Predicting Temperature Using ETS Model: A Multi-Language Platform

MAJOR PROJECT REPORT

Semester – 1

Four Year Undergraduate Program

(Design your degree)

SUBMITTED TO:

UNIVERSITY OF JAMMU, JAMMU

GROUP: ARYABHATA

Group members	Roll no.
Manan Sharma	DYD-24-20
Ishu Bharati Pandit	DYD-24-14
Dhwani Gupta	DYD-24-09
Rishita Gupta	DYD-24-28
Peehar Singh Charak	DYD-24-27

Under the Mentorship of:

Prof. K.S. Charak
Department of Mathematics
University of Jammu

Dr. Jatinder Manhas
SIIEDC
University of Jammu

Dr. Sunil Kumar
Department of Statistics
University of Jammu

SUBMITTED ON: 29th January, 2025

CERTIFICATE

The report titled "*Predicting Temperature Using ETS Model: A Multi-Language Platform"* was completed by group Aryabhata comprised of Manan Sharma, Ishu Bharati Pandit, Dhwani Gupta, Rishita Gupta, and Peehar Singh Charak, as a major project for Semester I. It was conducted under the guidance of Prof. K.S. Charak, Dr. Jatinder Manhas, Dr. Sunil Kumar for the partial fulfillment of the Design Your Degree, Four Year Undergraduate Program at the University of Jammu, Jammu. This project report is original and has not been submitted elsewhere for any academic recognition.

Students:

1. Manan Sharma
2. Ishu Bharati Pandit
3. Dhwani Gupta
4. Rishita Gupta
5. Peehar Singh Charak

Signature of the mentors:

Prof. K.S. Charak Dr. Jatinder Manhas Dr. Sunil Kumar

Prof. Alka Sharma

Director SIIDEC, University of Jammu

ACKNOWLEDGEMENT

We extend our sincere gratitude to Prof. K.S. Charak for his invaluable guidance and support in mathematics. His insightful feedback and patient explanations were crucial in shaping the project's mathematical foundations. We especially appreciate his willingness to discuss complex concepts, which were essential for the success of this work.

We are also grateful to Dr. Jatinder Manhas for his contributions to the IT aspects of the project. His expertise in software development and data management provided practical solutions to technical challenges, ensuring the smooth progress of the project.

Our heartfelt thanks also go to Dr. Sunil Kumar for his expert guidance in statistics. His deep knowledge of statistical methods and ability to clarify complex concepts significantly contributed to the validity of our findings. His input on data interpretation was crucial to our work.

We deeply appreciate Prof. Alka Sharma, Head of the Department, for her unwavering support, encouragement, and leadership throughout this project. Her vision and guidance were vital in fostering a supportive research environment.

Lastly, we acknowledge our classmates for their collaborative spirit, insightful discussions, and assistance. Their contributions helped overcome challenges and create a stimulating learning environment.

Thank you to everyone who contributed to the success of this project.

ABSTRACT

This project showcases an application of the ETS (Exponential Smoothing) model for temperature prediction, developed with the help of multi-language implementation based on Python and C Language. This project shows the complete method of time-series prediction by combining high-level statistical models with computational means for discrete derivative analysis. Historical data of daily average temperature of Jammu City was preprocessed in order to correct missing values for good prediction. The ETS model was set to capture the seasonal patterns, trend, and error components of the data, so it could provide more reliable predictions for next 45 days of daily average temperature.

In order to check daily average temperature variations and get a better understanding of short-term analysis, the discrete derivatives were calculated using a code written and designed in C Language. To make forecasting, calculating derivatives, and viewing the results easy, a command-line interface was created in C Language. This interface combines the data analytics libraries of Python such as statsmodels with the system level integration capabilities of C Language.

This project shows the multi-language implementations and efficient way of prediction in dealing with complex tasks and offers a robust workflow for extending predictive analytics to other areas. The results show that combining statistical models and computational tools can deliver good details, which makes this approach a good contribution to fields that requires time-series prediction and analysis.

CONTENTS

S.no	Contents	Pg. No
1	**INTRODUCTION**	**7**
1.1	Background	
1.2	Objectives	
1.3	Scope and Significance	
2	**LITERATURE REVIEW**	**8 – 9**
2.1	Introduction	
2.2	Short-Term Forecasting Capability	
2.3	Scalability and Regional Adaptability	
2.4	Simplicity and Computational Efficiency	
2.5	Seasonal Management	
2.6	Hybrid Model Potential	
2.7	Use of ETS Beyond Temperature Forecasting	
2.8	Limitations Along the Way	
3	**TOOLS AND TECHNOLOGY**	**10 - 12**
3.1	Introduction	
3.2	C Language	
3.3	Python	
3.4	IBM SPSS	
3.5	Microsoft Excel	
3.6	Conclusion	
4	**INTRODUCTION TO DATA**	**13 - 18**
4.1	Dataset Description	
4.2	Descriptive Statistics	
4.3	Insights Gained	
5	**METHODOLOGY**	**19 – 28**
5.1	ARIMA Model	
5.2	ARIMA's Parameters	
5.3	SARIMA Model	
5.4	Conclusion	
5.5	ETS Model	
5.6	Discrete Derivative	
5.7	Mathematical Formulation	
5.8	Application in Weather Analysis	
5.9	Conclusion	
5.10	Experimental Setup	

6	**RESULTS AND DISCUSSION**	**29 – 37**
6.1	Introduction	
6.2	The Interface and Outputs	
6.3	Code Implementation	
7	**FUTURE SCOPE AND RECOMMENDATIONS**	**38 – 42**
7.1	Future Scope	
7.2	Recommendations	
7.3	Proposed Hardware Design	
8	**REFERENCES**	**43 - 44**

Chapter 1

INTRODUCTION

1.1 Background

Temperature prediction is one of the important methods of modern meteorology to different applications in a range from agriculture to various fields. A good and accurate prediction is important because it reduces the impact of uncertainties in environmental conditions. Traditional statistical methods have also been effective but are often incapable in capturing seasonal and trend components presented in temperature data. however, the Exponential Smoothing (ETS), offer a new path for dealing with these problems.

