pol-regress-fl

March 30, 2025

Develop a program to demonstrate the working of linear regression and polynomial regression.Use Boston housing dataset for linear regression and auto MPG dataset for Polynomial regression

Polynomial Regression using Auto MPG Dataset

[33]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures, StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

import warnings
warnings.filterwarnings('ignore')
```

[34]:
```
data = sns.load_dataset('mpg')

    data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64
 2   displacement  398 non-null    float64
 3   horsepower    392 non-null    float64
 4   weight        398 non-null    int64
 5   acceleration  398 non-null    float64
 6   model_year    398 non-null    int64
 7   origin        398 non-null    object
 8   name          398 non-null    object
dtypes: float64(4), int64(3), object(2)
memory usage: 28.1+ KB
```

[35]:
```
data.nunique()
```

[35]:
```
mpg             129
cylinders         5
displacement     82
horsepower       93
weight          351
acceleration     95
model_year       13
origin            3
name            305
dtype: int64
```

[36]:
```
data.isnull().sum()
```

[36]:
```
mpg             0
cylinders       0
displacement    0
horsepower      6
weight          0
acceleration    0
model_year      0
origin          0
name            0
dtype: int64
```

[37]:
```
data['horsepower'].fillna(data['horsepower'].median(), inplace=True)
```

[38]:
```
data.isnull().sum()
```

[38]:
```
mpg             0
cylinders       0
displacement    0
horsepower      0
weight          0
acceleration    0
model_year      0
origin          0
name            0
dtype: int64
```

[39]:
```
# Define X (Feature) and y (Target)
    X = data[["horsepower"]]
    y = data["mpg"]
```

[40]:
```
# Split dataset into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[41]:
```
# Create polynomial features
degree = 2  # Change the degree of the polynomial
poly = PolynomialFeatures(degree)
X_poly_train = poly.fit_transform(X_train)
```

[42]:
```
# Fit a polynomial regression model
model = LinearRegression()
model.fit(X_poly_train, y_train)
```

[42]:
```
LinearRegression()
```

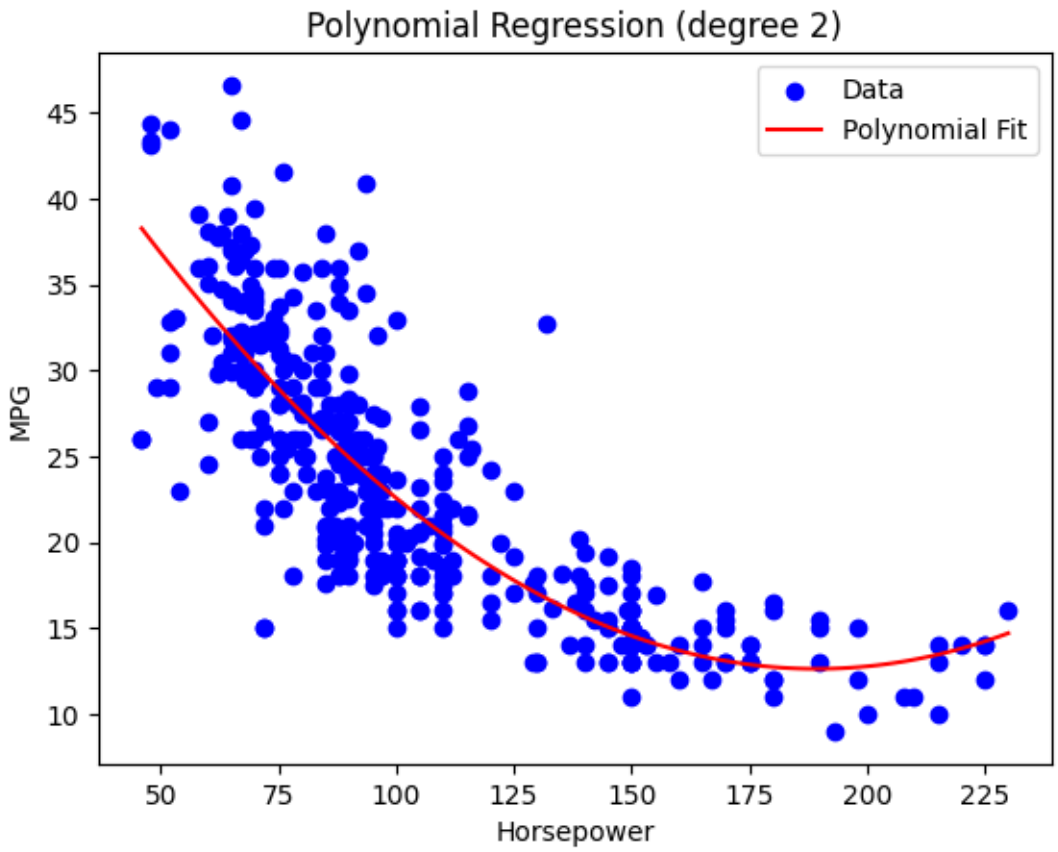
[43]:
```
# Make predictions
X_poly_test = poly.transform(X_test)
y_pred = model.predict(X_poly_test)
```

[44]:
```
# Calculate performance metrics
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

# Print metrics
print(f'Mean Squared Error: {mse}')
print(f'Root Mean Squared Error: {rmse}')
print(f'R-squared: {r2}')
```

```
Mean Squared Error: 13.941158940364115
Root Mean Squared Error: 3.7337861401483767
R-squared: 0.7407089260880471
```

[45]:
```
# Visualize the results
plt.scatter(X, y, color='blue', label='Data')
X_range = np.linspace(X.min(), X.max(), 100).reshape(-1, 1)
X_range_poly = poly.transform(X_range)
y_range_pred = model.predict(X_range_poly)
plt.plot(X_range, y_range_pred, color='red', label='Polynomial Fit')
plt.xlabel('Horsepower')
plt.ylabel('MPG')
plt.legend()
plt.title(f'Polynomial Regression (degree {degree})')
plt.show()
```


Polynomial Regression (degree 2)
Data
Polynomial Fit
MPG
Horsepower
45
40
35
30
25
20
15
10
50
75
100
125
150
175
200
225

[ ]: