```
# Step 1: Import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score


import pandas as pd
import numpy as np

# Set random seed for reproducibility
np.random.seed(42)

# Generate 500 rows of synthetic weather data
data = pd.DataFrame({
    'humidity': np.random.uniform(30, 90, 500),
    'wind_speed': np.random.uniform(0, 20, 500),
    'pressure': np.random.uniform(970, 1055, 500)
})

# Generate temperature using a linear combination of other features
# Add some noise for realism
data['temperature'] = (
    0.45 * data['humidity']
    - 0.25 * data['wind_speed']
    + 0.12 * data['pressure']
    + np.random.normal(0, 3, 500)
)

# Save the dataset to CSV
data.to_csv('weather_data_500.csv', index=False)
print("✅ File 'weather_data_500.csv' has been created.")
```

```
✅ File 'weather_data_500.csv' has been created.
```

```
# Step 2: Load the dataset
# If you just created the file in the same session:
df = pd.read_csv('weather_data_500.csv')
```

```
# Step 3: Display basic info
print("Dataset preview:")
display(df.head())
```

```
Dataset preview:
```

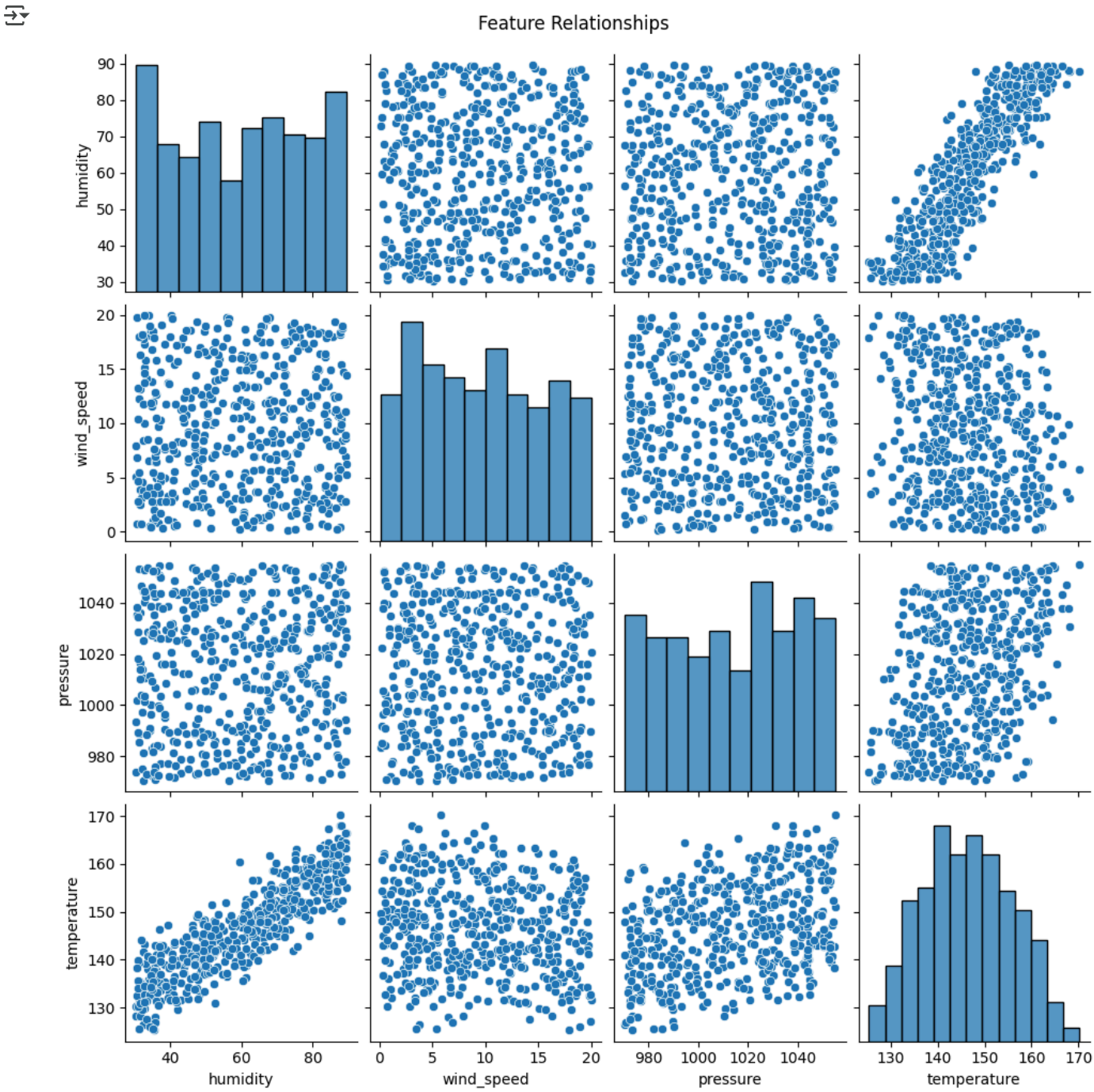
	humidity	wind_speed	pressure	temperature
0	52.472407	13.963234	985.736299	130.911913
1	87.042858	10.721927	1016.061581	165.289022
2	73.919637	6.190552	1044.200396	152.851528
3	65.919509	16.275900	1032.239115	144.527302
4	39.361118	13.694623	1038.557698	141.983482

```
print("\nDataset summary:")
print(df.describe())
```

```
Dataset summary:
          humidity  wind_speed     pressure  temperature
count  500.000000  500.000000   500.000000   500.000000
mean    59.913703    9.639028  1013.992440   146.410427
std     17.921305    5.709869    25.261399     9.603409
```

```
min     30.303695    0.092640   970.419898   125.266381
25%     44.476781    4.581985   990.504384   139.244636
50%     60.789825    9.436431  1015.877759   146.219637
75%     75.367493   14.526736  1036.074219   153.929314
max     89.577888   19.994353  1054.950167   170.249103
```

```
# Step 4: Visualize relationships
sns.pairplot(df)
plt.suptitle('Feature Relationships', y=1.02)
plt.show()
```




```
# Step 5: Split into features and target
X = df[['humidity', 'wind_speed', 'pressure']]
y = df['temperature']
```

```
# Step 6: Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Step 7: Train Linear Regression Model
model = LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression()
```

```
# Step 8: Make Predictions
y_pred = model.predict(X_test)
```

```
# Step 9: Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

```
print(f"\n📊 Model Evaluation:")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"R² Score: {r2:.2f}")
```

```
📊 Model Evaluation:
Mean Squared Error (MSE): 10.24
R² Score: 0.87
```

```
# Step 10: Plot Predictions
plt.figure(figsize=(8, 5))
plt.scatter(y_test, y_pred, color='blue', alpha=0.6)
plt.plot([y.min(), y.max()], [y.min(), y.max()], '--', color='red')
plt.xlabel("Actual Temperature")
plt.ylabel("Predicted Temperature")
plt.title("Actual vs Predicted Temperature")
plt.grid(True)
plt.show()
```