The ETS model is especially used for time-series data with both seasonal and trend components. This project will use this model to predict daily average temperatures for the next 45 days. Besides, the multi-language implementation approach is shown by using both Python and C language, it helps to maintain the balance between computational performance and more accurate results.

1.2 Objectives

1. To use ETS Model to predict average temperature data for next 45 days.
2. To check daily variations in predicted temperature using discrete derivative.
3. To develop a Command Line (CLI) Interface that uses Python's statistical libraries with C's system level integration capabilities.

1.3 Scope and Significance

In addition to the ETS model, this project shows how important is the multi-language implementation is in the field of scientific computing. Because of its good and robust design, this code can easily read past daily average temperature data and predict future values. Because it is modular and can be modified according to our needs, it offers a solid framework for future improvements and further uses cases in related fields. By combining the statistical models with effective computing tools, this work has expanded the field of predictive analysis and shows the concept of hybrid programming that can be used in real-world problem solving.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

Temperature forecasting is highly demanding as it requires analysis in terms of patterns and seasonal variability aside from minor fluctuations within the data over Small Period of time. In this study, we have chosen the Exponential Smoothing Model, commonly known as ETS, to project and predict temperature trend for the succeeding 45 days. ETS Model stood out because of its simplicity, robust capability to handle the seasonal trends, and scalability.

In addition, for this project we used the Discrete Derivative to show prediction trends. We will describe how prior studies have shaped and made our work easier, along with the challenges that were faced in choosing a model as well as problems faced throughout this project.

2.2 Short-Term Forecasting Capability

The ETS model has been proven efficient for short-range weather forecasting. A study titled *"An In-Depth Look at Rising Temperatures: Forecasting with Advanced Time Series Models in Major US Regions"* by Kinast & Fokoué (2024) [4] demonstrated its accuracy in predicting meteorological patterns across diverse regions. The findings validated our confidence in the ETS model's ability to handle temperature fluctuations, aligning with our goal of providing reliable 45-day forecasts, even under adverse conditions.

2.3 Scalability and Regional Adaptability

Temperature trends vary by location, necessitating a model that accommodates diverse climatic conditions. The study *"An In-Depth Look at Rising Temperatures: Forecasting with Advanced Time Series Models in Major US Regions"* (Kinast & Fokoué, 2024) [4] illustrates how ETS scales across various climatic zones. This adaptability reassured us that ETS would perform well over a 45-day period, even when dealing with seasonal complexities.

2.4 Simplicity and Computational Efficiency

ETS is computationally efficient compared to complex forecasting models like ARIMA. The paper *"Climate Impact on Evapotranspiration in the Yellow River Basin: Interpretable Forecasting with Advanced Time Series Models and Explainable AI"* by Khan et al. (2025) [3] demonstrates that ETS performs comparably to ARIMA while being easier to implement. Although ARIMA is mathematically rigorous, optimizing it for seasonal variations proved challenging for us.

2.5 Seasonal Management

Seasonal patterns in some regions are complex, such as varying summer lengths or sudden climate shifts. *"Forecasting Temperature Data with Complex Seasonality Using Time Series Methods"* by Elseidi (2023) [2] demonstrates ETS's ability to handle complex seasonality. This feature was crucial for our study, ensuring robust performance under unpredictable conditions.

2.6 Hybrid Model Potential

Although our study focuses on ETS, better accuracy may be achieved by integrating ETS with machine learning techniques. The paper *"A Hybrid ETS–ANN Model for Time Series Forecasting"* by Panigrahi & Behera (2017) [5] explores the integration of ETS with artificial neural networks (ANN). While we did not incorporate machine learning, this study inspired us to explore hybrid models in future research.

2.7 Use of ETS Beyond Temperature Forecasting

ETS models have been applied to other environmental forecasting tasks, such as predicting water level changes. Studies like *"Using ARIMA and ETS Models for Forecasting Water Level Changes for Sustainable Environmental Management"* by Agaj et al. (2024) [1] illustrate ETS's versatility.

2.8 Limitations and Challenges

Before using ETS, we used the ARIMA and SARIMA models, both of which had many limitations. We realized that ARIMA has an issue with forecasting as we don't have more variables in the dataset other than daily average temperature.

SARIMA, on the other hand, is more complex and it was computational heavy. These problems meant there was a need for an efficient model that is good and not complicated to implement in our code.

The problem during the selection of the suitable model highlighted the fact that choosing the correct approach for the task depends on the nature of the requirements of the project.

That is why ETS was the good choice that satisfied the needs for the specific project because it overcame all those problems that we are facing.

Chapter 3

TOOLS AND TECHNOLOGY

3.1 Introduction

The successful completion of this project was made possible by utilizing a carefully chosen combination of tools and technologies. Each tool was selected for its specific capabilities, enabling the efficient execution of complex tasks. This chapter introduces the technologies employed in the project, their relevance, and the roles they played in achieving the desired results.

C Language, renowned for its speed and system-level programming capabilities, was used to implement numerical computations and build a main menu interface for seamless user interaction.

Python, a versatile programming language with a rich ecosystem of libraries, facilitated data preprocessing, analysis, modeling, and visualization.

IBM SPSS, a powerful statistical software, was employed for descriptive statistical analysis to extract meaningful insights from the dataset before model implementation.

Microsoft Excel, a universally accepted spreadsheet application, acted as the default viewer for CSV files generated during the project**.**

3.2 C Language

C Language played a vital role in implementing computational algorithms and managing the project's user interface. Its speed and precision made it suitable for handling numerical computations, such as calculating forward, backward, and central derivatives from the temperature dataset.

The derivative calculations were implemented in a program that efficiently processed large CSV files. These calculations provided insights into the rate of temperature change, contributing to the understanding of the forecast’s behavior.

In addition to its computational capabilities, C was used to design a Main Menu Interface. This interface provided a user-friendly platform for accessing various functionalities of the C program. Users could select specific tasks, such as performing derivative calculations or exiting the application, using a simple and intuitive menu-driven system. The menu was implemented using C's control structures, ensuring smooth navigation and robust error handling.

C’s versatility and performance not only enabled precise numerical operations but also enhanced the overall user experience by integrating a practical interface for seamless interaction with the program.

3.3 Python

Python (3.12.5 64bit) served as the backbone of forecasting in the project. Its simplicity and extensive library support made it an invaluable tool for handling diverse tasks, from data cleaning to predictive modeling and visualization.

Several Python libraries were employed to address specific aspects of the workflow:

- **pandas**: Used for data preprocessing and manipulation. It facilitated tasks such as handling missing values, performing data interpolation, and organizing datasets into a structured format suitable for analysis.

- **numpy**: Assisted in performing numerical operations and managing arrays efficiently, which were essential for certain preprocessing tasks.

- **statsmodels**: Played a crucial role in implementing the Exponential Smoothing (ETS) model for forecasting. This library enabled the project to account for error, trend, and seasonality components, ensuring accurate temperature predictions.

- **matplotlib**: Provided robust visualization capabilities for creating line graphs and charts. These visualizations helped communicate key insights from the historical data and forecasted results.

- **tqdm**: Enhanced visual appeal and detail by showing progress of the model using progress bars.

By leveraging these libraries, Python enabled the smooth execution of complex analytical tasks while maintaining clarity and readability in the code. The modularity of Python libraries ensured that each task was handled using specialized tools, maximizing efficiency.

3.4 IBM SPSS

IBM SPSS 26 was used to conduct descriptive statistical analysis on the dataset, offering valuable insights into its underlying patterns. While SPSS supports a wide range of advanced statistical operations, its role in this project was focused on summarizing and understanding the data through key descriptive statistics.

The software calculated metrics such as the mean, standard deviation, and data distribution, providing a clear picture of the dataset's characteristics. These results were essential for verifying data integrity and ensuring the dataset was suitable for predictive modeling.

3.5 Microsoft Excel

Microsoft Excel (Office Home & Student 2021) was employed primarily as a default viewer for the CSV files generated during the project. Given its widespread use and compatibility with the CSV format, Excel provided a convenient means of inspecting and reviewing datasets.

The application allowed for quick verification of the outputs generated by the C and Python programs. For example, CSV files containing derivative calculations from the C program or forecasted temperature data from the Python model could be easily opened and inspected in Excel.

While Excel did not play a direct role in data manipulation or analysis, its simplicity and accessibility made it an ideal tool for reviewing and validating the data. Its role highlights the importance of having practical solutions for ensuring data accuracy and completeness during the workflow.

3.6 Conclusion

By integrating these tools and technologies, the project achieved its objectives effectively and efficiently. Each tool contributed uniquely to the overall process, demonstrating the importance of selecting technologies tailored to specific tasks.

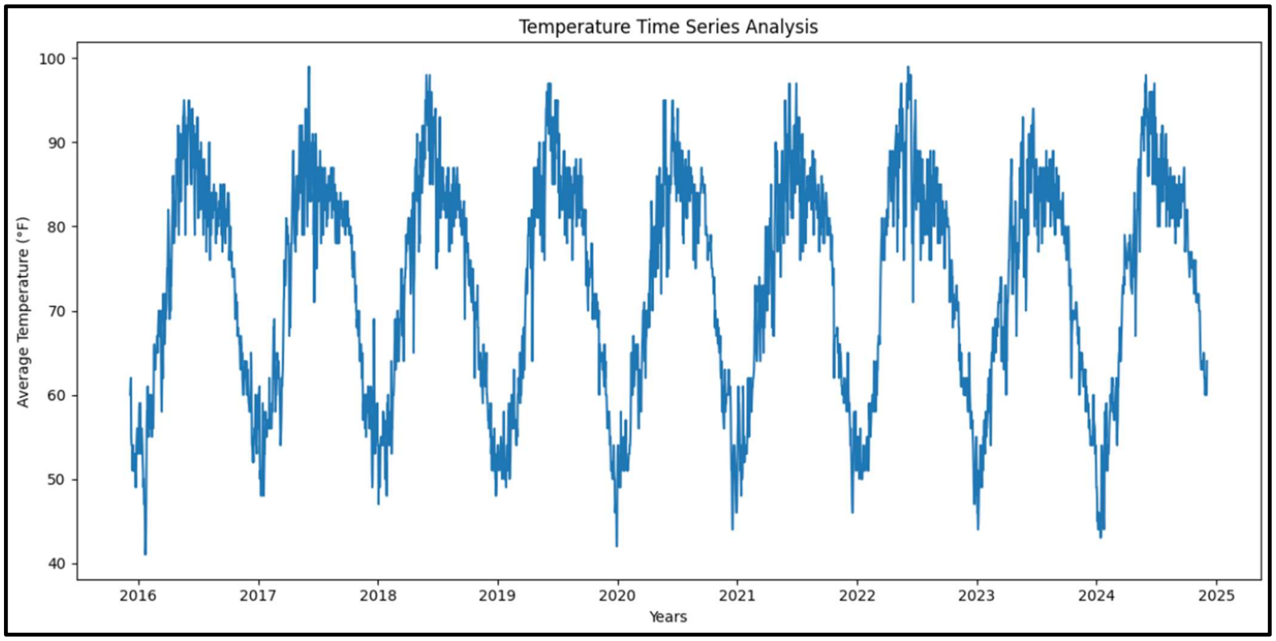
The project's success illustrates how the combination of computational efficiency (C), analytical power (Python), statistical insights (SPSS), and practical review tools (Excel) can address complex challenges. This cooperation of technologies provides a solid foundation for potential future enhancements and innovations.

Chapter 4

INTRODUCTION TO DATA

4.1 Dataset Description

The Dataset we have used for this project contains 3,255 observations of daily average temperature from December 8, 2015 to December 3, 2024. This dataset is in Time-Series format which is suitable for this research. Dataset downloaded from NOAA *“National Centers for Environmental Information”* [14]



Created using Matplotlib (Python)

4.1 (a) Key Features of the Dataset:

The dataset used in this study is suitable for time series analysis because to its systematic and complete structure. It has 3,255 entries overall, spanning almost nine years, from December 8, 2015, to December 3, 2024, and includes daily average temperature readings. Two primary variables are included in the dataset: Average Temperature (avtemp), which displays the daily average temperature in degrees Fahrenheit, and Years (date), which indicates the daily time points. Its completeness, which includes no missing value, is one of its main advantages. This makes it reliable for researching seasonal trends, temperature patterns, and any odd variations throughout time. This extensive and structured dataset offers a solid basis for detailed and insightful analysis.

4.1 (b) Dataset:

S.no	date	avtemp
1	12/08/2015	60
2	12/09/2015	61
3	12/10/2015	62
4	12/11/2015	55
5	12/12/2015	54
6	12/13/2015	54
7	12/14/2015	54
8	12/15/2015	51
9	12/16/2015	53
10	12/17/2015	54
11	12/18/2015	51
12	12/19/2015	51
13	12/20/2015	51
14	12/21/2015	53
15	12/22/2015	51
16	12/23/2015	51
17	12/24/2015	50
18	12/25/2015	49
19	12/26/2015	51
20	12/27/2015	54
21	12/28/2015	55
….	….	….
….	….	….
….	….	….
….	….	….
….	….	….
….	….	….
3235	11/08/2024	71
3236	11/09/2024	70
3237	11/10/2024	70
3238	11/11/2024	70
3239	11/12/2024	67
3240	11/13/2024	65
3241	11/14/2024	64
3242	11/15/2024	63
3243	11/16/2024	63
3244	11/17/2024	64
3245	11/22/2024	63
3246	11/23/2024	65
3247	11/24/2024	64
3248	11/25/2024	62
3249	11/26/2024	62
3250	11/27/2024	60
3251	11/28/2024	60
3252	11/29/2024	61
3253	11/30/2024	60
3254	12/02/2024	60
3255	12/03/2024	64

4.1 (b) Key Aspects of Time Series Data:

Determining the essential elements of trend, seasonality, noise, and stationarity is necessary for understanding time series data. When all of these components are present, patterns are determined, allowing for accurate forecasting.

The overall pattern of the data's evolution over time is shown by a trend. It might be going up, down, or staying the same. For example, in temperature data, a trend can show a slow rise or fall over several years. A trend that might point to long-term warming or cooling is noticeable in our dataset.

The term "seasonality" describes regular, recurrent patterns, such as those associated with the year's seasons. This may indicate that summer temperatures will be high and winter temperatures will be somewhat low. With consistent temperature variations throughout the year, the Jammu dataset clearly shows seasonal trends.

Anything in the data that fluctuates at random and doesn't appear to follow any seasonal or trend patterns is considered noise. This could be the result of sudden events, measurement errors, or a variety of other causes. In these situations, it is important to understand and minimize the impact of noise in order to take prediction properly.

When data's statistical characteristics, such its average and variation, remain consistent throughout time, it is said to be stationary. While weather data includes seasonal patterns and trends, it is typically non-stationary. We frequently need to apply techniques to stabilize non-stationary data, such as eliminating trend or seasonal effects, in order to study it.

Gaining a better understanding of trend, seasonality, noise, and stationarity improves time-series analysis. This will play an important part in providing precise models for trends and variations, especially with weather and temperature data.

4.1 (c) Exploratory Observations:

Two key findings from the study of time series temperature data involve temperature variance and extreme occurrences.

The time series shows significant daily fluctuations that must be mostly managed by annual weather patterns and seasonal cycles. Accordingly, peaks in this time series should fall during the hottest summer months, while troughs will fall during the coldest winter months. Thus, seasonal variance in temperature trends is supported by this predicted fluctuation.

A heat wave or a cold spell are examples of extreme weather events that could be linked to sudden temperature increases or drops into the abnormal zones. Because they will reveal information about unique climatic circumstances and the potential reasons or effects that may have led to them, these unusual occurrences should be further investigated.

Understanding these elements will enable us to recognize patterns and anomalies in temperature data, which is essential for accurate climate research and forecasting.

4.1 (d) Importance for Forecasting:

The dataset's time series analysis provides useful chances to understand temperature trends, predict future events, and help in decision-making. We can gain a better understanding of the underlying dynamics of variations in temperature over time by breaking down the time series into its constituent parts: trend, seasonality, and residuals. Long-term changes, recurrent seasonal patterns, and random fluctuations in the data can all be found using this analysis.

In order to generate accurate future temperature predictions, the dataset can also be used to train models such as ETS (Error, Trend, and Seasonality), which successfully capture these patterns. These kinds of models are essential for predicting and becoming ready for future developments.

