Michael Maytin – Ryan Auclair – Reid McClellan

Shifat Mithila

CIS4930 – Python Programming

April 11, 2025

Project Report

**Introduction:**

This project, *Climate Change Impact Analyzer*, is a Python-based software application designed to collect, analyze, and visualize weather data for cities around the world. The goal of the project is to examine recent climate conditions, detect anomalous temperature patterns, and predict future temperature trends using historical weather data. Our tool interacts with the Weather API to retrieve real-time and historical weather information, processes and stores the data in structured CSV files, and leverages statistical techniques to identify anomalies and forecast future conditions. In addition, users can compare multiple cities based on their average temperature and anomaly frequency using clustering algorithms and parallel coordinate visualizations.

The program was built with a modular structure, separating responsibilities into distinct functions for data collection, anomaly detection, prediction, and plotting. Anomalies are flagged based on deviations of more than ±3°C from historical averages for the same calendar day across ten prior years. Future temperatures are estimated by averaging those same historical values. To allow for visual inspection, the project includes several Matplotlib and Seaborn-powered plots that clearly represent normal, anomalous, and predicted temperatures over time. The final tool includes a command-line interface for user interaction and was developed following Python best practices for readability, reusability, and scalability. Overall, this analyzer serves as a foundation for studying localized climate trends and communicating the potential impacts of global climate variability through data.

**Algorithm and Process Explanation:**

The Climate Change Impact Analyzer was developed with a modular architecture to streamline the processes of data collection, processing, anomaly detection, prediction, and visualization. The project begins by retrieving climate data from the WeatherAPI, which provides both real-time and historical weather information. Users can input a city name and a date range, and the system will query the API for daily weather statistics including average temperature, air quality index, sunrise and sunset times, and general conditions. To maintain a clean and reusable data structure, the retrieved data is stored in CSV format within a designated folder, with filenames sanitized to ensure compatibility across file systems. Although the data returned by the API is already structured in JSON format, minor preprocessing such as filename sanitization and header initialization is performed to prepare it for analysis.

Anomaly detection is performed using a simple but effective statistical approach. For each day within the user-defined date range, the system retrieves historical temperature data for that same calendar day over the past ten years. Once this data is gathered, it calculates the historical average temperature. If the current year’s temperature on that day deviates by more than three degrees Celsius from this average, it is marked as an anomaly. This threshold was chosen because it provides a meaningful indication of outlier conditions while accounting for normal weather variability. Anomalies are clearly marked in the resulting dataset and visualizations, making it easy to identify patterns of extreme deviation over time.

For temperature forecasting, the analyzer uses a historical averaging technique to estimate the temperatures for the seven days following the user’s selected date range. The system looks at each of those future days and gathers weather data for the same calendar days over the previous ten years. By averaging these historical temperatures, the tool generates a predicted temperature for each upcoming day. While this method does not involve complex machine learning models, it effectively captures the seasonal trends in climate data and offers reasonably accurate short-term predictions. The approach also ensures transparency and simplicity, which is valuable for users seeking to understand how the forecast is generated.

The project also includes a comparison feature that analyzes multiple cities. For each city, it calculates the average temperature and the number of anomalies over the selected period. These values are normalized to a common scale, and then the K-Means clustering algorithm is applied to group cities with similar climate patterns. This allows users to explore regional similarities or differences in temperature trends and anomaly frequency. The results of this clustering are visualized using parallel coordinates plots, which display multidimensional data in a way that clearly illustrates how each city compares across metrics.

Visualization plays a central role in this project. The tool generates several types of plots using Matplotlib and Seaborn, including temperature trend lines, scatter plots highlighting anomalies, overlaid actual versus predicted temperature charts, and multi-city comparison graphs using parallel coordinates. These visualizations not only enhance the interpretability of the data but also provide users with a compelling and intuitive way to understand the local and comparative impacts of climate change.

**Algorithm Justification:**

The algorithms used in the Climate Change Impact Analyzer were chosen based on their simplicity, effectiveness, and interpretability, especially given the nature of climate data and the goals of this project. For anomaly detection, we opted for a historical mean-based approach rather than a complex statistical or machine learning method. This decision was made because temperature data tends to exhibit consistent seasonal patterns year over year, and significant deviations from these patterns can often be detected with a straightforward comparison to historical averages. By flagging days where the temperature deviates more than three degrees Celsius from the ten-year average for that specific calendar day, the algorithm is able to highlight meaningful anomalies without overcomplicating the process. This threshold was selected as it balances sensitivity and specificity—it’s large enough to avoid flagging natural fluctuations, yet small enough to capture genuine outliers.

For temperature prediction, we deliberately chose a model-free, historical averaging technique. Instead of training a predictive model such as linear regression or time-series forecasting algorithms, we used a simple method of averaging historical temperatures for the corresponding days over the past ten years. This approach was favored because of its transparency and ease of implementation, making it highly accessible to users and developers alike. It also performs reasonably well for short-term forecasts in climate datasets where long-term seasonal patterns are strong and relatively stable. More advanced models may have added complexity without significantly improving accuracy in this specific use case, especially given the constraints of limited real-time features and noisy environmental data.

In the comparison and clustering of cities, the K-Means algorithm was selected for its efficiency and clarity in grouping cities based on two intuitive features: average temperature and anomaly count. These features were normalized to ensure fair comparison, and K-Means provided a fast and interpretable way to assign each city to a cluster. While there are more advanced clustering algorithms available, such as DBSCAN or hierarchical clustering, K-Means was sufficient given the low dimensionality and clean structure of our input data. Additionally, it pairs well with parallel coordinates plots, which were used to visualize the cluster groupings and facilitate easy comparison across multiple cities.

Overall, our choices prioritized clarity, performance, and alignment with the project’s educational and analytical goals. Each selected algorithm serves its purpose while maintaining a balance between functionality and user interpretability. The modular design of the tool also allows for future extensions, such as replacing or enhancing these algorithms with more complex models if higher accuracy or deeper analysis is desired.

**Conclusion:**

The Climate Change Impact Analyzer demonstrates the power of modular, data-driven software to reveal patterns in local and regional climate behavior. Through the integration of historical data, API-based collection, and intuitive statistical algorithms, the tool offers users a practical way to detect temperature anomalies, generate short-term forecasts, and compare climate trends across different cities. The decision to prioritize interpretable methods—such as historical averaging and threshold-based anomaly detection—allowed for accurate insights while maintaining a user-friendly and transparent analytical process. By coupling these insights with clear visualizations and interactive prompts, the tool not only highlights the impact of climate variability but also provides a foundation for further exploration. Whether used as an educational resource, a prototype for larger climate platforms, or a standalone research tool, this project successfully meets its goal of translating raw weather data into actionable understanding. Future extensions could involve integrating more complex machine learning models, refining the anomaly thresholds with statistical baselines, or expanding the visualization capabilities to include real-time dashboards or geospatial maps. Nonetheless, the current implementation offers a robust and scalable framework for ongoing climate analysis, built entirely with open tools and Python best practices.