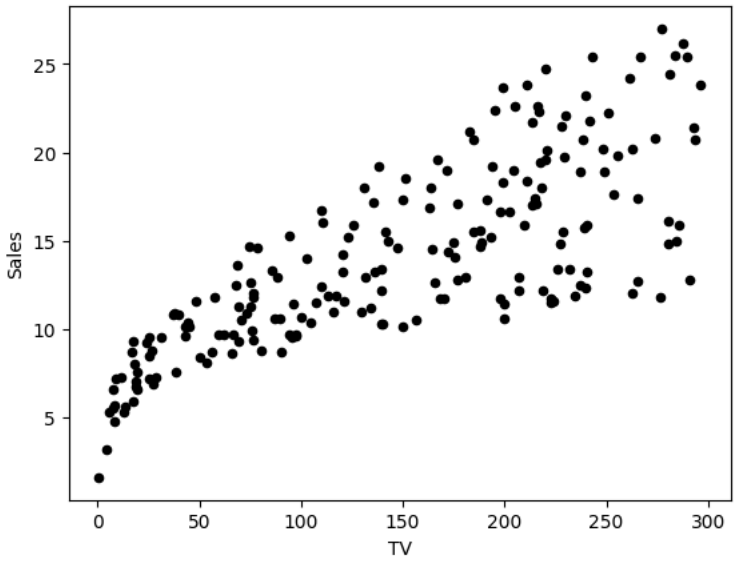
```
import numpy as np
# A library for programmatic plot generation.
import matplotlib.pyplot as plt
# A library for data manipulation and analysis.
import pandas as pd
# LinearRegression from sklearn.
from sklearn.linear_model import LinearRegression


path = "/tvmarketing.csv"

### START CODE HERE ### (~ 1 line of code)
adv = pd.read_csv(path)
### END CODE HERE ###


# Print some part of the dataset.
adv.head()
```

	TV	Sales
0	230.1	22.1
1	44.5	10.4
2	17.2	9.3
3	151.5	18.5
4	180.8	12.9

Next steps: Generate code with adv | View recommended plots | New interactive sheet

```
adv.plot(x='TV', y='Sales', kind='scatter', c='black')
```

```
<Axes: xlabel='TV', ylabel='Sales'>
```




```
X = adv['TV']
Y = adv['Sales']


m_numpy, b_numpy = np.polyfit(X, Y, 1)

print(f"Linear regression with NumPy. Slope: {m_numpy}. Intercept: {b_numpy}")
```

```
Linear regression with NumPy. Slope: 0.04753664043301975. Intercept: 7.0325935491276965
```

```
# This is organised as a function only for grading purposes.
def pred_numpy(m, b, X):
    ### START CODE HERE ### (~ 1 line of code)
    Y = m * X + b
    ### END CODE HERE ###

    return Y


X_pred = np.array([50, 120, 280])
Y_pred_numpy = pred_numpy(m_numpy, b_numpy, X_pred)
```

```
print(f"TV marketing expenses:\n{X_pred}")
print(f"Predictions of sales using NumPy linear regression:\n{Y_pred_numpy}")
```

```
TV marketing expenses:
[ 50 120 280]
Predictions of sales using NumPy linear regression:
[ 9.40942557 12.7369904  20.34285287]
```

```
lr_sklearn = LinearRegression()
```

```
print(f"Shape of X array: {X.shape}")
print(f"Shape of Y array: {Y.shape}")

try:
    lr_sklearn.fit(X, Y)
except ValueError as err:
    print(err)
```

```
Shape of X array: (200,)
Shape of Y array: (200,)
Expected a 2-dimensional container but got <class 'pandas.core.series.Series'> instead. Pass a DataFrame containing a si
```