4.2 Descriptive Statistics

Descriptive statistics help summarize the temperature data, providing key insights into how the values are spread and how they vary. The dataset includes 3,255 valid temperature records with no missing values, making the analysis reliable. The average temperature (mean) is 73.12°F, which gives a general idea of typical temperatures. The middle value (median) is 76.00°F, meaning half of the recorded temperatures are below this number and half are above.

The most common temperature (mode) in the dataset is 83°F. The standard deviation is 13.179, showing how much temperatures vary from the average, while the variance (173.694) further measures this spread.

The temperature range extends from 41°F to 99°F, covering a difference of 58 degrees. The skewness value of -0.304 suggests that higher temperatures occur more often than lower ones, though the difference is small.

Total Values	3255
Missing Values	0
Mean	73.12
Median	76.00
Mode	83
Standard Deviation	13.179
Variance	173.694
Skewness	-0.304

4.3 Insights Gained

4.3 (a) Central Tendency

It is observed that even though the average temperature or mean is at 73.12 degrees, mode being higher at 83 degrees shows a tendency of more presence of higher temperatures in this dataset.

4.3 (b) Spread and Shape

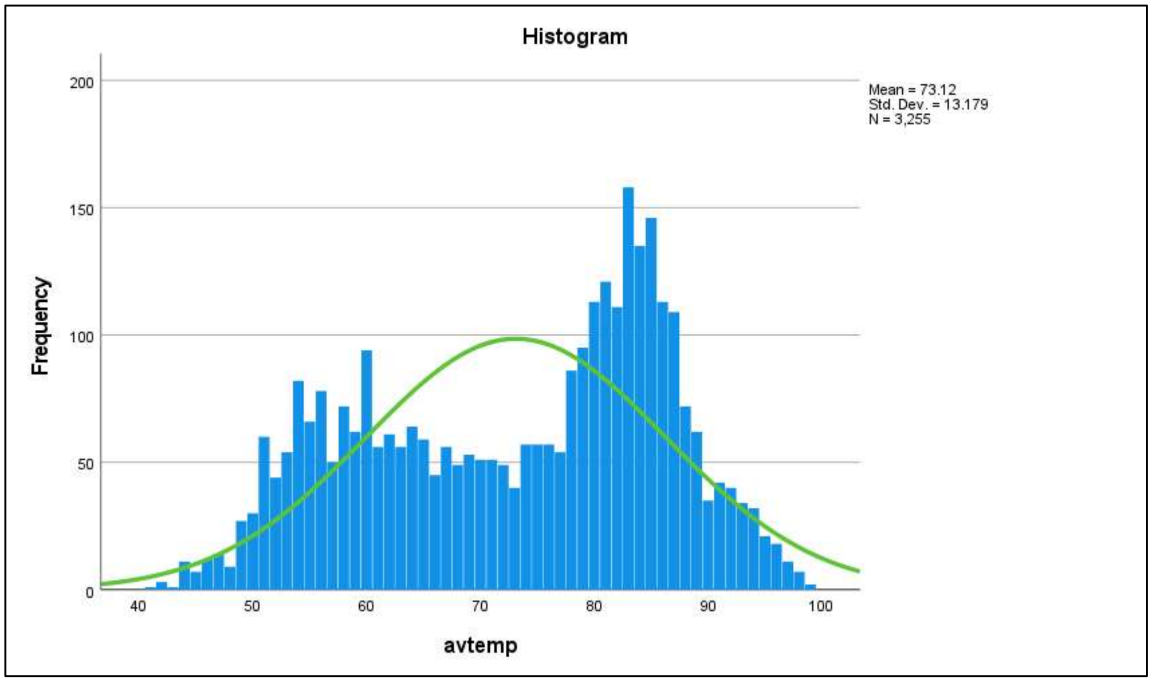
The standard deviation is moderate, which means that there is some variability in the temperatures, but most values are close to the mean. The slight negative skewness indicates that while there are some lower values, they do not significantly distort the overall distribution.

4.3 (c) Outliers

Although the range is from 41 to 99 degrees, most of the data points cluster around the mean, indicating few extreme outliers that might affect overall trends.

4.3 (d) Histogram Analysis

This can be supported by the histogram representation, which roughly shows a normal distribution with its peak around the temperatures between 80 and 85 degrees. This means that these values are indeed among the most frequently recorded. Here Frequency refers to range of average temperature.



Chapter 5

METHODOLOGY

5.1 ARIMA Model

The ARIMA model (Auto Regressive Integrated Moving Average) is a widely used statistical model, it works on the system of analyzing past data values to predict future data values. It was the very first model that we used for our study on temperature prediction. The model is popular for being used in variety of fields such as economics and finance to temperature prediction. ARIMA model always work efficiently for short-term forecasting. ARIMA model had identified many complex patterns of past observations with future needs which makes it a good technique but only for short term forecasting. From predicting the price of stocks, forecasting weather patterns to getting an idea about consumer demand, ARIMA is a great method to make accurate predictions.

5.2 ARIMA's PARAMETERS

Each component in ARIMA functions as a parameter. For ARIMA model, a standard notation would be ARIMA with p, d, and q, where integer values substitute for the parameters to indicate the type of ARIMA model used.

These parameters can be defined as:

p: it is the lag observations in the model, also known as the lag order.

d: the number of times the raw observations are differenced; also known as the degree of differencing.

q: the size of the moving average window, also known as the order of the moving average.

5.3 SARIMA Model

SARIMA which stands for Seasonal Autoregressive Integrated Moving Average, is a widely used time series forecasting model. It is an extension of ARIMA which is a non-seasonal model, it is designed to handle data which has seasonal patterns. SARIMA can work on both short-term and long-term datasets, making it a robust and good tool for forecasting. It combines all the concepts of ARIMA with seasonal components.

5.4 Conclusion

The ARIMA model is a robust and it is a widely used model for time series forecasting, it offers flexibility and reliability for data with consistent patterns and trends. However, in our study, the model's outcomes were not as we desired, it was mainly due to the data variable we have, as we only had a single variable in the dataset. The SARIMA model is extension of ARIMA, it is built to handle both trends and seasonality in time series data at once. The ability of SARIMA model to handle seasonality is an advantage but it did not perform effectively in our study. Although the model is good for many applications but its performance limited when our dataset did not align with its assumptions and also this model was computational heavy so at the end, we needed to drop this model for better working for our study and opt for ETS (Exponential Smoothing) Model.

5.5 ETS Model

The ETS (Error, Trend, Seasonality) model is a powerful and widely used time series forecasting method that decomposes a time series into its underlying components—error, trend, and seasonality. These components are combined through either an additive or multiplicative approach to better capture the dynamics of the data. The ETS model is particularly effective in cases where the time series data exhibits clear patterns of trend and seasonality, such as in weather forecasting, financial market predictions, and energy consumption. It works by smoothing the data with weighted averages, where more recent observations are given greater weight than older ones. This method is adaptive and can update its components as new data is received, making it a useful tool for both short-term and long-term forecasting.

In temperature prediction, the ETS model is valuable because it captures the underlying seasonal patterns, long-term trends, and irregular fluctuations in temperature data. By considering the past temperature data in relation to seasonal cycles and trends, the ETS model can provide accurate forecasts, accounting for both expected fluctuations and unexpected changes in temperature over time. Its flexibility and ability to incorporate different types of patterns in the data make it an ideal choice for temperature forecasting where both trend and seasonal variations are crucial.

5.5 (a) ETS Model Parameters

The ETS model consists of three main components—Error (E), Trend (T), and Seasonality (S)—each of which captures different aspects of the time series data. These components are combined using smoothing techniques, which can be either additive or multiplicative depending on the characteristics of the data.

- **Error (E)**: This represents the unpredictable or random fluctuations in the data. It accounts for any noise or irregularities that do not follow any systematic pattern. The error can be modeled in two ways:
    - **Additive (A)**: The error is constant and added to the other components regardless of the level of the data.
    - **Multiplicative (M)**: The error is proportional to the level of the data, meaning it changes based on the magnitude of the series.

- **Trend (T)**: The trend component captures the long-term movement in the data, such as an upward or downward direction over time. It can be modeled in several ways:
    - **No trend (N)**: There is no long-term directional movement in the data.
    - **Linear (A)**: The data exhibits a constant rate of change over time, either increasing or decreasing.
    - **Exponential (M)**: The rate of change itself accelerates or decelerates over time, meaning the trend becomes steeper or flatter.

- **Seasonality (S)**: This component accounts for recurring patterns in the data that repeat over a fixed period, such as daily, weekly, or yearly cycles. Seasonality can be modeled as:
    - **Additive (A)**: The seasonal effect is constant over time, meaning the seasonal variation remains the same regardless of the level of the data.
    - **Multiplicative (M)**: The seasonal effect is proportional to the level of the data, meaning the seasonal variation changes as the data increases or decreases.

5.5 (b) Decomposition Process

The ETS model breaks down the time series data into these components (error, trend, and seasonality), and assumes that the observed data is a combination of them. The combination can either be additive or multiplicative:

- **Additive**: The components are summed together: Data = Trend + Seasonality + Error
- **Multiplicative**: The components are multiplied together: Data = Trend * Seasonality * Error

5.5 (c) Smoothing Process

The core of the ETS model lies in exponential smoothing, where weighted averages of past observations are used to estimate the current level, trend, and seasonal components. More recent data points are given more weight, and this smoothing process updates the components over time. The parameters that control this smoothing process include:

- **Alpha (α)**: The smoothing parameter for the level, controlling how much weight is given to the most recent observation.
- **Beta (β)**: The smoothing parameter for the trend, controlling the influence of the recent trend on the model.
- **Gamma (γ)**: The smoothing parameter for the seasonal component, controlling the impact of recent seasonal variations.

5.5 (d) Forecasting

Once the model estimates the current level, trend, and seasonal components, it can generate forecasts by extrapolating these components into the future. This enables the model to predict future values based on the assumption that the current patterns will continue.

5.5 (e) Types of ETS Models

There are different variations of the ETS model, depending on how the error, trend, and seasonality are combined and whether the trend is damped (i.e., the impact of the trend diminishes over time). Some common types of ETS models include:

- **ETS(A, N, N)**: Additive error, no trend, no seasonality (Simple Exponential Smoothing).
- **ETS(A, A, N)**: Additive error, additive trend, no seasonality (Holt's Linear Trend).

- **ETS(A, A, A)**: Additive error, additive trend, additive seasonality.
- **ETS(M, M, M)**: Multiplicative error, multiplicative trend, multiplicative seasonality.

5.5 (f) Key Advantages of ETS

- Effectively handles trend and seasonality, making it ideal for forecasting data with clear patterns.
- Simple to understand and implement, which makes it a popular choice for time series forecasting.
- Provides reliable short-term forecasts by continuously updating its components based on recent data.

In summary, the ETS model is a robust and flexible method for time series forecasting, decomposing data into its core components (error, trend, and seasonality) and using exponential smoothing to generate accurate forecasts. Its adaptability to different data behaviors makes it a valuable tool for temperature prediction and other forecasting tasks.

5.6 Discrete Derivative

In this study, Discrete Derivative is used in order to find the rate of change of the average temperature with respect to time, the variable used in the study is daily average temperature for the next 45 days which is generated using our program. Discrete derivative in mathematical terms known as the difference between the values of a function at two different points, divided by the difference between those points. It is a way of approximating discrete data such as daily average temperature in our study which have derivative of the functions in discrete terms.

