```
sarahburtenshaw@Sarahs-MacBook-Air Module 1; Phase 1 % python3 main.py
CSV METHOD: 99517
PANDAS METHOD: 99516
  row ID Location  MinTemp  MaxTemp  ...  Temp9am  Temp3pm  RainToday RainTomorrow
0   Row0   Albury     13.4     22.9  ...     16.9     21.8         No           0
1   Row1   Albury      7.4     25.1  ...     17.2     24.3         No           0
2   Row2   Albury     17.5     32.3  ...     17.8     29.7         No           0
3   Row3   Albury     14.6     29.7  ...     20.6     28.9         No           0
4   Row4   Albury      7.7     26.7  ...     16.3     25.5         No           0

[5 rows x 23 columns]
```

```
(.venv) sarahburtenshaw@Sarahs-MacBook-Air src % python3 main.py
Reading CSV, computing stats, and saving JSON…
CSV path: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-3-Phase-3 /archive/Weather Training Data.csv
Completed Processing File -->

- Numeric values processed: 1487707

- Columns summarized: MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm, Pressure9am, Pres
sure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm, RainTomorrow

- Saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-3-Phase-3 /dist/summary.json
```

```
sarahburtenshaw@Sarahs-MacBook-Air-2 src % python3 main.py
Reading CSV, computing stats, and saving JSON…
CSV path: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-4-Phase-4/archive/Weather Training Data.csv
[INFO] data_store: Wrote summary to /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-4-Phase-4/dist/summary.json
Completed Processing File -->

- Numeric values processed: 1487707

- Columns summarized: row ID, Location, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustDir, WindGustSpeed, WindDir9am, WindDir3pm, WindSpeed9am, WindSpeed3p
m, Humidity9am, Humidity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm, RainToday, RainTomorrow

- Saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-4-Phase-4/dist/summary.json

sarahburtenshaw@Sarahs-MacBook-Air-2 src %
```

```
(.venv) sarahburtenshaw@Sarahs-MacBook-Air-2 Module-5-Phase-5 % pytest -q
coverage run -m pytest -q && coverage report -m
# or
pytest --cov=src --cov-report=term-missing -q

....................                                                     [100%]
....................                                                     [100%]
Name                                Stmts   Miss  Cover   Missing
-----------------------------------------------------------------
src/__init__.py                         3      0   100%
src/core.py                            14      4    71%   18-21
src/data_fetcher.py                    52     24    54%   14, 18-19, 22-31, 40-41, 47, 53-54, 59-64
src/data_processor.py                 132     54    59%   12-15, 19-49, 78, 86-103, 122-126, 136, 139-142, 154, 172-173
src/data_store.py                      26      3    88%   32-34
src/main.py                            68     20    71%   14-16, 43-44, 66-71, 78-81, 86-89, 100
src/models.py                          20      0   100%
tests/conftest.py                       5      0   100%
tests/test_core.py                     24      0   100%
tests/test_data_fetcher.py             10      0   100%
tests/test_data_processor.py           63     22    65%   8-9, 28, 32-34, 42-49, 56-59, 77-80, 94
tests/test_data_store.py               11      0   100%
tests/test_main.py                     25      2    92%   7-8
tests/test_models.py                   13      0   100%
-----------------------------------------------------------------
TOTAL                                 466    129    72%
zsh: command not found: #
....................                                                     [100%]
=============================================================== tests coverage ================================================================
_____________________________________________ coverage: platform darwin, python 3.12.8-final-0 _____________________________________________

Name                          Stmts   Miss  Cover   Missing
-----------------------------------------------------------
src/__init__.py                   3      0   100%
src/core.py                      14      4    71%   18-21
src/data_fetcher.py              52     24    54%   14, 18-19, 22-31, 40-41, 47, 53-54, 59-64
src/data_processor.py           132     54    59%   12-15, 19-49, 78, 86-103, 122-126, 136, 139-142, 154, 172-173
src/data_store.py                26      3    88%   32-34
src/main.py                      68     20    71%   14-16, 43-44, 66-71, 78-81, 86-89, 100
src/models.py                    20      0   100%
-----------------------------------------------------------
TOTAL                           315    105    67%
(.venv) sarahburtenshaw@Sarahs-MacBook-Air-2 Module-5-Phase-5 %
```

```
sarahburtenshaw@dhcp-10-5-67-209 Module-6-Phase-6 % python3 src/main.py
Reading CSV, computing stats, and saving JSON…
CSV path: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-6-Phase-6/archive/Weather Training Data.csv
[INFO] data_store: Wrote summary to /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-6-Phase-6/dist/summary.json

============================================================
Starting Data Visualization and Pattern Analysis:
============================================================

============================================================
Weather Pattern Analysis
Using: map, filter, reduce, and lambda
============================================================

Analyzing hot vs cold temperature patterns:
Creating Hot vs Cold comparison chart:
Chart save to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-6-Phase-6/dist/how_vs_cold.png

 Analyzing rainy vs dry weather patterns:
Creating rainy vs dry comparison chart:

 Computing additional statistics using map/filter/reduce

============================================================
Analysis Complete
============================================================

 Total Records Analyzed: 99516

 Temperature Analysis:
   - Overall Average Max Temperature: 23.2°C
   - Hot Days (>25°C): 38056
   - Cold Days (<15°C): 11603
   - Very Hot Days (>30°C): 18393
   - Moderate Days (15-25°C): 49627

  Rainfall Analysis:
   - Rainy Days: 22056
   - Dry Days: 76481
   - Total Rainfall: 226055.3mm
   - Average Temperature on Rainy Days: 24.0°C
   - Average Temperature on Dry Days: 24.0°C

 Charts saved to:
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-6-Phase-6/dist/how_vs_cold.png
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-6-Phase-6/dist/rainy_vs_dry.png

============================================================
```

```
============================================================


 Visualization Complete
Completed Processing File -->

- Numeric values processed: 1487707

- Columns summarized: row ID, Location, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustDir, WindGustSpeed, WindDir9am, WindDir3pm, WindSpeed9am, W
indSpeed3pm, Humidity9am, Humidity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm, RainToday, RainTomorrow

- Saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-6-Phase-6/dist/summary.json

sarahburtenshaw@dhcp-10-5-67-209 Module-6-Phase-6 %
```

```
sarahburtenshaw@dhcp-10-5-102-107 Module-7-Phase-7 % python3 src/main.py
python3 src/main.py --sync
python3 src/main.py --parallel

============================================================
Running ASYNCHRONOUS version...
============================================================

reading CSV, computing stats, and saving JSON (ASYNC)...
CSV path: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/archive/Weather Training Data.csv

[1 of 4] Reading CSV file asynchronously...
[INFO] data_fetcher: Successfully read %d records from %s, len(records), path
✅ Read 99516 records

[2 of 4] Computing statistics...
✅ Processed 1487707 numeric values across 23 columns

[3 of 4] Saving results and creating visualizations concurrently...
          (JSON save + 2 charts happening at the same time)

============================================================
Weather Pattern Analysis
Using: map, filter, reduce, and lambda
============================================================

Creating both charts concurrently...
Creating Hot vs Cold comparison chart:
Creating rainy vs dry comparison chart:
[INFO] data_store: Wrote summary to /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/summary.json
Chart save to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/how_vs_cold.png

 Computing additional statistics using map/filter/reduce

============================================================
Analysis Complete
============================================================

 Total Records Analyzed: 99516

 Temperature Analysis:
   - Overall Average Max Temperature: 23.2°C
   - Hot Days (>25°C): 38056
   - Cold Days (<15°C): 11603
   - Very Hot Days (>30°C): 18393
   - Moderate Days (15-25°C): 49627
```

```
  Rainfall Analysis:
   - Rainy Days: 22056
   - Dry Days: 76481
   - Total Rainfall: 226055.3mm
   - Average Temperature on Rainy Days: 24.0°C
   - Average Temperature on Dry Days: 24.0°C

 Charts saved to:
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/how_vs_cold.png
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/rainy_vs_dry.png

============================================================


✅ All files saved successfully

============================================================
Async Processing Complete
============================================================
✅ Numeric values processed: 1487707
✅ Columns summarized: row ID, Location, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustDir, WindGustSpeed, WindDir9am, WindDir3pm, WindSpeed9am, WindSpeed3pm, Humidity9am, Humi
dity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm, RainToday, RainTomorrow
✅ JSON saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/summary.json
✅ Charts saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist
============================================================


============================================================
   ASYNC execution time: 2.74 seconds
============================================================


============================================================
MODE: SYNCHRONOUS
============================================================

Reading CSV, computing stats, and saving JSON…
CSV path: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/archive/Weather Training Data.csv
[INFO] data_store: Wrote summary to /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/summary.json

============================================================
Starting Data Visualization and Pattern Analysis:
============================================================
```

```
============================================================
Weather Pattern Analysis
Using: map, filter, reduce, and lambda
============================================================

Analyzing hot vs cold temperature patterns:
Creating Hot vs Cold comparison chart:
Chart save to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/hot_vs_cold.png

 Analyzing rainy vs dry weather patterns:
Creating rainy vs dry comparison chart:

 Computing additional statistics using map/filter/reduce

============================================================
Analysis Complete
============================================================

 Total Records Analyzed: 99516

 Temperature Analysis:
   - Overall Average Max Temperature: 23.2°C
   - Hot Days (>25°C): 38056
   - Cold Days (<15°C): 11603
   - Very Hot Days (>30°C): 18393
   - Moderate Days (15-25°C): 49627

  Rainfall Analysis:
   - Rainy Days: 22056
   - Dry Days: 76481
   - Total Rainfall: 226055.3mm
   - Average Temperature on Rainy Days: 20.2°C
   - Average Temperature on Dry Days: 24.0°C

 Charts saved to:
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/hot_vs_cold.png
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/rainy_vs_dry.png

============================================================
```

```
 Visualization Complete
Completed Processing File -->

- Numeric values processed: 1487707

- Columns summarized: row ID, Location, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustDir, WindGustSpeed, WindDir9am, WindDir3pm, WindSpeed9am, WindSpeed3pm, Humidity9am, Humid
ity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm, RainToday, RainTomorrow

- Saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/summary.json


============================================================
   SYNC execution time: 2.93 seconds
============================================================


============================================================
MODE: ASYNC I/O + MULTIPROCESSING
Demonstrates parallel CPU processing
============================================================

reading CSV, computing stats, and saving JSON (ASYNC + PARALLEL)...
CSV path: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/archive/Weather Training Data.csv

[1 of 4] Reading CSV file asynchronously...
[INFO] data_fetcher: Successfully read %d records from %s, len(records), path
✅ Read 99516 records

[2 of 4] Computing statistics in parallel using multiprocessing...
Multiprocessing: Using 9 CPU cores to process 23 columns
Multiprocessing: Completed processing 23 columns
✅ Processed 1487707 numeric values across 23 columns

[3 of 4] Saving results and creating visualizations concurrently...
     (JSON save + 2 charts happening at the same time)

============================================================
Weather Pattern Analysis
Using: map, filter, reduce, and lambda
============================================================
```

```
Creating both charts concurrently...
Creating Hot vs Cold comparison chart:
Creating rainy vs dry comparison chart:
[INFO] data_store: Wrote summary to /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/summary.json
Chart save to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/how_vs_cold.png

 Computing additional statistics using map/filter/reduce

============================================================
Analysis Complete
============================================================

 Total Records Analyzed: 99516

 Temperature Analysis:
   - Overall Average Max Temperature: 23.2°C
   - Hot Days (>25°C): 38056
   - Cold Days (<15°C): 11603
   - Very Hot Days (>30°C): 18393
   - Moderate Days (15-25°C): 49627

  Rainfall Analysis:
   - Rainy Days: 22056
   - Dry Days: 76481
   - Total Rainfall: 226055.3mm
   - Average Temperature on Rainy Days: 24.0°C
   - Average Temperature on Dry Days: 24.0°C

 Charts saved to:
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/how_vs_cold.png
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/rainy_vs_dry.png

============================================================


✅ All files saved successfully

============================================================
Async + Multiprocessing Complete
============================================================
✅ Numeric values processed: 1487707
✅ Columns summarized: row ID, Location, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustDir, WindGustSpeed, WindDir9am, WindDir3pm, WindSpeed9am, WindSpeed3pm, Humidity9am, Humi
dity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm, RainToday, RainTomorrow
✅ JSON saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/summary.json
✅ Charts saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist
============================================================
```

```
============================================================
Analysis Complete
============================================================

 Total Records Analyzed: 99516

 Temperature Analysis:
   - Overall Average Max Temperature: 23.2°C
   - Hot Days (>25°C): 38056
   - Cold Days (<15°C): 11603
   - Very Hot Days (>30°C): 18393
   - Moderate Days (15-25°C): 49627

  Rainfall Analysis:
   - Rainy Days: 22056
   - Dry Days: 76481
   - Total Rainfall: 226055.3mm
   - Average Temperature on Rainy Days: 24.0°C
   - Average Temperature on Dry Days: 24.0°C

 Charts saved to:
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/how_vs_cold.png
   - /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/rainy_vs_dry.png

============================================================


✅ All files saved successfully

============================================================
Async + Multiprocessing Complete
============================================================
✅ Numeric values processed: 1487707
✅ Columns summarized: row ID, Location, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustDir, WindGustSpeed, WindDir9am, WindDir3pm, WindSpeed9am, WindSpeed3pm, Humidity9am, Humi
dity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm, RainToday, RainTomorrow
✅ JSON saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist/summary.json
✅ Charts saved to: /Users/sarahburtenshaw/Documents/UVU/7.Fall Semester 2025/CS-3270/Module-7-Phase-7/dist
============================================================



============================================================
   ASYNC + PARALLEL execution time: 8.28 seconds
============================================================
```

[45] ✓ 5s

```
# Part 1
!pip install pyspark
```

```
Requirement already satisfied: pyspark in /usr/local/lib/python3.12/dist-packages (3.5.1)
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.12/dist-packages (from pyspark) (0.10.9.7)
```

[46] ✓ 0s

```
# Part 2
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, mean, median, stddev, min, max, count
import matplotlib.pyplot as plt

print("✅ Libraries imported successfully!")
```

```
✅ Libraries imported successfully!
```

[47] ✓ 0s

```
# Part 3
# Create SparkSession
spark = SparkSession.builder \
    .appName("WeatherAnalysis") \
    .master("local[*]") \
    .config("spark.driver.memory", "4g") \
    .getOrCreate()

# Display information
print("="*60)
print("✅ PySpark Environment Ready!")
print("="*60)
print(f"Spark Version: {spark.version}")
print(f"Application Name: {spark.sparkContext.appName}")
print(f"Master: {spark.sparkContext.master}")
print(f"Available Cores: {spark.sparkContext.defaultParallelism}")
print("="*60)
```

```
============================================================
✅ PySpark Environment Ready!
============================================================
Spark Version: 3.5.1
Application Name: WeatherAnalysis
Master: local[*]
Available Cores: 2
============================================================
```

[48]

```
# Part 4
```

[48] ✓ 0s

```
# Part 4
from os.path import getsize
# Check if the file exists
import os

csv_path = "/content/Weather Training Data.csv"

if os.path.exists(csv_path):
  file_size = os.path.getsize(csv_path) / (1024 * 1024) # Convert to MB
  print(f"File found: {csv_path}")
  print(f"File size: {file_size:.2f} MB")
else:
  print(f"File not found!")
  print("Please upload the file using the folder icon on the left")
```

```
File found: /content/Weather Training Data.csv
File size: 9.64 MB
```

[49] ✓ 0s

```
# Part 5
# Read CSV file into a PySpark DataFrame
print("Reading CSV file with PySpark")
print("="*60)

csv_path = "/content/Weather Training Data.csv"

# Spark reads and distributes the data
df = spark.read.csv(csv_path, header=True, inferSchema=True)

# Display information
print(f"Data loaded: successfully!")
print(f"Total rows: {df.count():,}")
print(f"Total columns: {len(df.columns)}")
print("="*60)
```

```
Reading CSV file with PySpark
============================================================
Data loaded: successfully!
Total rows: 99,516
Total columns: 23
============================================================
```

[50] ✓ 0s

```
# Part 6
# Display the schema (structure) of the data
```

[50] ✓ 0s

```
# Part 6
# Display the schema (structure) of the data
print("Data Schema (Column Names and Types):")
print("="*60)
df.printSchema()
print()
```

```
Data Schema (Column Names and Types):
============================================================
root
 |-- row ID: string (nullable = true)
 |-- Location: string (nullable = true)
 |-- MinTemp: double (nullable = true)
 |-- MaxTemp: double (nullable = true)
 |-- Rainfall: double (nullable = true)
 |-- Evaporation: double (nullable = true)
 |-- Sunshine: double (nullable = true)
 |-- WindGustDir: string (nullable = true)
 |-- WindGustSpeed: integer (nullable = true)
 |-- WindDir9am: string (nullable = true)
 |-- WindDir3pm: string (nullable = true)
 |-- WindSpeed9am: integer (nullable = true)
 |-- WindSpeed3pm: integer (nullable = true)
 |-- Humidity9am: integer (nullable = true)
 |-- Humidity3pm: integer (nullable = true)
 |-- Pressure9am: double (nullable = true)
 |-- Pressure3pm: double (nullable = true)
 |-- Cloud9am: integer (nullable = true)
 |-- Cloud3pm: integer (nullable = true)
 |-- Temp9am: double (nullable = true)
 |-- Temp3pm: double (nullable = true)
 |-- RainToday: string (nullable = true)
 |-- RainTomorrow: integer (nullable = true)
```

[51] ✓ 0s

```
# Part 7
# Show first 5 rows
print("First 5 Rows of Data:")
print("="*60)
df.show(5)
```

```
First 5 Rows of Data:
============================================================
+------+--------+-------+-------+--------+-----------+--------+-----------+-------------+----------+----------+------------+------------+-----------+-----------+-----------+-----------+
|row ID|Location|MinTemp|MaxTemp|Rainfall|Evaporation|Sunshine|WindGustDir|WindGustSpeed|WindDir9am|WindDir3pm|WindSpeed9am|WindSpeed3pm|Humidity9am|Humidity3pm|Pressure9am|Pressure3pm|(
+------+--------+-------+-------+--------+-----------+--------+-----------+-------------+----------+----------+------------+------------+-----------+-----------+-----------+-----------+-
|  Row0|  Albury|   13.4|   22.9|     0.6|       NULL|    NULL|          W|           44|         W|       WNW|          20|          24|         71|         22|     1007.7|     1007.1|
```

[51] ✓ 0s

```
# Part 7
# Show first 5 rows
print("First 5 Rows of Data:")
print("="*60)
df.show(5)
```

```
First 5 Rows of Data:
============================================================
+------+--------+-------+-------+--------+-----------+--------+----------+-------------+----------+----------+-----------+-----------+----------+----------+----------+----------+
|row ID|Location|MinTemp|MaxTemp|Rainfall|Evaporation|Sunshine|WindGustDir|WindGustSpeed|WindDir9am|WindDir3pm|WindSpeed9am|WindSpeed3pm|Humidity9am|Humidity3pm|Pressure9am|Pressure3pm|(
+------+--------+-------+-------+--------+-----------+--------+----------+-------------+----------+----------+-----------+-----------+----------+----------+----------+----------+
|  Row0|  Albury|   13.4|   22.9|     0.6|       NULL|    NULL|         W|           44|         W|       WNW|         20|         24|        71|        22|    1007.7|    1007.1|
|  Row1|  Albury|    7.4|   25.1|     0.0|       NULL|    NULL|       WNW|           44|       NNW|       WSW|          4|         22|        44|        25|    1010.6|    1007.8|
|  Row2|  Albury|   17.5|   32.3|     1.0|       NULL|    NULL|         W|           41|       ENE|        NW|          7|         20|        82|        33|    1010.8|    1006.0|
|  Row3|  Albury|   14.6|   29.7|     0.2|       NULL|    NULL|       WNW|           56|         W|         W|         19|         24|        55|        23|    1009.2|    1005.4|
|  Row4|  Albury|    7.7|   26.7|     0.0|       NULL|    NULL|         W|           35|       SSE|         W|          6|         17|        48|        19|    1013.4|    1010.1|
+------+--------+-------+-------+--------+-----------+--------+----------+-------------+----------+----------+-----------+-----------+----------+----------+----------+----------+
only showing top 5 rows
```

[52] ✓ 0s

```
# Part 8
# Display all column names
print("All Column Names:")
print("="*60)
for i, col_name in enumerate(df.columns, 1):
  print(f"{i:2d}. {col_name}")
print("="*60)
print(f"Total: {len(df.columns)} columns")
```

```
All Column Names:
============================================================
 1. row ID
 2. Location
 3. MinTemp
 4. MaxTemp
 5. Rainfall
 6. Evaporation
 7. Sunshine
 8. WindGustDir
 9. WindGustSpeed
10. WindDir9am
11. WindDir3pm
12. WindSpeed9am
13. WindSpeed3pm
14. Humidity9am
15. Humidity3pm
16. Pressure9am
17. Pressure3pm
```

[52] ✓ 0s

```
# Part 8
# Display all column names
print("All Column Names:")
print("="*60)
for i, col_name in enumerate(df.columns, 1):
  print(f"{i:2d}. {col_name}")
print("="*60)
print(f"Total: {len(df.columns)} columns")
```

```
All Column Names:
============================================================
 1. row ID
 2. Location
 3. MinTemp
 4. MaxTemp
 5. Rainfall
 6. Evaporation
 7. Sunshine
 8. WindGustDir
 9. WindGustSpeed
10. WindDir9am
11. WindDir3pm
12. WindSpeed9am
13. WindSpeed3pm
14. Humidity9am
15. Humidity3pm
16. Pressure9am
17. Pressure3pm
18. Cloud9am
19. Cloud3pm
20. Temp9am
21. Temp3pm
22. RainToday
23. RainTomorrow
============================================================
Total: 23 columns
```

[53] ✓ 0s

```
# Part 9
# Calculate statistics for all numeric columns
print("Calculating Statistics for All Numeric Columns")
print("="*60)

# Get list of numeric columns
numeric_columns = [field.name for field in df.schema.fields
                   if str(field.dataType) in ['DoubleType', 'DoubleType()', 'IntegerType', 'IntegerType()']]
print(f"Found {len(numeric_columns)} numeric columns:")
for column_name in numeric_columns:
  print(f"  - {column_name}")
```

[53] ✓ 0s

```
# Part 9
# Calculate statistics for all numeric columns
print("Calculating Statistics for All Numeric Columns")
print("="*60)

# Get list of numeric columns
numeric_columns = [field.name for field in df.schema.fields
                   if str(field.dataType) in ['DoubleType', 'DoubleType()', 'IntegerType', 'IntegerType()']]
print(f"Found {len(numeric_columns)} numeric columns:")
for column_name in numeric_columns:
  print(f"  - {column_name}")
print("="*60)
```

```
Calculating Statistics for All Numeric Columns
============================================================
Found 17 numeric columns:
  - MinTemp
  - MaxTemp
  - Rainfall
  - Evaporation
  - Sunshine
  - WindGustSpeed
  - WindSpeed9am
  - WindSpeed3pm
  - Humidity9am
  - Humidity3pm
  - Pressure9am
  - Pressure3pm
  - Cloud9am
  - Cloud3pm
  - Temp9am
  - Temp3pm
  - RainTomorrow
============================================================
```

[57]

```
# Part 10

print("\nConverting Statistics to Dictionary...")
print("="*60)

stats_rows = stats_df.collect()

stats_by_column = {}

column_names = [col_name for col_name in stats_df.columns if col_name != 'summary']
```

[57]

```
# Part 10

print("\nConverting Statistics to Dictionary...")
print("="*60)

stats_rows = stats_df.collect()

stats_by_column = {}

column_names = [col_name for col_name in stats_df.columns if col_name != 'summary']

print(f"Processing {len(column_names)} columns...")

for col_name in column_names:
    try:
        # Get statistics
        stats_by_column[col_name] = {}

        for row in stats_rows:
            stat_name = row['summary']
            stat_value = row[col_name]

            # Convert to float if possible
            if stat_value is None or stat_value == 'NULL':
                stats_by_column[col_name][stat_name] = None
            else:
                try:
                    stats_by_column[col_name][stat_name] = float(stat_value)
                except (ValueError, TypeError):
                    # This is a text column, skip it
                    stats_by_column[col_name][stat_name] = stat_value

        # Calculate range if we have min and max
        if ('min' in stats_by_column[col_name] and
            'max' in stats_by_column[col_name] and
            isinstance(stats_by_column[col_name]['min'], (int, float)) and
            isinstance(stats_by_column[col_name]['max'], (int, float))):
            min_val = stats_by_column[col_name]['min']
            max_val = stats_by_column[col_name]['max']
            stats_by_column[col_name]['data_range'] = max_val - min_val

    except Exception as e:
        print(f"⚠️  Warning: Could not process {col_name}: {e}")
        continue

# Filter to numeric columns
```

[57]

```python
# Filter to numeric columns
numeric_stats = {k: v for k, v in stats_by_column.items()
                 if 'mean' in v and isinstance(v.get('mean'), (int, float))}

print(f"✅ Successfully processed {len(stats_by_column)} total columns")
print(f"✅ Found {len(numeric_stats)} numeric columns with statistics")

# Display the statistics
print("\n" + "="*60)
print("STATISTICS SUMMARY")
print("="*60)

columns_to_show = ['MaxTemp', 'MinTemp', 'Rainfall', 'Humidity9am', 'Pressure9am']

for col_name in columns_to_show:
    if col_name in numeric_stats:
        print(f"\n{col_name}:")
        stats = numeric_stats[col_name]

        # Check if values exist
        if stats.get('count') is not None:
            print(f"  Count:      {stats['count']:,.0f}")

        if stats.get('mean') is not None:
            print(f"  Mean:       {stats['mean']:.2f}")

        if stats.get('stddev') is not None:
            print(f"  Std Dev:    {stats['stddev']:.2f}")

        if stats.get('min') is not None:
            print(f"  Min:        {stats['min']:.2f}")

        if stats.get('max') is not None:
            print(f"  Max:        {stats['max']:.2f}")

        if stats.get('data_range') is not None:
            print(f"  Range:      {stats['data_range']:.2f}")

print("\n" + "="*60)
```

```
📋 Converting Statistics to Dictionary...
============================================================
Processing 23 columns...
✅ Successfully processed 23 total columns
✅ Found 17 numeric columns with statistics
```

```
📋 Converting Statistics to Dictionary...
============================================================
Processing 23 columns...
✅ Successfully processed 23 total columns
✅ Found 17 numeric columns with statistics

============================================================
STATISTICS SUMMARY
============================================================

MaxTemp:
  Count:      99,286
  Mean:       23.22
  Std Dev:    7.12
  Min:        -4.10
  Max:        48.10
  Range:      52.20

MinTemp:
  Count:      99,073
  Mean:       12.18
  Std Dev:    6.39
  Min:        -8.50
  Max:        33.90
  Range:      42.40

Rainfall:
  Count:      98,537
  Mean:       2.35
  Std Dev:    8.49
  Min:        0.00
  Max:        371.00
  Range:      371.00

Humidity9am:
  Count:      98,283
  Mean:       68.87
  Std Dev:    19.07
  Min:        0.00
  Max:        100.00
  Range:      100.00

Pressure9am:
  Count:      89,768
  Mean:       1017.68
  Std Dev:    7.11
  Min:        980.50
  Max:        1041.00
  Range:      60.50

============================================================
```

[59] ✓ 1s

```
# Hot vs Cold Days

print("\nHot vs Cold Days Analysis")
print("="*60)

# Filter hot days (MaxTemp > 25)
hot_days_df = df.filter(col("MaxTemp") > 25.0)
hot_count = hot_days_df.count()

# Filter cold days (MaxTemp < 15)
cold_days_df = df.filter(col("MaxTemp") < 15.0)
cold_count = cold_days_df.count()

print(f"Hot days (MaxTemp > 25°C): {hot_count:,}")
print(f"Cold days (MaxTemp < 15°C): {cold_count:,}")

# Calculate average temperatures for hot and cold days
hot_avg = hot_days_df.agg(mean("MaxTemp")).collect()[0][0]
cold_avg = cold_days_df.agg(mean("MaxTemp")).collect()[0][0]

print(f"Average temp on hot days: {hot_avg:.2f}°C")
print(f"Average temp on cold days: {cold_avg:.2f}°C")
print("="*60)
```

```
Hot vs Cold Days Analysis
============================================================
Hot days (MaxTemp > 25°C): 38,056
Cold days (MaxTemp < 15°C): 11,603
Average temp on hot days: 30.59°C
Average temp on cold days: 12.41°C
============================================================
```

[60] ✓ 1s

```
# Rainy vs Dry

print("\nRainy vs Dry Days Analysis")
print("="*60)

# Filter rainy days
rainy_days_df = df.filter(col("RainToday") == "Yes")
rainy_count = rainy_days_df.count()

# Filter dry days
dry_days_df = df.filter(col("RainToday") == "No")
dry_count = dry_days_df.count()

print(f"Rainy days: {rainy_count:,}")
print(f"Dry days: {dry_count:,}")

# Calculate total rainfall
total_rainfall = df.agg({"Rainfall": "sum"}).collect()[0][0]
print(f"Total rainfall: {total_rainfall:,.2f} mm")

# Average temps on rainy vs dry days
rainy_avg_temp = rainy_days_df.agg(mean("MaxTemp")).collect()[0][0]
dry_avg_temp = dry_days_df.agg(mean("MaxTemp")).collect()[0][0]

print(f"Average MaxTemp on rainy days: {rainy_avg_temp:.2f}°C")
print(f"Average MaxTemp on dry days: {dry_avg_temp:.2f}°C")
print("="*60)
```

```
Rainy vs Dry Days Analysis
============================================================
Rainy days: 22,056
Dry days: 76,481
Total rainfall: 231,859.90 mm
Average MaxTemp on rainy days: 20.21°C
Average MaxTemp on dry days: 24.10°C
============================================================
```

[62]

```python
# Collect samples for visualization

print("\nPreparing Data")
print("="*60)

# Collect hot days sample
hot_temps_sample = hot_days_df.select("MaxTemp").limit(1000).collect()
hot_temps = [row['MaxTemp'] for row in hot_temps_sample if row['MaxTemp'] is not None]

# Collect cold days sample
cold_temps_sample = cold_days_df.select("MaxTemp").limit(1000).collect()
cold_temps = [row['MaxTemp'] for row in cold_temps_sample if row['MaxTemp'] is not None]

# Collect rainy days data
rainy_data_sample = rainy_days_df.select("Rainfall", "MaxTemp").limit(1000).collect()
rainy_rainfall = [row['Rainfall'] for row in rainy_data_sample if row['Rainfall'] is not None]
rainy_temps = [row['MaxTemp'] for row in rainy_data_sample if row['MaxTemp'] is not None]

# Collect dry days data
dry_temps_sample = dry_days_df.select("MaxTemp").limit(1000).collect()
dry_temps = [row['MaxTemp'] for row in dry_temps_sample if row['MaxTemp'] is not None]

print(f"   Collected samples for visualization:")
print(f"   Hot days sample: {len(hot_temps)} records")
print(f"   Cold days sample: {len(cold_temps)} records")
print(f"   Rainy days sample: {len(rainy_rainfall)} records")
print(f"   Dry days sample: {len(dry_temps)} records")
print("="*60)
```

```
Preparing Data
============================================================
   Collected samples for visualization:
   Hot days sample: 1000 records
   Cold days sample: 1000 records
   Rainy days sample: 1000 records
   Dry days sample: 999 records
============================================================
```

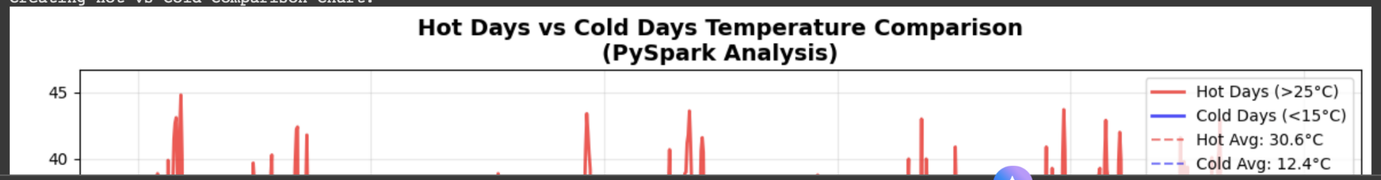
[63]

```python
# Create Hot vs Cold comparison chart
print("\nCreating Hot vs Cold Comparison Chart.")

plt.figure(figsize=(12, 6))

# Plot hot days
plt.plot(range(len(hot_temps)),
         hot_temps,
```

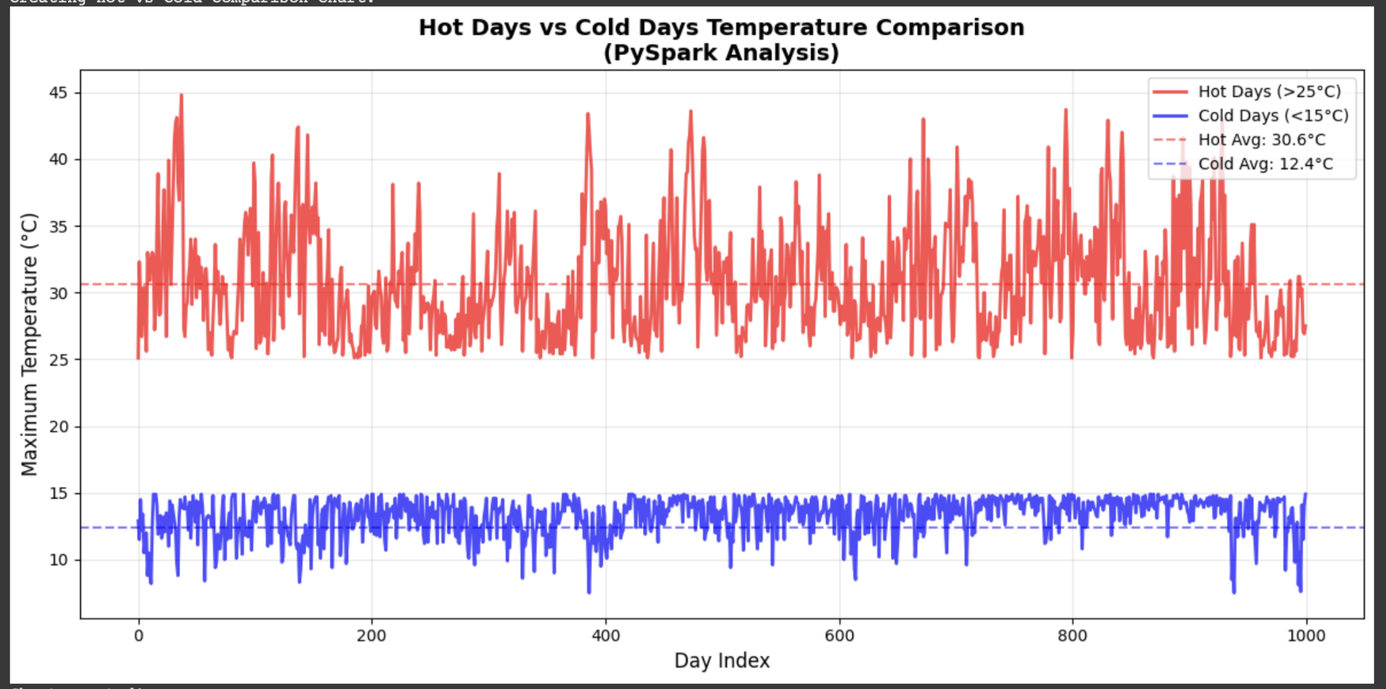
[63] ✓ 0s

```python
# Create Hot vs Cold comparison chart
print("\nCreating Hot vs Cold Comparison Chart.")

plt.figure(figsize=(12, 6))

# Plot hot days
plt.plot(range(len(hot_temps)),
         hot_temps,
         color="red",
         label=f"Hot Days (>25°C)",
         alpha=0.7,
         linewidth=2)

# Plot cold days
plt.plot(range(len(cold_temps)),
         cold_temps,
         color="blue",
         label=f"Cold Days (<15°C)",
         alpha=0.7,
         linewidth=2)

# Add average lines
plt.axhline(y=hot_avg,
            color="red",
            linestyle="--",
            label=f"Hot Avg: {hot_avg:.1f}°C",
            alpha=0.5)

plt.axhline(y=cold_avg,
            color="blue",
            linestyle="--",
            label=f"Cold Avg: {cold_avg:.1f}°C",
            alpha=0.5)

# Labels and formatting
plt.xlabel("Day Index", fontsize=12)
plt.ylabel("Maximum Temperature (°C)", fontsize=12)
plt.title("Hot Days vs Cold Days Temperature Comparison\n(PySpark Analysis)",
          fontsize=14, fontweight="bold")
plt.legend(loc="best")
plt.grid(True, alpha=0.3)
plt.tight_layout()

plt.show()
print("Chart created!")
```

```
Creating Hot vs Cold Comparison Chart.
```



```
Creating Hot vs Cold Comparison Chart.
```




```
Chart created!
```

