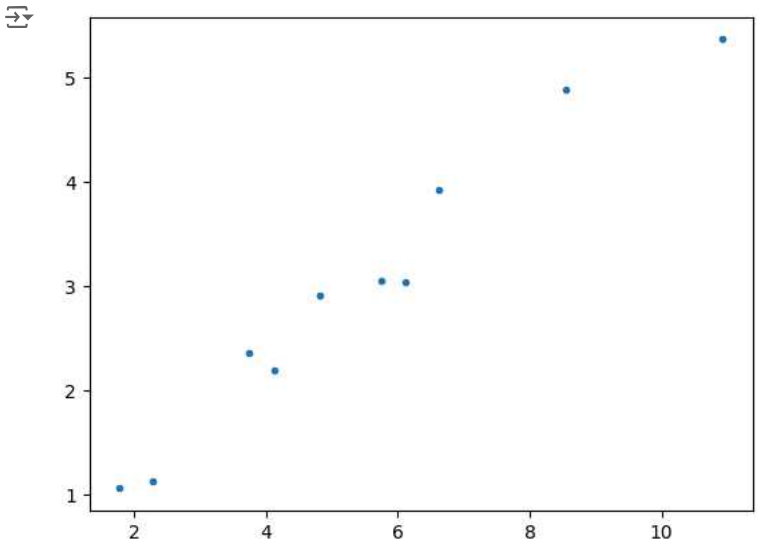
R2

Linear Regression

```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(44)

x = np.array([1.78, 6.12, 6.63, 4.13, 8.55, 10.92, 4.81, 3.75, 2.28, 5.74]).reshape(-1, 1)
y = np.array([1.06, 3.04, 3.93, 2.19, 4.89, 5.37, 2.91, 2.36, 1.13, 3.05])

# graph
plt.plot(x, y, '.')
plt.show()
```



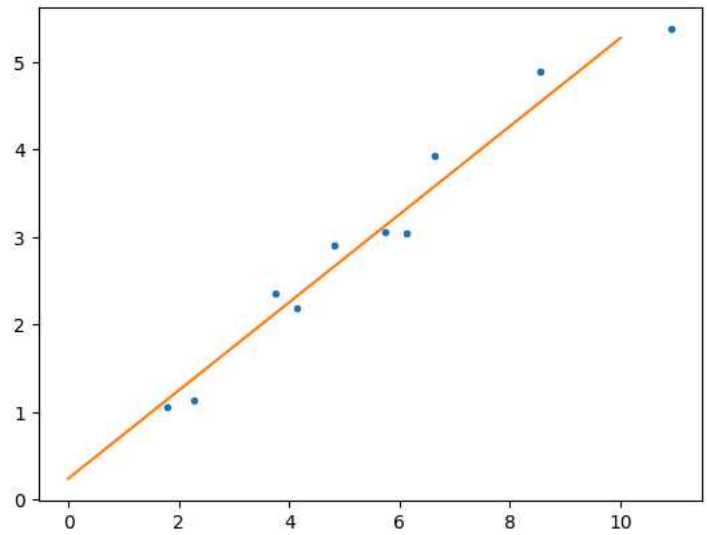

```
## linear regression using scikit learn
from sklearn.linear_model import LinearRegression

model_linear = LinearRegression()
model_linear.fit(x, y)
```

```
LinearRegression()
```

```
## predict using test data set
x_test = np.linspace(0, 10, 100).reshape(100, 1)
y_test = model_linear.predict(x_test)

## graph
plt.plot(x, y, '.')
plt.plot(x_test, y_test)
plt.show()
```



R2

```
from sklearn.metrics import r2_score

def my_re_score(y_true, y_pred):
  y_mean = np.mean(y_true)
  tss = np.sum((y_true - y_mean) ** 2)
  ess = np.sum((y_true - y_pred) ** 2)
  r2 = 1 - (ess / tss)
  return r2

y_hat = model_linear.predict(x)

r2 = my_re_score(y.reshape(-1, 1), y_hat.reshape(-1, 1))
print(f'R2: {r2}')