In this study of predicting temperature using ETS model, time series data consists of parameters like average temperature as in our study which makes it important for us to use discrete derivative for finding the rate of change at which temperature changes with respect to time.

Discrete derivative is widely used for measuring these rates of change, it is also used by meteorologists to detect extreme weather conditions also helps them to analyze ongoing weather systems, and improve existing weather forecasting models. Since weather data is mostly available in intervals (e.g., hourly or daily), the discrete derivative act as efficient tool to analyze how these variables change over time.

5.7 Mathematical Formulation

5.7(a) Forward Difference:

The forward difference method is defined as the difference between the function's value at a point and the same function's value at the next point.
It can be expressed as:

$$\frac{dy}{dx} \approx \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

Explanation of Terms:

- **y_i:** The value of dependent variable at the current point.
- **y_{i+1}:** Value of y at the next point.
- **x_i:** Value of the independent variable at the current point.
- **x_{i+1}:** Value of x at the next point.

WORKING: The forward difference method can estimate the rate change using the difference between the current point(i) and next point (i+1). It is called forward because it looks ahead to the next point.

APPLICABILITY: this method is really useful for the starting of the dataset where there is no previous point (i-1) to complete backward or central difference.

ACCURACY: Less accurate compared to the central difference method, as it only considers the change in one direction (forward).

5.7 (b) Backward Difference:
The backward difference looks at the difference between the function's value at a point and the function's value at the previous point.

The formula is:

$$\frac{dy}{dx} \approx \frac{y_i - y_{i-1}}{x_i - x_{i-1}}$$

This is also an approximation of the derivative but with a different approach to sampling the data points.

Explanation of Terms:

- **y_i :** Value of the dependent variable at the current point.
- **y_{i-1}:** Value of the dependent variable at the previous point.
- **x_i:** Value of the independent variable at the current point.
- **x_{i-1}:** Value of the independent variable at the previous point.

WORKING: This method calculates the rate of change (approximation of the derivative)between the current point and the previous point.

APPLICABILITY: It is applicable at the end of a dataset where forward or central difference methods are not workable.

ACCURACY: This method is less accurate as compared to the central difference method, as it only considers the change in one direction (backward).

5.7 (c) Central Difference:

The central difference is a more correct method because it takes both the forward and backward differences into account and then calculates the rate of change.

The formula is:

$$\frac{dy}{dx} \approx \frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}}$$

This method provides a more balanced approximation of the derivative and is often preferred in numerical methos for its higher accuracy compared to the forward and backward differences.

Explanation of Terms:

1. **yi+1:** The value of the dependent variable y at the next point.
2. **yi−1:** The value of y at the previous point.
3. **xi+1:** The value of the independent variable x at the next point.
4. **xi−1:** The value of x at the previous point.

WORKING: The central method uses both forward and backward differences to calculate derivative making it more accurate than other methods.

APPLICABILITY: It works well for internal point where x_{i+1} and x_{i-1} are available. It cannot be applied for the first and last points in the data set.

ACCURACY: It is more accurate than forward and backward alone, as its average changes on both sides.

5.8 Applications of Discrete Derivatives in Weather Analysis:

Discrete derivatives play a crucial role in analyzing weather data. It helps us understand how weather conditions like temperature, pressure, wind speed, and precipitation change over time. By calculating the rate of change, discrete derivatives give insights into weather patterns and helps to improve the weather forecasts, making it easier to predict weather events**.**

- **Measuring Rates of Change:** Through discrete derivative we analyzed sudden shifts in temperature by calculating how quickly weather variables change.

- **Trend Detection:** Use of discrete derivative helped in identifying long-term weather trends, such as climate change, and predicting future weather patterns.

- **Anomaly Detection:** Spotting abrupt changes, such as rapid temperature drops, to identify extreme weather events early.

- **Improved Forecasting:** Enhancing weather predictions by tracking the rate of change in weather variables.

5.9 Conclusion

The discrete derivative provided us a way to analyze the rate of change of average temperature at discrete points of next 45 days. It is significantly valuable in our study. By using methods like forward, backward, and central differences, discrete derivatives not only offered the study flexibility but also adaptability. Although there may be trade-offs between accuracy and processing efficiency, the accuracy of these approximations is primarily determined by the data's characteristics and the step size selection. Discrete derivative is an effective and efficient mathematical tool for our study.

5.10 Experimental Setup

The experimental setup for the temperature prediction project integrates software tools, datasets, and computational processes to ensure accurate and efficient outcomes. This section describes the architecture of the system and the sequential workflow.

5.10 (a) System Architecture

The project employs data preprocessing and statistical modeling techniques to develop a robust temperature prediction model using the ETS (Error, Trend, Seasonal) method. The architecture is structured into three primary layers.

The Input Layer involves historical temperature data in CSV format as the raw dataset, with Microsoft Excel serving as the default tool for preliminary dataset inspection.

The Processing Layer handles data preprocessing and statistical modeling. Libraries like Pandas and NumPy in Python are utilized to treat missing values and normalize the dataset. The ETS model, implemented using the Statsmodels library, is applied for analyzing and predicting temperature trends based on error, trend, and seasonal components.

The Output Layer visualizes prediction using Python libraries like Matplotlib and displaying output CSV files using Microsoft Excel.

