In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

In [2]:

```
import pandas as pd

df = pd.read_csv("weather.csv")
df.head()
```

Out[2]:

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed
0	8.0	24.3	0.0	3.4	6.3	NW	30.0
1	14.0	26.9	3.6	4.4	9.7	ENE	39.0
2	13.7	23.4	3.6	5.8	3.3	NW	85.0
3	13.3	15.5	39.8	7.2	9.1	NW	54.0
4	7.6	16.1	2.8	5.6	10.6	SSE	50.0

5 rows × 22 columns

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 366 entries, 0 to 365
Data columns (total 22 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   MinTemp        366 non-null    float64
 1   MaxTemp        366 non-null    float64
 2   Rainfall       366 non-null    float64
 3   Evaporation    366 non-null    float64
 4   Sunshine       363 non-null    float64
 5   WindGustDir    363 non-null    object 
 6   WindGustSpeed  364 non-null    float64
 7   WindDir9am     335 non-null    object 
 8   WindDir3pm     365 non-null    object 
 9   WindSpeed9am   359 non-null    float64
 10  WindSpeed3pm   366 non-null    int64  
 11  Humidity9am    366 non-null    int64  
 12  Humidity3pm    366 non-null    int64  
 13  Pressure9am    366 non-null    float64
 14  Pressure3pm    366 non-null    float64
 15  Cloud9am       366 non-null    int64  
 16  Cloud3pm       366 non-null    int64  
 17  Temp9am        366 non-null    float64
 18  Temp3pm        366 non-null    float64
 19  RainToday      366 non-null    object 
 20  RISK_MM        366 non-null    float64
 21  RainTomorrow   366 non-null    object 
dtypes: float64(12), int64(5), object(5)
memory usage: 63.0+ KB
```

In [4]:

```
df.describe()
```

Out[4]:

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	W
count	366.000000	366.000000	366.000000	366.000000	363.000000	364.000000	
mean	7.265574	20.550273	1.428415	4.521858	7.909366	39.840659	
std	6.025800	6.690516	4.225800	2.669383	3.481517	13.059807	
min	-5.300000	7.600000	0.000000	0.200000	0.000000	13.000000	
25%	2.300000	15.025000	0.000000	2.200000	5.950000	31.000000	
50%	7.450000	19.650000	0.000000	4.200000	8.600000	39.000000	
75%	12.500000	25.500000	0.200000	6.400000	10.500000	46.000000	
max	20.900000	35.800000	39.800000	13.800000	13.600000	98.000000	

In [5]:

```
df.shape
```

Out[5]:

```
(366, 22)
```

In [6]:

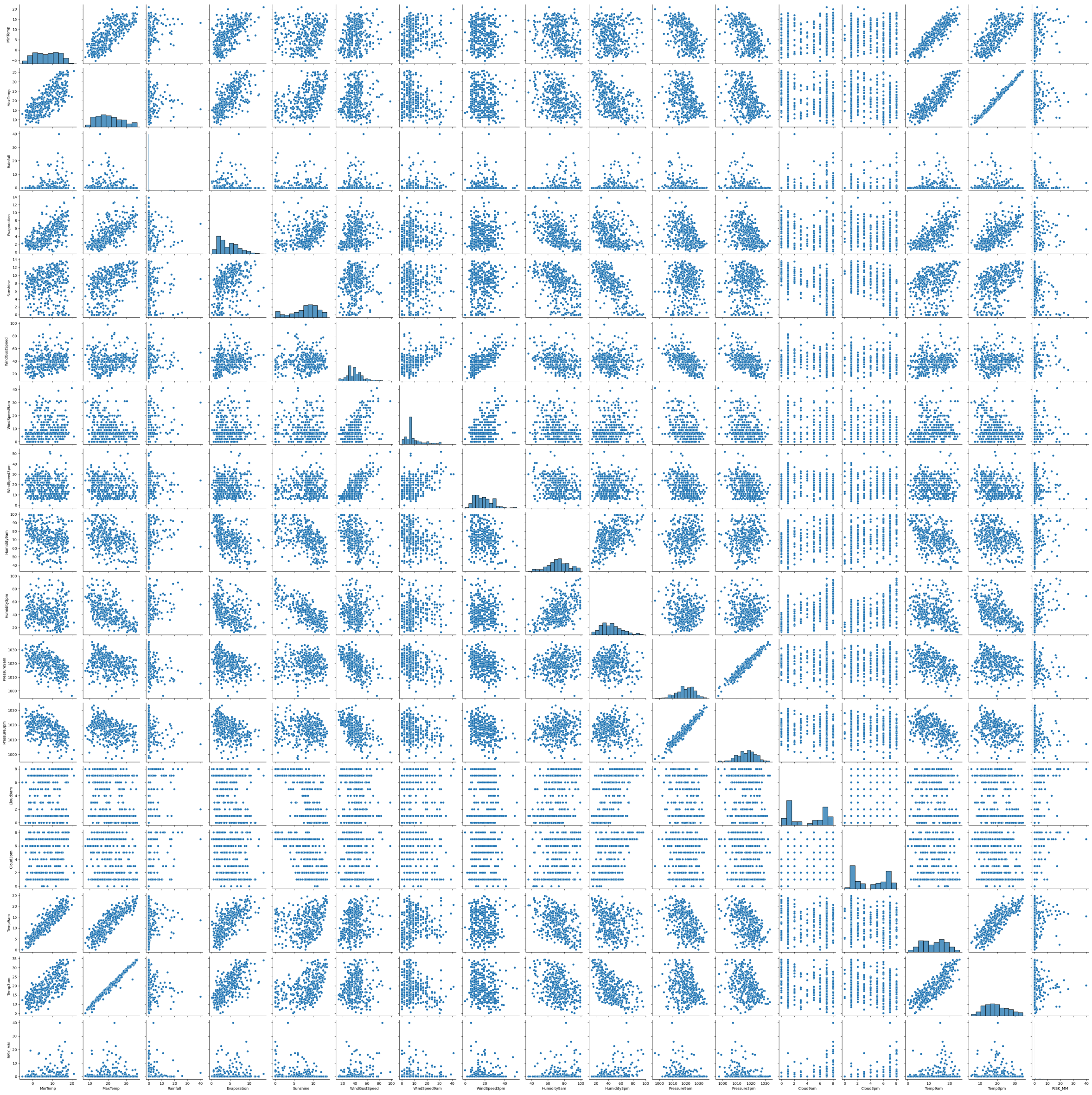
```
df.columns
```

Out[6]:

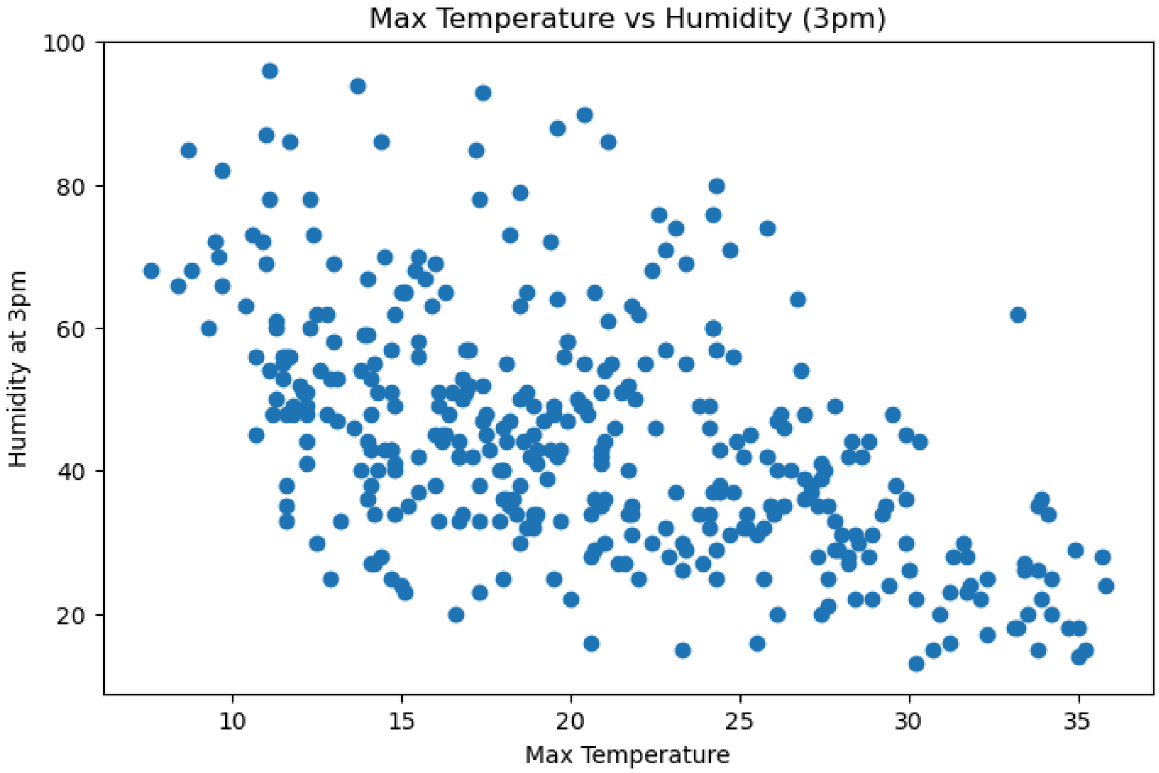
```
Index(['MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine',
       'WindGustDir', 'WindGustSpeed', 'WindDir9am', 'WindDir3pm',
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
       'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am',
       'Temp3pm', 'RainToday', 'RISK_MM', 'RainTomorrow'],
      dtype='object')
```

In [7]:

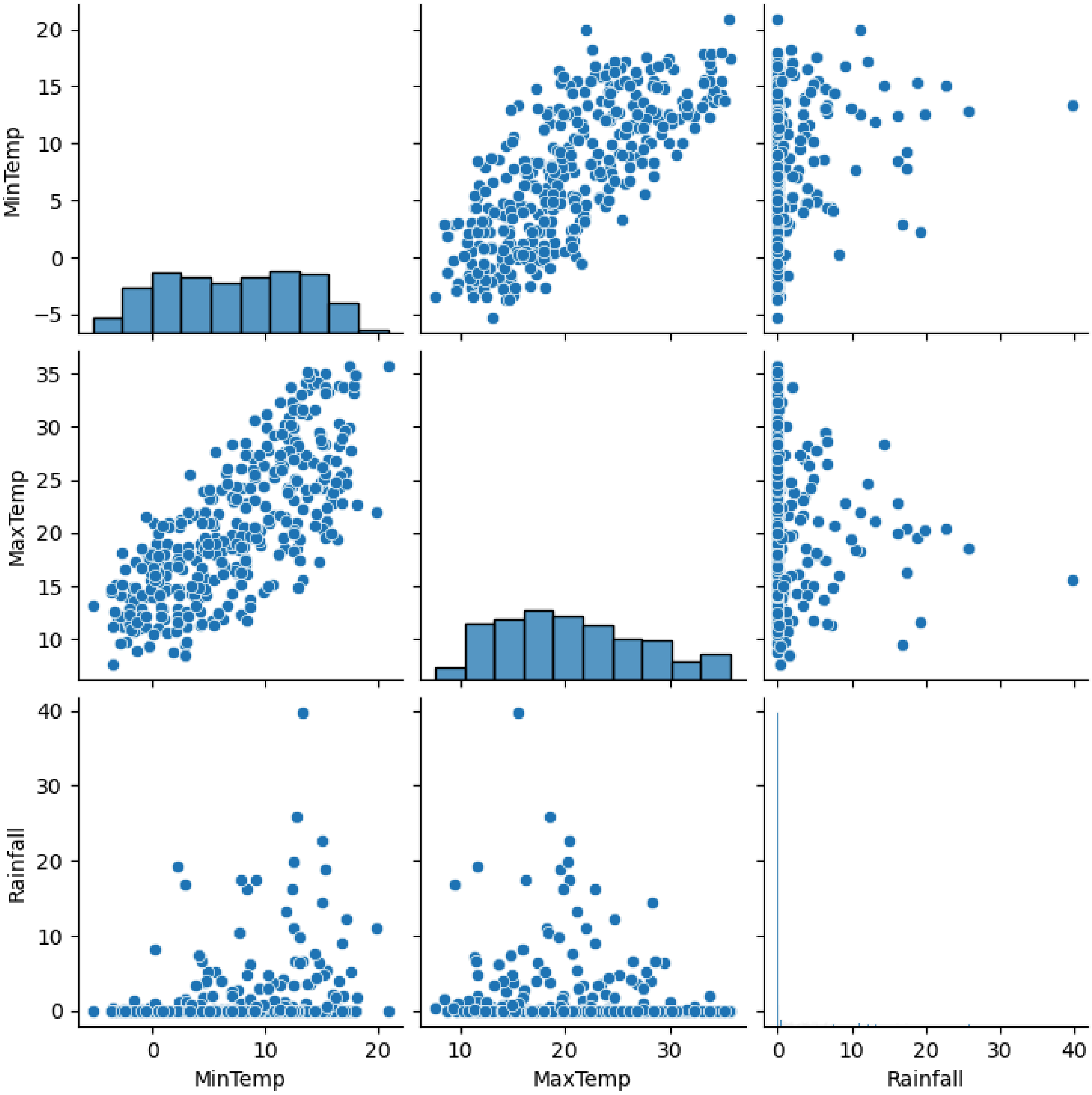
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.pairplot(df)
plt.show()
```



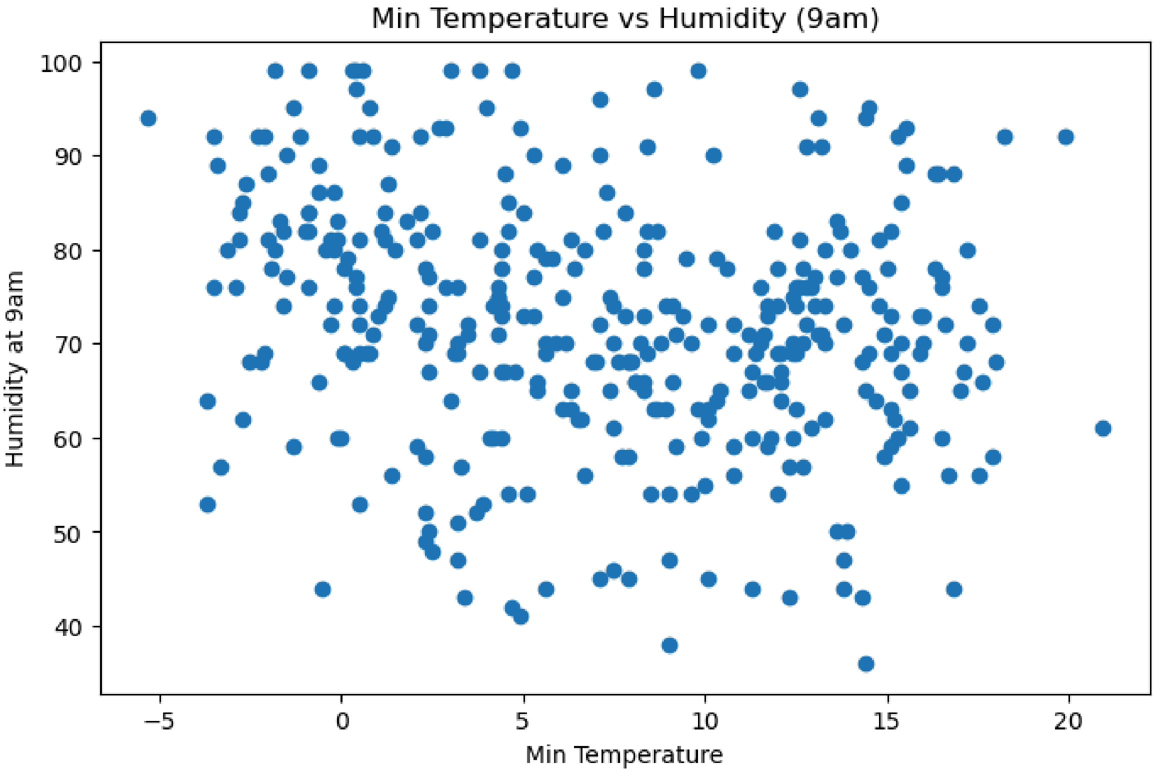
In [8]:
```
plt.figure(figsize=(8,5))
plt.scatter(df['MaxTemp'], df['Humidity3pm'])
plt.xlabel('Max Temperature')
plt.ylabel('Humidity at 3pm')
plt.title('Max Temperature vs Humidity (3pm)')
plt.show()
```



