Duality gap: 181.96106388117
983, tolerance: 0.44512240000000003
  model = cd_fast.enet_coordinate_descent(
```

In this case, the lasso is the best method of adjustment, with a regularization value of 1.

In [ ]: