Lucas Hardeman

Aiden Harrison

Mason Hartle

CIS4930

Climate Analyzer Project

The Climate Analyzer Project is our Python-based toolset for exploring weather data—specifically temperature and humidity. Our project takes in JSON data for Tallahassee then uses it to train the HumidityPredictor, the TemperaturePredictor and finds anomalies with our Anomaly function. On top of the Tallahassee data, our project also takes data from Chicago and New York City to cluster the temperatures into groups.

The main goals of the project are: To predict humidity and temperature using linear regression, to group similar weather data using clustering techniques and finally to detect weird or extreme weather patterns (anomalies) in historical data.

The Temperature and Humidity predictors are nearly identical in structure, and both aim to learn a linear relationship between climate-related input features (like day, month) and the output we care about—humidity or temperature.

**How Do These Models Work?**

Here’s a step-by-step breakdown of what these models are doing:

1. **Initialization**:  
   Each model starts with random weights that are close to zero. These weights represent how much each input feature should influence the final prediction.
2. **Training Loop**:  
   Over 10,000 iterations (a large number to ensure the model has time to learn), the model:
   * Makes a prediction using a dot product between the feature values and weights.
   * Calculates how far off that prediction is from the real value (this is the error).
   * Uses gradient descent to shift the weights in the right direction—reducing the error bit by bit.
3. **Prediction**:  
   Once trained, the model can take in new data and generate pretty good guesses for temperature or humidity based on what it’s learned.

**Why Linear Regression?**

The choice to use linear regression makes sense for a few reasons:

* It’s simple, fast, and easy to interpret.
* In many real-world datasets, especially climate-related ones, there’s at least *some* linear correlation between features (like time of year) and weather.

While more complex models like decision trees or neural networks could offer better accuracy, linear regression is a solid baseline and easier to understand and work with.

**Clustering Function**

**What’s This Function Doing?**

The Clustering function in the code is basically our homegrown version of the popular K-Means clustering algorithm. Its job is to find natural groupings within the dataset—for example, grouping similar weather days or climate conditions together.

**How It Works:**

1. **Pick Random Starting Points**:  
   The algorithm starts by randomly selecting a few data points to act as the initial “centers” of clusters.
2. **Assign Points to Nearest Cluster**:  
   It loops through the dataset, measuring the distance from each data point to the current cluster centers. Each point is assigned to the nearest one.
3. **Update Centers**:  
   Once all points are assigned, the algorithm recalculates each cluster center by averaging the positions of the points within it.
4. **Repeat Until Stable**:  
   The process continues until the cluster centers stop changing significantly (or until the loop hits the max number of iterations).

**Anomaly Function**

**What’s the Goal Here?**

The detect anomalies function is designed to find days where the temperature is *way* out of line with what’s typical for that month.

**How It Works:**

1. **Add a Date Column**:  
   Using the year, month, and day columns, the function creates a proper datetime column for each record.
2. **Calculate Monthly Averages**:  
   It then computes the average temperature for each month across the dataset.
3. **Compare Each Day to Its Month's Average**:  
   For every record, the function checks how different the day’s temperature is from the average for that month.
4. **Flag Outliers**:  
   If the difference is greater than a given threshold (default is 15 degrees), that day is considered an anomaly.

**Why This Matters**

Climate data often contains outliers—either due to natural weather variation or data collection errors. By flagging anomalies:

* You can highlight unusual weather events.
* You can clean your dataset by filtering out these values.
* You can study the impact of events like heatwaves or cold snaps.

All in all, the Climate Analyzer Project is a clean and educational example of how machine learning can be applied to real-world weather data. This project focused on the below topics and we implemented them to get our models.

* Linear regression
* Gradient descent
* Clustering logic
* Outlier detection

**Visualization**

For our project, we implemented a way to visualize three of our algorithms through graphs using the Matplotlib library. Our project can show the following which are derived from the data we get by running the algorithms we created:

* Humidity Prediction Line Graph
  + Our machine learning algorithm will predict what the humidity will be for a given month and compare it to the actual data where you can see the relations between the two graphs.
* Cluster Grouping Scatter Plot
  + Our cluster scatter plot function will take the cluster groupings from the algorithm and represent which months from what location fall within each cluster based on their monthly average temperature.
* Anomaly Bar Graph
  + This bar graph will show any anomalies detected by our anomaly algorithm, either blue and below zero or red and above zero, based on how far the temperature for those days differed from the monthly average.