Project Portofolio

Seattle Weather Regression with Linear Regression

By: Muhammad Gilang Ghazy

Linear regression

Linear regression is a method used to model the relationship between independent variables (features) and a dependent variable (target). It aims to predict the target by finding the best-fit straight line. If there are multiple features, it's called multiple linear regression.

Library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

Dataset

	date	precipitation	temp_max	temp_min	wind	weather
0	2012-01-01	0.0	12.8	5.0	4.7	drizzle
1	2012-01-02	10.9	10.6	2.8	4.5	rain
2	2012-01-03	0.8	11.7	7.2	2.3	rain
3	2012-01-04	20.3	12.2	5.6	4.7	rain
4	2012-01-05	1.3	8.9	2.8	6.1	rain
5	2012-01-06	2.5	4.4	2.2	2.2	rain
6	2012-01-07	0.0	7.2	2.8	2.3	rain
7	2012-01-08	0.0	10.0	2.8	2.0	sun
8	2012-01-09	4.3	9.4	5.0	3.4	rain
9	2012-01-10	1.0	6.1	0.6	3.4	rain
10	2012-01-11	0.0	6.1	-1.1	5.1	sun

Preproccesing

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1461 entries, 0 to 1460
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   date           1461 non-null   object 
 1   precipitation  1461 non-null   float64
 2   temp_max       1461 non-null   float64
 3   temp_min       1461 non-null   float64
 4   wind           1461 non-null   float64
 5   weather        1461 non-null   object 
dtypes: float64(4), object(2)
memory usage: 68.6+ KB
```

Description Data

```
df.describe()
```

	precipitation	temp_max	temp_min	wind
count	1461.000000	1461.000000	1461.000000	1461.000000
mean	3.029432	16.439083	8.234771	3.241136
std	6.680194	7.349758	5.023004	1.437825
min	0.000000	-1.600000	-7.100000	0.400000
25%	0.000000	10.600000	4.400000	2.200000
50%	0.000000	15.600000	8.300000	3.000000
75%	2.800000	22.200000	12.200000	4.000000
max	55.900000	35.600000	18.300000	9.500000

Variabel

```
#Memilih Fitur & Target
X = df[['precipitation', 'temp_min', 'wind']]  # Variabel Independen (Fitur)
y = df['temp_max']  #Variabel dependen (target)
```

Split data & test, training model, and predict

```
#Membagi Data menjadi Train & Test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

#Membuat & Melatih Model Machine Learning
model = LinearRegression()
model.fit(X_train, y_train)

#Memprediksi Data Test
y_pred = model.predict(X_test)
```

Model Evalution

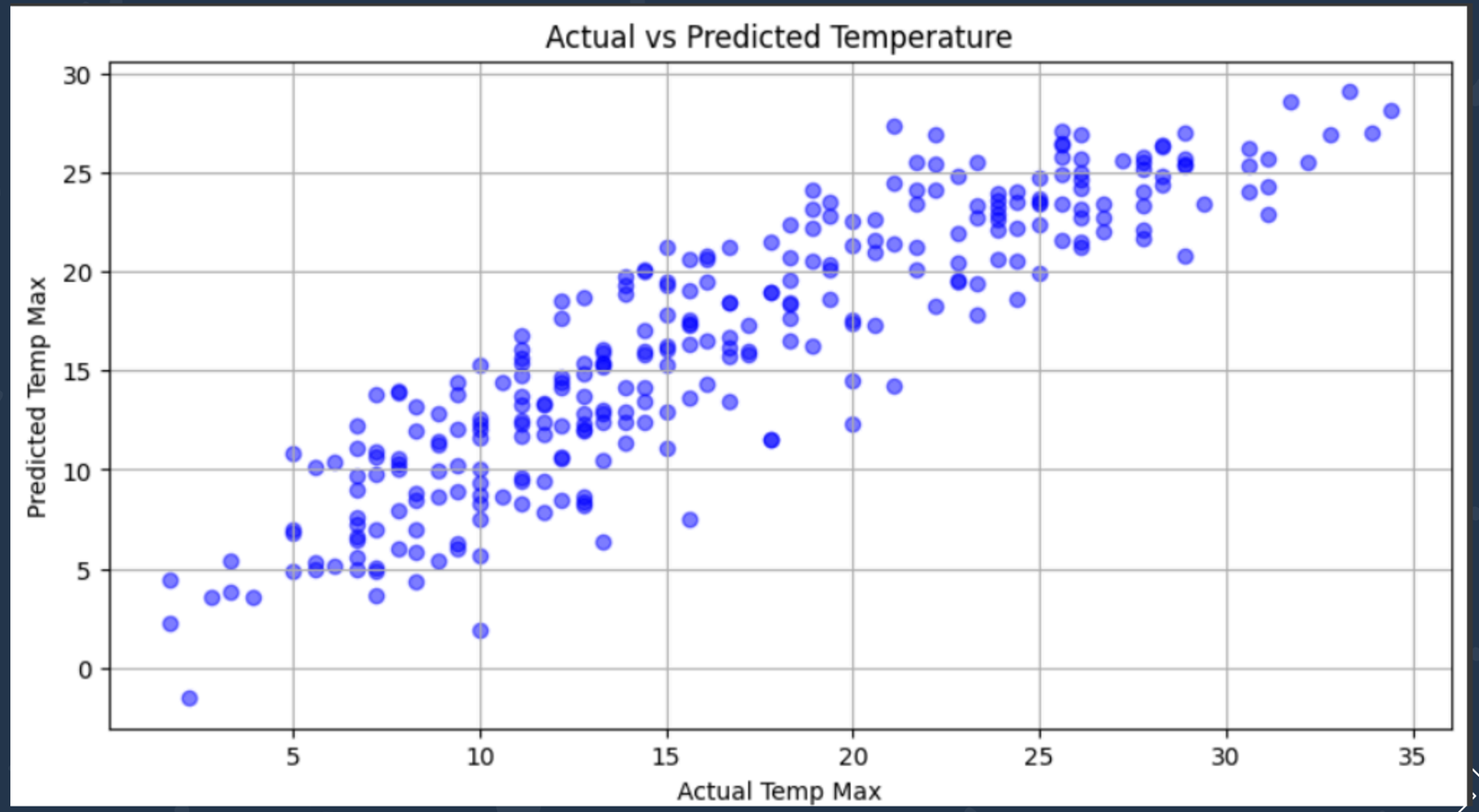
```
#Evaluasi Model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print(f"Mean Absolute Error (MAE): {mae}")
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
```

```
Mean Absolute Error (MAE): 2.7089141869157913
Mean Squared Error (MSE): 11.10351869094823
Root Mean Squared Error (RMSE): 3.3321942756910543
```

prediction results visualization

```
#Visualisasi Hasil Prediksi
plt.figure(figsize=(10, 5))
plt.scatter(y_test, y_pred, alpha=0.5, color='blue')
plt.xlabel("Actual Temp Max")
plt.ylabel("Predicted Temp Max")
plt.title("Actual vs Predicted Temperature")
plt.grid(True)
plt.show()
```

prediction results visualization



Conclusion

This Linear Regression model provides fairly good results in predicting the maximum temperature based on the given features. Although there are some prediction errors (measured by MAE, MSE, and RMSE), the model can be used as a foundation for predicting the maximum temperature, with the potential to improve accuracy through further tuning or the use of more complex models.

Thank You