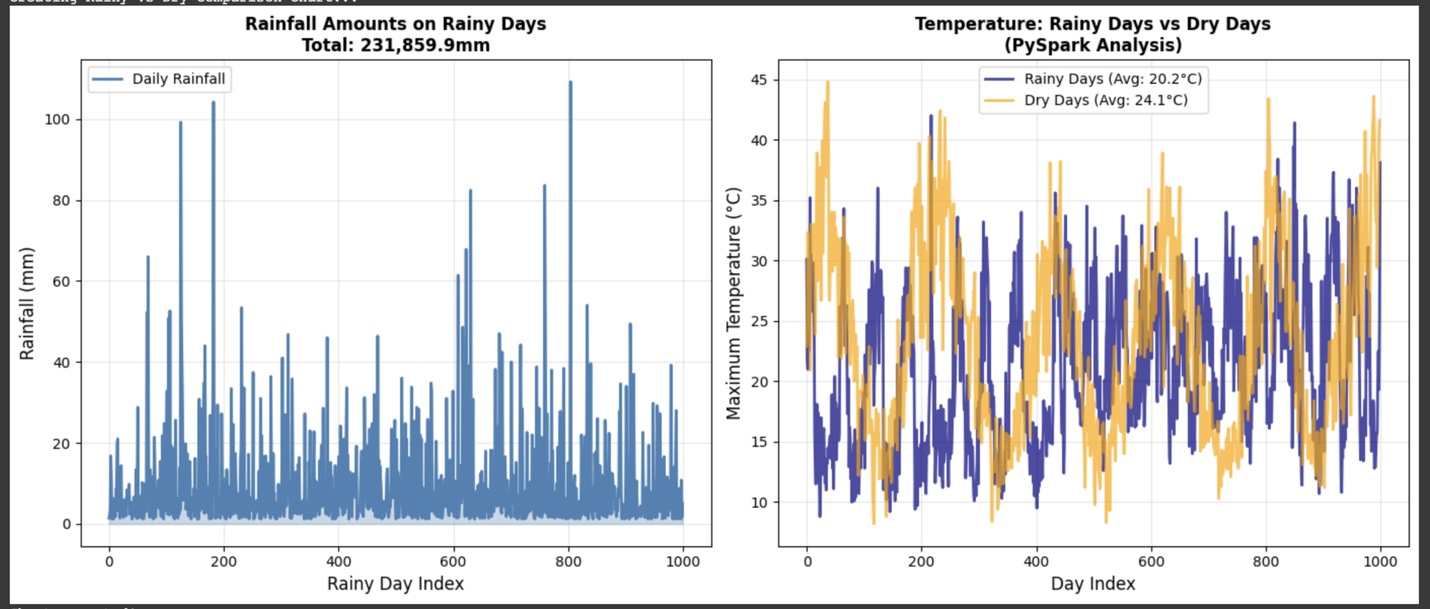
[64] ✓ 0s

```python
# Create Rainy vs Dry comparison chart
print("\nCreating Rainy vs Dry Comparison Chart...")

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 6))

# Left plot: Rainfall amounts
ax1.plot(range(len(rainy_rainfall)),
         rainy_rainfall,
         color="steelblue",
         label="Daily Rainfall",
         linewidth=2)
ax1.fill_between(range(len(rainy_rainfall)),
                 rainy_rainfall,
                 alpha=0.3,
                 color="steelblue")
ax1.set_xlabel("Rainy Day Index", fontsize=12)
ax1.set_ylabel("Rainfall (mm)", fontsize=12)
ax1.set_title(f"Rainfall Amounts on Rainy Days\nTotal: {total_rainfall:,.1f}mm",
              fontsize=12, fontweight="bold")
ax1.grid(True, alpha=0.3)
ax1.legend()
```

[64] ✓ 0s

```
ax1.set_ylabel("Rainfall (mm)", fontsize=12)
ax1.set_title(f"Rainfall Amounts on Rainy Days\nTotal: {total_rainfall:,.1f}mm",
              fontsize=12, fontweight="bold")
ax1.grid(True, alpha=0.3)
ax1.legend()

# Right plot: Temperature comparison
ax2.plot(range(len(rainy_temps)),
         rainy_temps,
         color="navy",
         label=f"Rainy Days (Avg: {rainy_avg_temp:.1f}°C)",
         alpha=0.7,
         linewidth=2)
ax2.plot(range(len(dry_temps)),
         dry_temps,
         color="orange",
         label=f"Dry Days (Avg: {dry_avg_temp:.1f}°C)",
         alpha=0.7,
         linewidth=2)
ax2.set_xlabel("Day Index", fontsize=12)
ax2.set_ylabel("Maximum Temperature (°C)", fontsize=12)
ax2.set_title("Temperature: Rainy Days vs Dry Days\n(PySpark Analysis)",
              fontsize=12, fontweight="bold")
ax2.legend()
ax2.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()
print("Charts created!")
```

```
Creating Rainy vs Dry Comparison Chart...
```




```
Charts created!
```