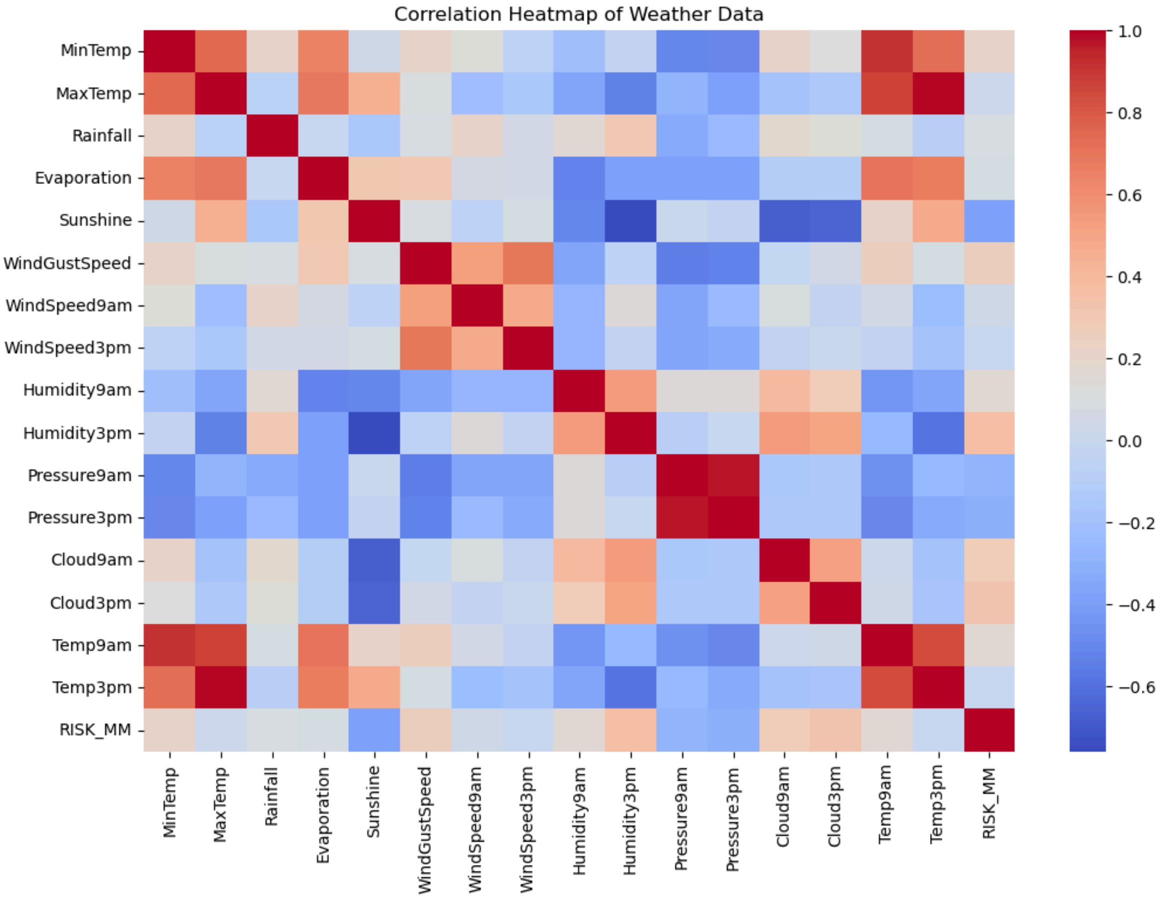
```
In [9]: sns.pairplot(df[['MinTemp', 'MaxTemp', 'Rainfall']])
        plt.show()
```



