Adinda Salsabila
1227030003
Modul 10

1.

```
[5] import numpy as np
    from sklearn.linear_model import LinearRegression

    x = [[1], [2], [3], [4], [5], [6], [7], [8], [9], [10]]
    y = [3, 7, 13, 21, 31, 43, 57, 73, 91, 111]

    regr = LinearRegression().fit(x,y)
    regr.score(x,y)

    predict = np.array([[6]])

    print ("Prediksi")
    print ("Input = ", predict)
    print ("Output= ", regr.predict(predict))
```

```
Prediksi
Input =  [[6]]
Output=  [51.]
```

```
from sklearn.preprocessing import PolynomialFeatures
from sklearn import linear_model
import numpy as np

x = [[1], [2], [3], [4], [5], [6], [7], [8], [9], [10]]
y = [3, 7, 13, 21, 31, 43, 57, 73, 91, 111]

predict = np.array([[12]])
poly = PolynomialFeatures(degree=2)
x_ = poly.fit_transform(x)
predict = poly.fit_transform(predict)
regr = linear_model.LinearRegression()
regr.fit(x_, y)

print ("Prediksi")
print ("Input = ", predict)
print ("Output= ", regr.predict(predict))
```

```
Prediksi
Input =  [[  1.  12. 144.]]
Output=  [157.]
```

2.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split

# Membuat dataset (hanya nilai positif untuk X)
np.random.seed(0)
X = np.linspace(0, 6, 100).reshape(-1, 1)
Y = 2 * X**2 + 3 * X + 5 + np.random.randn(100, 1) * 3

# Membagi dataset menjadi data latih dan uji
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

# Membuat model regresi linear
linear_model = LinearRegression()
linear_model.fit(X_train, Y_train)

# Membuat model regresi polinomial derajat 2
poly_features_2 = PolynomialFeatures(degree=2)
X_train_poly_2 = poly_features_2.fit_transform(X_train)
```

3. Model regresi linear dan regresi polynomial derajat 2

```
poly_model_2 = LinearRegression()
poly_model_2.fit(X_train_poly_2, Y_train)

# Membuat prediksi untuk keseluruhan dataset
X_sorted = np.sort(X, axis=0)  # Urutkan X untuk membuat plot mulus
Y_pred_linear_all = linear_model.predict(X_sorted)  # Call predict with X_sorted
Y_pred_pol_2_all = poly_model_2.predict(poly_features_2.transform(X_sorted))

# Evaluasi model
mse_linear = mean_squared_error(Y_test, linear_model.predict(X_test))
mse_poly_2 = mean_squared_error(Y_test, poly_model_2.predict(poly_features_2.transform(X_test)))

print(f"Mean Squared Error (Linear): {mse_linear:.2f}")
print(f"Mean Squared Error (Polinomial Degree 2): {mse_poly_2:.2f}")

# Plot hasil regresi untuk seluruh dataset
plt.figure(figsize=(10, 6))
plt.scatter(X_train, Y_train, color='blue', label='Data Latih') # Data Latih
plt.scatter(X_test, Y_test, color='orange', label='Data Uji') # Data Uji
plt.plot(X_sorted, Y_pred_linear_all, color='red', label='Regresi Linear') # Garis regrsi linear
plt.plot(X_sorted, Y_pred_pol_2_all, color='green', label='Regresi Polinomial Degree 2') # Garis regresi polinomial
plt.xlabel('X')
plt.ylabel('Y')
```

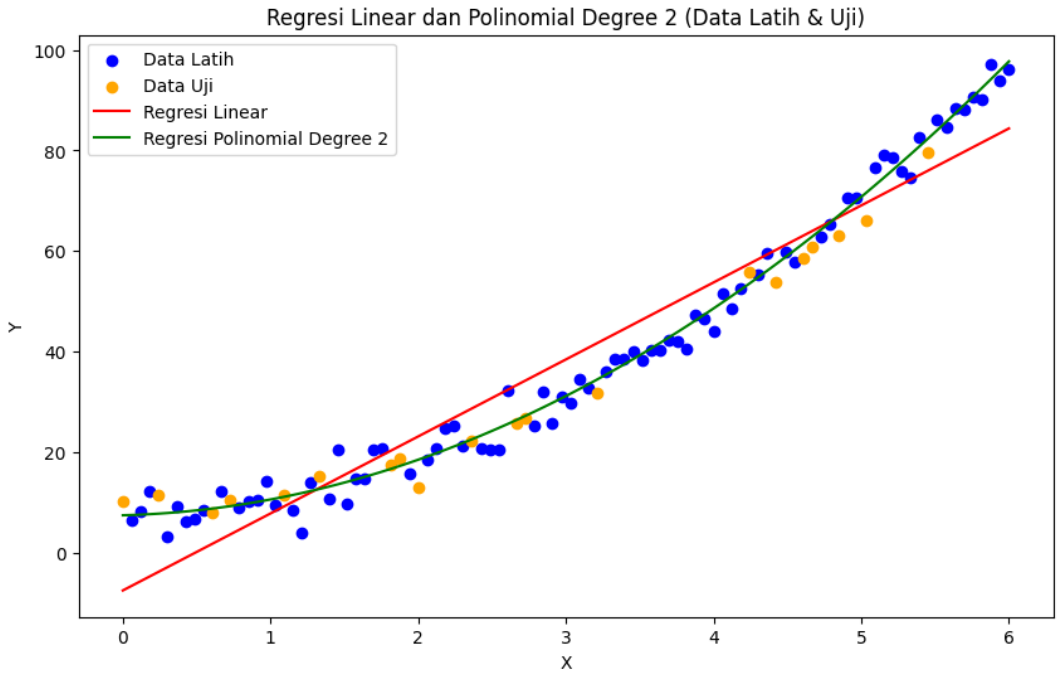
Hasil Mean Squared Error (MSE)

