

```
In [30]: import pandas as pd

df = pd.read_csv('./dataset 1.csv')

df.head()
```

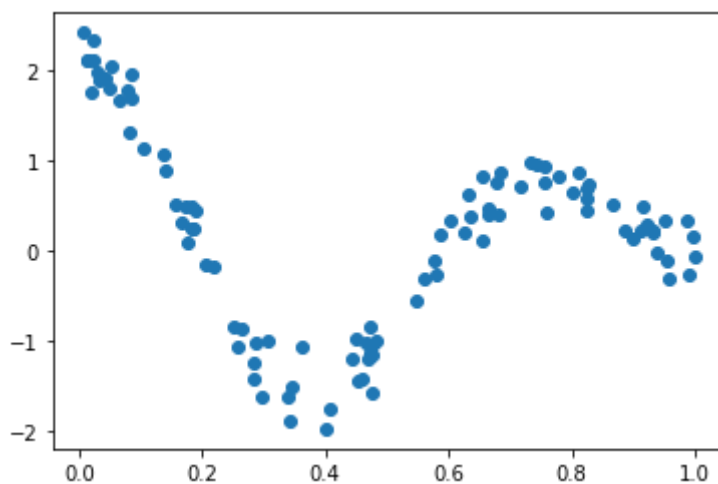
```
Out[30]:
```

	X1	X2
0	0.007486	2.420864
1	0.013216	2.112454
2	0.013879	2.122164
3	0.019414	1.756409
4	0.022808	2.342029

```
In [31]: import matplotlib.pyplot as plt

plt.scatter(df.X1, df.X2)
```

```
Out[31]: <matplotlib.collections.PathCollection at 0x2422173ebc8>
```



```
In [32]: import numpy as np

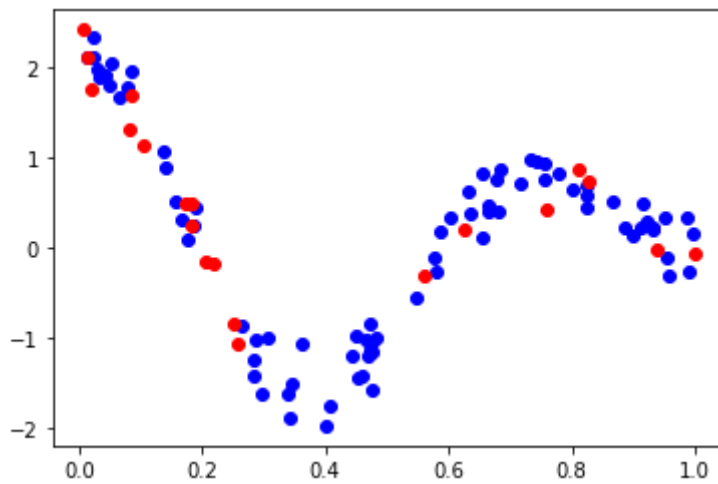
x = df[['X1']].to_numpy()
y = df[['X2']].to_numpy()
```

```
In [33]: 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_stat
```

```
In [34]: plt.scatter(x_train, y_train, color='blue')
plt.scatter(x_test, y_test, color='red')
```

```
Out[34]: <matplotlib.collections.PathCollection at 0x242202fa888>
```

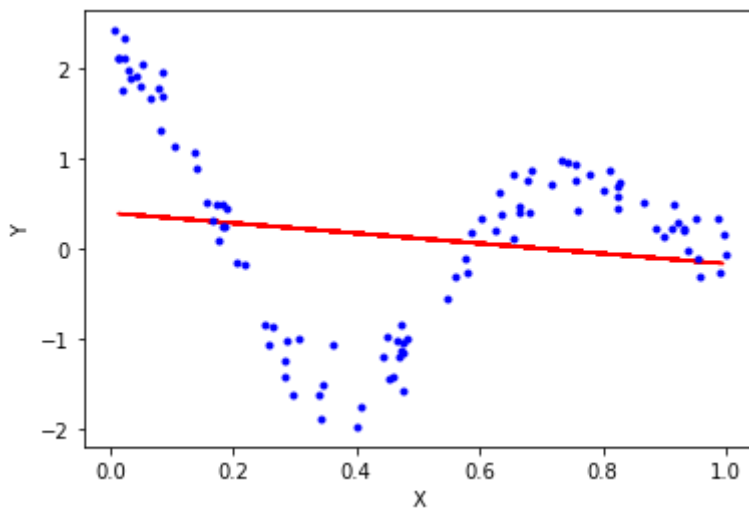


```
In [35]: from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import r2_score

lr = LinearRegression()
lr.fit(x_train, y_train)
y_pred = lr.predict(x_test)
print(r2_score(y_test, y_pred))
```