```

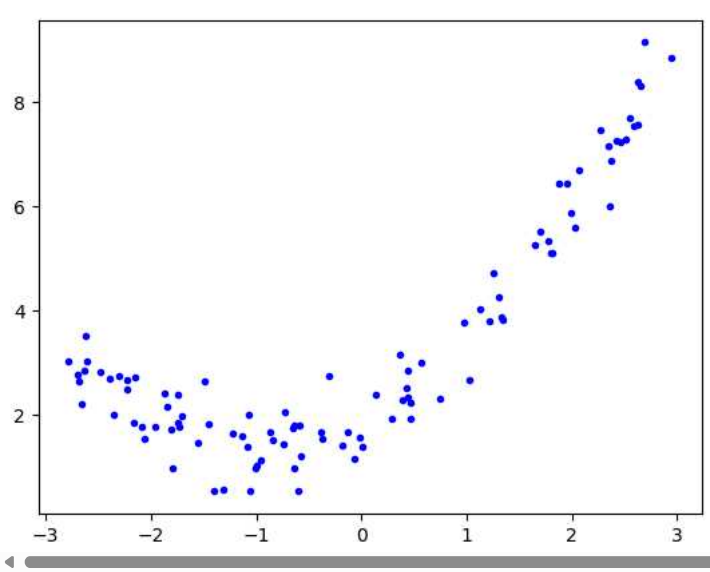
```
R2: 0.9636537722997088
```

Polynomial Regression

$$y = 0.5x^2 + x + 2 + \epsilon$$

```
# generate data set for test
m = 100
x = 6 * np.random.rand(m, 1) - 3
y = 0.5 * x**2 + x + 2 + np.random.randn(m, 1) * 0.5

# graph
plt.plot(x, y, "b.")
plt.show()
```



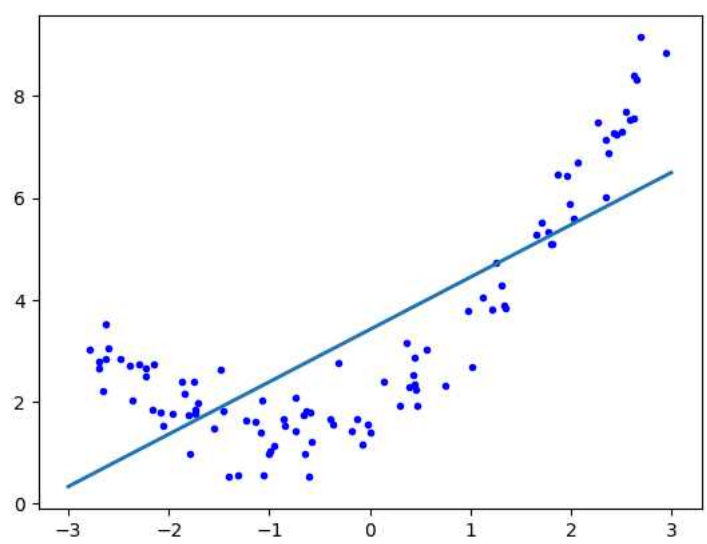
```
# try linear regression
from sklearn.linear_model import LinearRegression

model_linear = LinearRegression()
model_linear.fit(x, y)
```

```
LinearRegression()
```

```
# predict using test data set
x_new = np.linspace(-3, 3, 100).reshape(100, 1)
y_pred = model_linear.predict(x_new)

# graph
plt.plot(x, y, "b.")
plt.plot(x_new, y_pred, "-", linewidth=2, label="linear regression")
plt.show()
```




```
# r2 score
y_hat = model_linear.predict(x)

r2 = r2_score(y, y_hat)
print(f'R2: {r2}')
```

```
R2: 0.6219346688532572
```