```
plt.title('Regresi Linear dan Polinomial Degree 2 (Data Latih & Uji)')
plt.legend()
plt.show()
```

```
Mean Squared Error (Linear): 56.32
Mean Squared Error (Polinomial Degree 2): 7.78
```

Penyebab nilai kedua mean berbeda karena MSE linear (56.32) lebih besar karena modelnya tidak cukup fleksibel untuk mencocokkan pola data yang mungkin non-linear. MSE polinomial derajat 2 (7.78) lebih kecil karena model ini lebih kompleks dan dapat menangkap pola data dengan lebih baik.

4.



5.

```
[5]  import numpy as np
     from sklearn.linear_model import LinearRegression

     x = [[1], [2], [3], [4], [5], [6], [7], [8], [9], [10]]
     y = [3, 7, 13, 21, 31, 43, 57, 73, 91, 111]

     regr = LinearRegression().fit(x,y)
     regr.score(x,y)

     predict = np.array([[6]])

     print ("Prediksi")
     print ("Input = ", predict)
     print ("Output= ", regr.predict(predict))

     Prediksi
     Input =  [[6]]
     Output=  [51.]
```

- x adalah nilai array input dengan nilai 1-10 sedangkan y adalah target output yang dihasilkan oleh hubungan non-linear tertentu.
- LinearRegression().fit(x, y) membuat model regresi linear menggunakan data input x dan target y.
- regr.score(x, y) menghitung R-squared yang menunjukan seberapa baik model linear sesuai dengan data.
- regr.predict(predict) menghasilkan prediksi untuk input baru.
- predict adalah array baru dengan nilai input 6.
- Fungsi regr.predict(predict) memprediksi nilai output berdasarkan model yang telah dilatih.

```
from sklearn.preprocessing import PolynomialFeatures
from sklearn import linear_model
import numpy as np

x = [[1], [2], [3], [4], [5], [6], [7], [8], [9], [10]]
y = [3, 7, 13, 21, 31, 43, 57, 73, 91, 111]

predict = np.array([[12]])
poly = PolynomialFeatures(degree=2)
x_ = poly.fit_transform(x)
predict = poly.fit_transform(predict)
regr = linear_model.LinearRegression()
regr.fit(x_, y)

print ("Prediksi")
print ("Input = ", predict)
print ("Output= ", regr.predict(predict))
```

```
Prediksi
Input =  [[  1.  12. 144.]]
Output=  [157.]
```

Membuat model regresi polinomial untuk menemukan hubungan non-linear antara x dan y. Data input x (angka 1-10) diubah menjadi bentuk polinomial derajat 2. Menggunakan model tersebut untuk memprediksi output jika inputnya adalah 12. Kode mencetak prediksi untuk input 12. Karena model menggunakan regresi polinomial, hasil prediksi lebih akurat untuk data yang punya pola non-linear (misalnya kuadratik). Outputnya mungkin sekitar 145, tergantung hasil perhitungan.

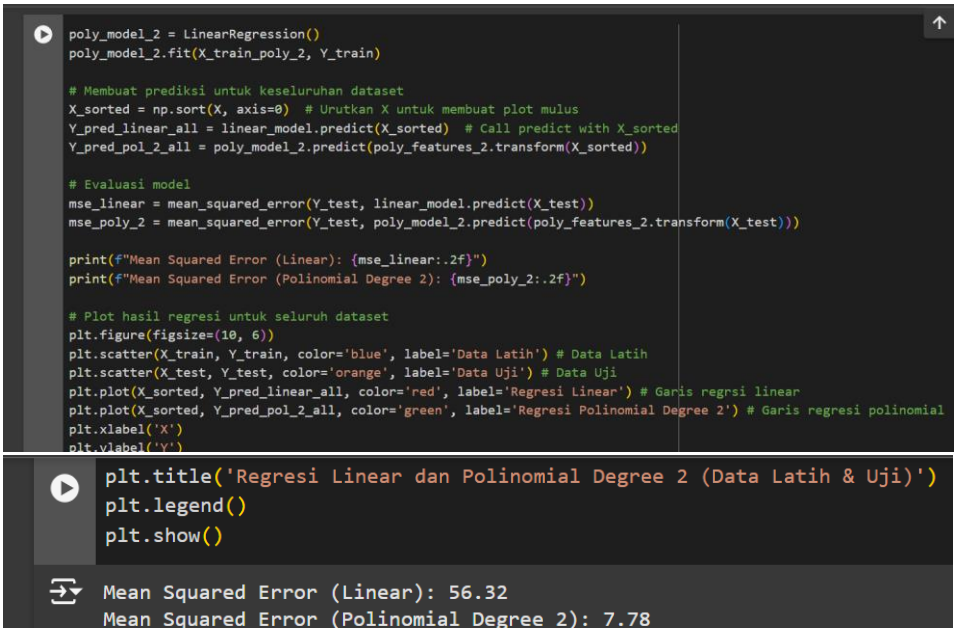
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split

# Membuat dataset (hanya nilai positif untuk X)
np.random.seed(0)
X = np.linspace(0, 6, 100).reshape(-1, 1)
Y = 2 * X**2 + 3 * X + 5 + np.random.randn(100, 1) * 3

# Membagi dataset menjadi data latih dan uji
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

# Membuat model regresi linear
linear_model = LinearRegression()
linear_model.fit(X_train, Y_train)

# Membuat model regresi polinomial derajat 2
poly_features_2 = PolynomialFeatures(degree=2)
X_train_poly_2 = poly_features_2.fit_transform(X_train)
```