-0.03873526902255553

```
In [36]: plt.plot(x_train, lr.predict(x_train), color="r")
plt.plot(x, y, "b.")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```



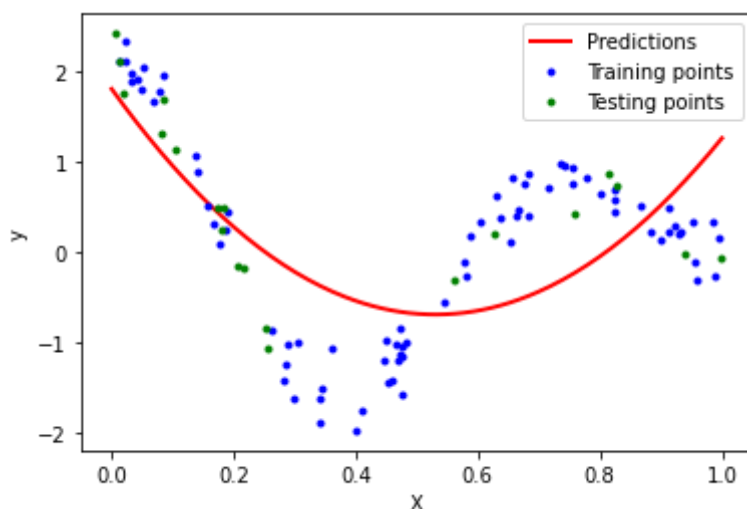
```
In [37]: #applying polynomial regression degree 2
poly = PolynomialFeatures(degree=2, include_bias=True)
x_train_trans = poly.fit_transform(x_train)
x_test_trans = poly.transform(x_test)
#include bias parameter
lr = LinearRegression()
lr.fit(x_train_trans, y_train)
y_pred = lr.predict(x_test_trans)
print(r2_score(y_test, y_pred))
```

0.5289836811559705

```
In [38]: print(lr.coef_)
print(lr.intercept_)
```

```
[[ 0.          -9.43093592  8.88472968]]
[1.80772619]
```

```
In [39]: X_new = np.linspace(0, 1, 200).reshape(200, 1)
X_new_poly = poly.transform(X_new)
y_new = lr.predict(X_new_poly)
plt.plot(X_new, y_new, "r-", linewidth=2, label="Predictions")
plt.plot(x_train, y_train, "b.", label='Training points')
plt.plot(x_test, y_test, "g.", label='Testing points')
plt.xlabel("X")
plt.ylabel("y")
plt.legend()
plt.show()
```



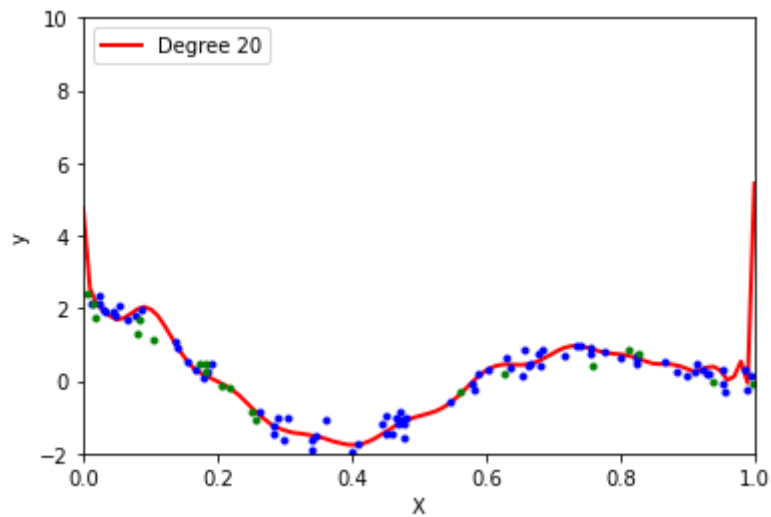
```
In [40]: from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

def polynomial_regression(degree):
    X_new=np.linspace(0, 1, 100).reshape(100, 1)
    X_new_poly = poly.transform(X_new)
    polybig_features = PolynomialFeatures(degree=degree, include_bias=False)
    std_scaler = StandardScaler()
    lin_reg = LinearRegression()
    polynomial_regression = Pipeline([
        ("poly_features", polybig_features),
        ("std_scaler", std_scaler),
        ("lin_reg", lin_reg),
    ])
    polynomial_regression.fit(x_train_trans, y_train)

    y_newbig = polynomial_regression.predict(X_new_poly)

    #plotting prediction line
    plt.plot(X_new, y_newbig, 'r', label="Degree " + str(degree), linewidth=2)
    plt.plot(x_train, y_train, "b.", linewidth=3)
    plt.plot(x_test, y_test, "g.", linewidth=3)
    plt.legend(loc="upper left")
    plt.xlabel("X")
    plt.ylabel("y")
    plt.axis([0, 1, -2, 10])
    plt.show()
```

In [41]: `polynomial_regression(20)`



```
In [42]: from sklearn.model_selection import KFold
kf = KFold(n_splits=2, random_state=None)

for train_index, test_index in kf.split(x_train_trans):
    print("Train:", train_index, "Validation:", test_index)
    x_train, x_test = x[train_index], x[test_index]
    y_train, y_test = y[train_index], y[test_index]
```

```
Train: [40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63
 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79] Validation: [ 0  1  2  3  4  5  6
 7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39]
Train: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39] Validation: [40 41 42 43 44 45 46
 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63
 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79]
```