In [10]:
```
plt.figure(figsize=(8,5))
plt.scatter(df['MinTemp'], df['Humidity9am'])
plt.xlabel('Min Temperature')
plt.ylabel('Humidity at 9am')
plt.title('Min Temperature vs Humidity (9am)')
plt.show()
```



```
In [11]: numeric_df = df.select_dtypes(include='number')

         plt.figure(figsize=(12,8))
         sns.heatmap(numeric_df.corr(), annot=False, cmap="coolwarm")
         plt.title("Correlation Heatmap of Weather Data")
         plt.show()
```



In [12]:

```
# Select only numeric columns
numeric_df = df.select_dtypes(include='number')

# Separate features and target
X = numeric_df.drop('Rainfall', axis=1)
y = numeric_df['Rainfall']

X.shape, y.shape
```

Out[12]:

```
((366, 16), (366,))
```

In [13]:

```
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_state=42
)

X_train.shape, X_test.shape
```

Out[13]:

```
((292, 16), (74, 16))
```

In [15]:

```
numeric_df.isnull().sum()
```

Out[15]:

```
MinTemp          0
MaxTemp          0
Rainfall         0
Evaporation      0
Sunshine         3
WindGustSpeed    2
WindSpeed9am     7
WindSpeed3pm     0
Humidity9am      0
Humidity3pm      0
Pressure9am      0
Pressure3pm      0
Cloud9am         0
Cloud3pm         0
Temp9am          0
Temp3pm          0
RISK_MM          0
dtype: int64
```

In [16]:

```
import numpy as np
np.isinf(numeric_df).sum()
```

Out[16]:

```
MinTemp          0
MaxTemp          0
Rainfall         0
Evaporation      0
Sunshine         0
WindGustSpeed    0
WindSpeed9am     0
WindSpeed3pm     0
Humidity9am      0
Humidity3pm      0
Pressure9am      0
Pressure3pm      0
Cloud9am         0
Cloud3pm         0
Temp9am          0
Temp3pm          0
RISK_MM          0
dtype: int64
```

In [17]:

```python
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy='mean')

X_train_imputed = imputer.fit_transform(X_train)
X_test_imputed = imputer.transform(X_test)
```

In [18]:

```python
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error

model = LinearRegression()
model.fit(X_train_imputed, y_train)

y_pred = model.predict(X_test_imputed)

r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)

r2, mse
```

Out[18]:

```
(0.19041298786157834, 34.2256782862109)
```

In [21]:

```python
X = df[['MinTemp', 'MaxTemp']]
y = df['Rainfall']
```

In [22]:

```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [24]:

```python
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[24]:

```
▾ LinearRegression

▶ Parameters
```

In [25]:

```python
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error for Rainfall Prediction: {mse}')
```

```
Mean Squared Error for Rainfall Prediction: 37.0768456005826
```

In [27]:
```
df.groupby("RainToday")["MaxTemp"].mean()
```

Out[27]:
```
RainToday
No     20.756667
Yes    19.612121
Name: MaxTemp, dtype: float64
```

In [28]:
```
df.groupby("RainTomorrow")["MaxTemp"].mean()
```

Out[28]:
```
RainTomorrow
No     20.396000
Yes    21.251515
Name: MaxTemp, dtype: float64
```