- np.linspace: Membuat nilai X secara merata dari 0 hingga 6.
- train_test_split: Membagi dataset menjadi data latih (80%) dan data uji (20%).
- PolynomialFeatures: Mengubah data X agar mencakup elemen kuadratnya (untuk model polinomial).
- fit: Melatih model berdasarkan data latih.
- Membagi data menjadi dua bagian:

1. Data latih: digunakan untuk melatih model.
2. Data uji: digunakan untuk menguji akurasi model di data baru.

- Melatih dua model regresi:

1. Regresi Linear: Mencoba memodelkan data dengan garis lurus.
2. Regresi Polinomial Derajat 2: Memodelkan data menggunakan persamaan kuadratik (yang lebih sesuai dengan pola data).



```
poly_model_2 = LinearRegression()
poly_model_2.fit(X_train_poly_2, Y_train)

# Membuat prediksi untuk keseluruhan dataset
X_sorted = np.sort(X, axis=0)  # Urutkan X untuk membuat plot mulus
Y_pred_linear_all = linear_model.predict(X_sorted)  # Call predict with X_sorted
Y_pred_pol_2_all = poly_model_2.predict(poly_features_2.transform(X_sorted))

# Evaluasi model
mse_linear = mean_squared_error(Y_test, linear_model.predict(X_test))
mse_poly_2 = mean_squared_error(Y_test, poly_model_2.predict(poly_features_2.transform(X_test)))

print(f"Mean Squared Error (Linear): {mse_linear:.2f}")
print(f"Mean Squared Error (Polinomial Degree 2): {mse_poly_2:.2f}")

# Plot hasil regresi untuk seluruh dataset
plt.figure(figsize=(10, 6))
plt.scatter(X_train, Y_train, color='blue', label='Data Latih') # Data Latih
plt.scatter(X_test, Y_test, color='orange', label='Data Uji') # Data Uji
plt.plot(X_sorted, Y_pred_linear_all, color='red', label='Regresi Linear') # Garis regrsi linear
plt.plot(X_sorted, Y_pred_pol_2_all, color='green', label='Regresi Polinomial Degree 2') # Garis regresi polinomial
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Regresi Linear dan Polinomial Degree 2 (Data Latih & Uji)')
plt.legend()
plt.show()
```

```
Mean Squared Error (Linear): 56.32
Mean Squared Error (Polinomial Degree 2): 7.78
```

Mean Squared Error (MSE) untuk regresi linear lebih besar dibandingkan dengan regresi polinomial derajat 2. Artinya, model polinomial derajat 2 lebih mampu menangkap pola data yang sebenarnya, karena data tersebut mengikuti hubungan kuadratik. Garis merah (regresi linear) hanya berupa garis lurus dan tidak mampu mengikuti pola lengkung data. Garis hijau (regresi polinomial derajat 2) menyesuaikan bentuk parabola yang sesuai dengan pola data.

- Regresi Linear: Kurang cocok untuk data ini karena hanya mampu menangkap hubungan garis lurus, sehingga prediksinya kurang akurat.
- Regresi Polinomial: Lebih fleksibel dan dapat menangkap hubungan kuadratik, menghasilkan prediksi yang jauh lebih akurat.

Jika data memiliki pola non-linear seperti kuadratik, model polinomial adalah pilihan yang lebih baik daripada model linear. Model polinomial memberikan representasi yang lebih akurat terhadap hubungan antara variabel input dan output.