🌤️ Weather Analysis Dashboard

🏠 Home | 📊 Statistics | 🔍 Filter Data | 📈 Visualizations | 🤖 ML Prediction

Welcome to the Weather Analysis Application

A 3-tier web application analyzing Australian weather data

Quick Statistics

Total Records	Avg Max Temp	Avg Min Temp	Total Rainfall
1000	22.0°C	9.2°C	2216mm

View Detailed Statistics →

🌤️ Weather Analysis Dashboard

🏠 Home | 📊 Statistics | 🔍 Filter Data | 📈 Visualizations | 🤖 ML Prediction

📊 Database Statistics

📝 Total Records	🌡️ Avg Max Temp	❄️ Avg Min Temp	🌧️ Total Rainfall
1000	22.00°C	9.22°C	2216.2mm

🔥 Highest Temp	🥶 Lowest Temp
44.8°C	-2.0°C

🌤️ Weather Analysis Dashboard

🏠 Home | 📊 Statistics | 🔍 Filter Data | 📈 Visualizations | 🤖 ML Prediction

🔍 Filter Weather Data

Filter by Temperature Range

Minimum Temperature (°C):

Maximum Temperature (°C):

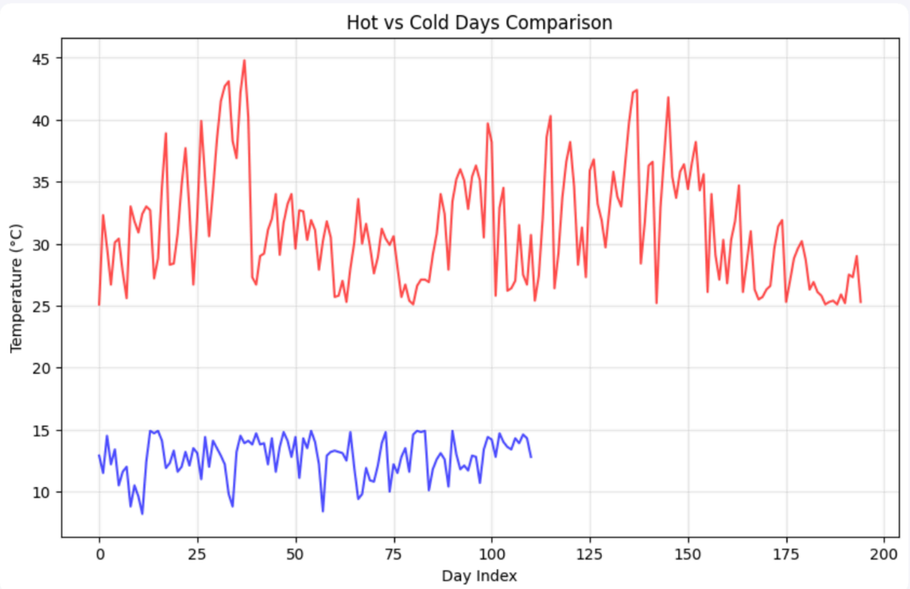
🔍 Search

🌤️ Weather Analysis Dashboard

📈 Weather Data Visualizations

Hot Days vs Cold Days Comparison

Hot Days (>25°C): 195 records
Cold Days (<15°C): 111 records



🌤️ Weather Analysis Dashboard

Machine Learning Rain Prediction

Enter Today's Weather Conditions:

Location:

Sydney

Minimum Temperature (°C):

e.g., 15.5

Maximum Temperature (°C):

e.g., 25.0

Rainfall Today (mm):

e.g., 5.0

Did it Rain Today?

No

Predict Rain Tomorrow

← Back to Home

🌤️ Weather Analysis Dashboard

Machine Learning Rain Prediction

Enter Today's Weather Conditions:

Location:

Sydney

Minimum Temperature (°C):

e.g., 15.5

Maximum Temperature (°C):

e.g., 25.0

Rainfall Today (mm):

e.g., 5.0

Did it Rain Today?

No

Predict Rain Tomorrow

☀️ No Rain Expected Tomorrow

Prediction Details:

Prediction: No

Confidence: 78.2%

Rain Probability: 21.8%

No Rain Probability: 78.2%

Input Conditions:

Location: Hobart

Min Temp: 15.0°C

Max Temp: 30.0°C

Rainfall: 5.0mm

Rain Today: No

This prediction is based on historical weather patterns and has an 80.74% accuracy rate.

← Back to Home

🌤️ Weather Analysis Dashboard

Machine Learning Rain Prediction

Enter Today's Weather Conditions:

Location:
Sydney

Minimum Temperature (°C):
e.g., 15.5

Maximum Temperature (°C):
e.g., 25.0

Rainfall Today (mm):
e.g., 5.0

Did it Rain Today?
No

Predict Rain Tomorrow

Rain Expected Tomorrow

Prediction Details:
Prediction: Yes
Confidence: 65.9%
Rain Probability: 65.9%
No Rain Probability: 34.1%

Input Conditions:
Location: Sydney
Min Temp: 10.0°C
Max Temp: 15.0°C
Rainfall: 20.0mm
Rain Today: Yes

This prediction is based on historical weather patterns and has an 80.74% accuracy rate.

← Back to Home

```
sarahburtenshaw@dhcp-10-5-78-232 Module-8-Phase-8 % python3 webapp/app.py
============================================================
Starting Weather Web Application
============================================================
Open your browser and go to: http://localhost:5000
Press Ctrl+C to stop the server
============================================================
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
============================================================
Starting Weather Web Application
============================================================
Open your browser and go to: http://localhost:5000
Press Ctrl+C to stop the server
============================================================
 * Debugger is active!
 * Debugger PIN: 852-217-445
```

```
sarahburtenshaw@Sarahs-MacBook-Air-2 Module-8-Phase-8 % python3 -m pytest -q
........................................                                 [100%]
```