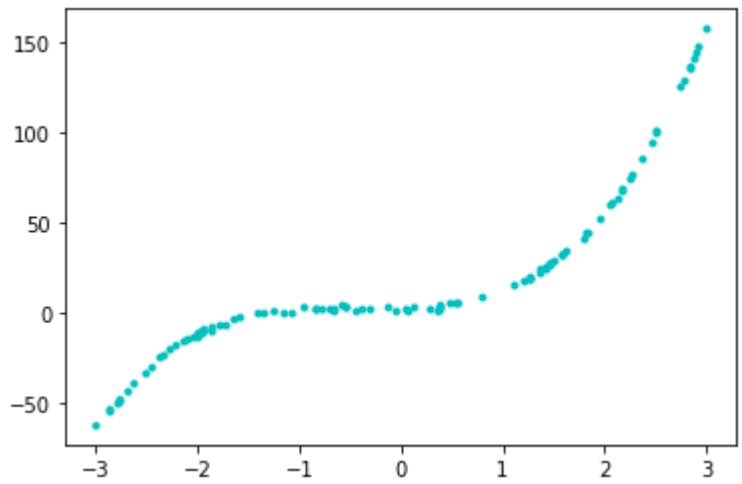
Open in Colab

Homework 4

This homework assignment will build three polynomial models on the same dataset and compare their performance. You can use tools from `sklearn` to complete this task.

```
# Link to notebook: https://colab.research.google.com/drive/11AjVUV4WXlb8qHeyuDuDk5nHkvR
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import PolynomialFeatures
%matplotlib inline
```

```
# Generate a data set X and Y
m = 100
X = 6 * np.random.rand(m, 1) - 3
Y = 4 * X ** 3 + 5 * X ** 2 + X + 2 + np.random.randn(m, 1)
plt.plot(X, Y, 'c.')
plt.show()
```



1. Use train_test_split to split the data into training set (80%) and test set (20%).

```
df = pd.DataFrame(list(zip(map(float, X), map(float, Y))), columns=['X', 'Y'])
train, test = train_test_split(df, test_size=0.2)
```

```
plt.plot(train.X, train.Y, 'c.')
plt.show()
```

2. Build a degree 2 polynomial regression model with `X` as input variables and `Y` as output variable. Name the model model_pr2. Train the model on the training set and obtain the MSE on the test set.

```
pre_proc2 = PolynomialFeatures(degree=2, include_bias=False)
model_pr2 = LinearRegression()
X2 = train[['X']]
Y2 = train.Y
X_POLY2 = pre_proc2.fit_transform(X2)
model_pr2.fit(X_POLY2, Y2)
y_pred2 = model_pr2.predict(X_POLY2)
theta01 = model_pr2.intercept_
theta11, theta21 = model_pr2.coef_
x_coor2 = np.linspace(-4, 4, 30)
y_coor2 = [theta01 + theta11 * x + theta21 * x ** 2 for x in x_coor2]
plt.plot(train.X, Y2, 'b.', alpha=0.7)
plt.plot(x_coor2, y_coor2, 'g-')
mean_squared_error(Y2, y_pred2)
```

3. Build a degree 3 polynomial regression model with `X` as input variables and `Y` as output variable. Name the model model_pr3. Train the model on the training set and obtain the MSE on the test set.

```
pre_proc3 = PolynomialFeatures(degree=3, include_bias=False)
model_pr3 = LinearRegression()
X3 = train[['X']]
Y3 = train.Y
X_POLY3 = pre_proc2.fit_transform(X3)
model_pr3.fit(X_POLY3, Y3)
y_pred3 = model_pr3.predict(X_POLY2)
theta03 = model_pr3.intercept_
theta13, theta23 = model_pr3.coef_
x_coor3 = np.linspace(-4, 4, 30)
y_coor3 = [theta03 + theta13 * x + theta23 * x ** 2 + theta23 * x ** 3 for x in x_coor3]
plt.plot(train.X, Y3, 'b.', alpha=0.7)
plt.plot(x_coor3, y_coor3, 'g-')
mean_squared_error(Y3, y_pred3)
```

4. Build a degree 10 polynomial regression model with `X` as input variables and `Y` as output variable. Name the model model_pr10. Train the model on the training set and obtain the MSE on the test set.

```
pre_proc10 = PolynomialFeatures(degree=10, include_bias=False)
model_pr10 = LinearRegression()
X10 = train[['X']]
Y10 = train.Y
X_POLY10 = pre_proc2.fit_transform(X10)
model_pr10.fit(X_POLY10, Y10)
```

```
y_pred10 = model_pr10.predict(X_POLY10)
theta010 = model_pr10.intercept_
theta110, theta210 = model_pr10.coef_
x_coor10 = np.linspace(-4, 4, 30)

y_coor10 = [theta010 + theta110 * x + theta210 * x ** 2 +
theta210 * x ** 3 + theta210 * x ** 4 + theta210 * x ** 5 +
theta210 * x ** 6 + theta210 * x ** 7 + theta210 * x ** 8 +
theta210 * x ** 9 + theta210 * x ** 10 for x in x_coor10]

plt.plot(train.X, Y10, 'b.', alpha=0.7)
plt.plot(x_coor10, y_coor10, 'g-')
mean_squared_error(Y10, y_pred10)
```