5.10 (b) Workflow

The workflow for this project follows a structured methodology to ensure the reliable generation of predictions. First, historical temperature data is collected and stored in CSV format. The data is then preprocessed to address missing values and ensure consistency. Once cleaned, descriptive statistics are applied to identify key trends in the data.
The ETS model is employed to decompose the dataset into its error, trend, and seasonal components, which are used to generate forecasts. The results, including predictions, are visualized to provide insights into the model's accuracy and the observed trends. Additionally, a proposed future step includes integrating hardware for real-time data collection and seamless interaction with the software system

Chapter 6

RESULTS & DISCUSSION

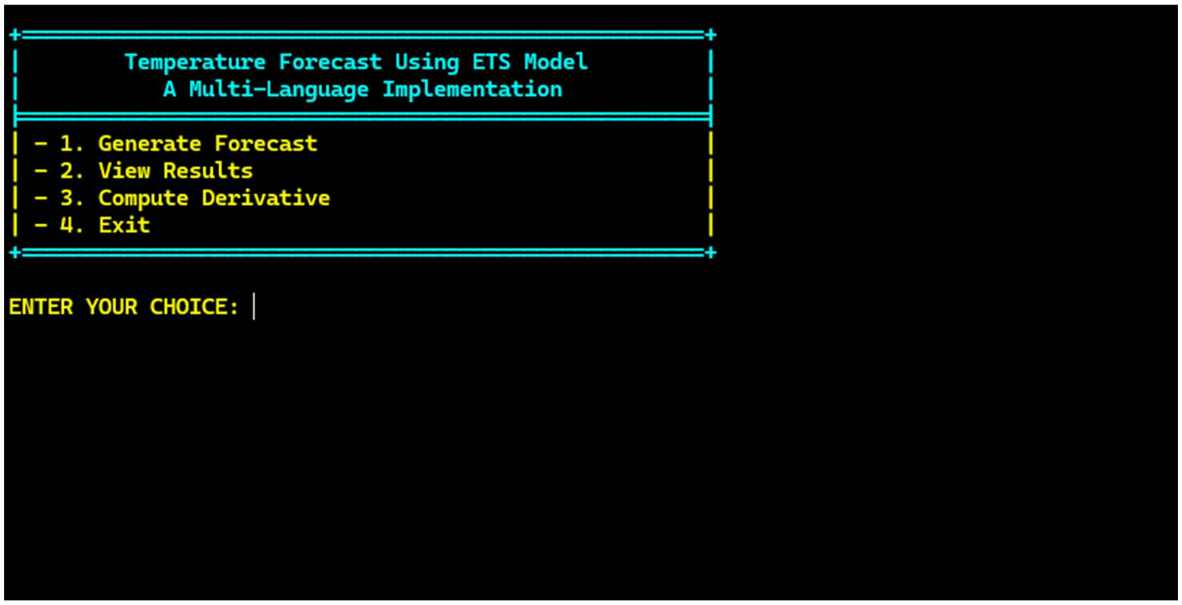
6.1 Introduction

The developed system's execution time and performance were tracked across various environments and hardware combinations. The entire procedure, which includes data preparation, ETS model training, prediction generation, and derivative computation, takes about **25 to 30 minutes** on cloud platforms such as **Google Collab**.

We used our systems equipped with **Intel Core Ultra 5 125H, Core i5 13450HX**, and **Core i7 13650HX** Processors and **16 GB of DDR5 RAM** were tested locally to ensure the system operated as intended. Even for demanding workloads or systems with moderate power, it was observed that the model achieved a similar execution time despite the variations in configurations. This suggests that the implementation would be feasible and would guarantee timely results for forecasting and analysis**.**

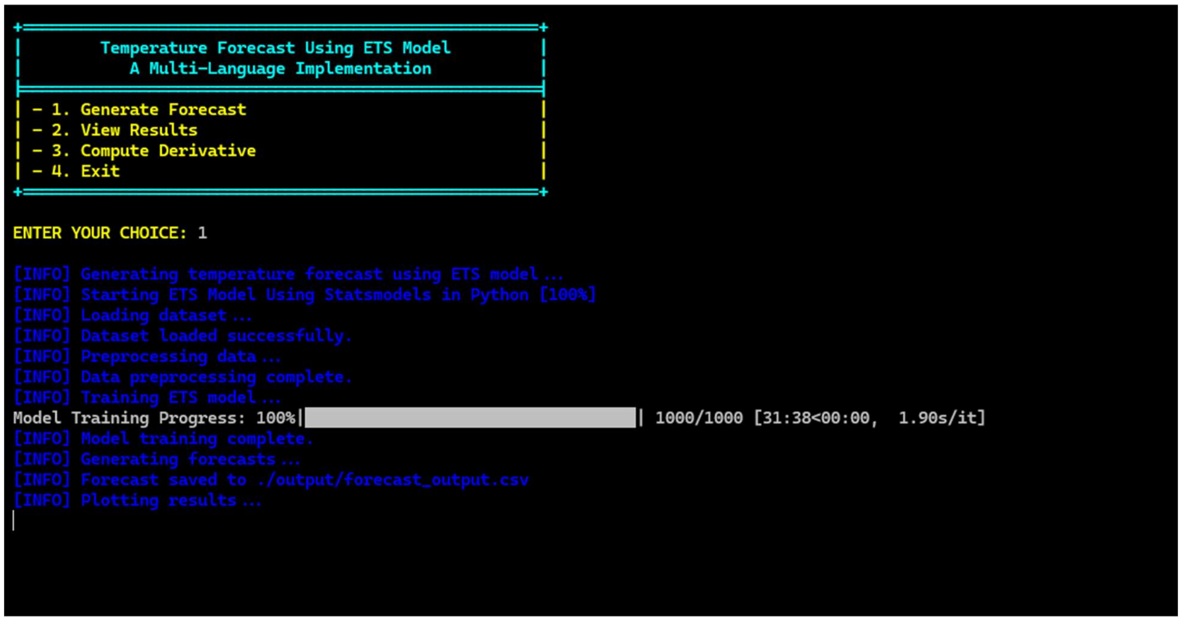
6.2 The Interface and Outputs

1. Interface developed Using C Language



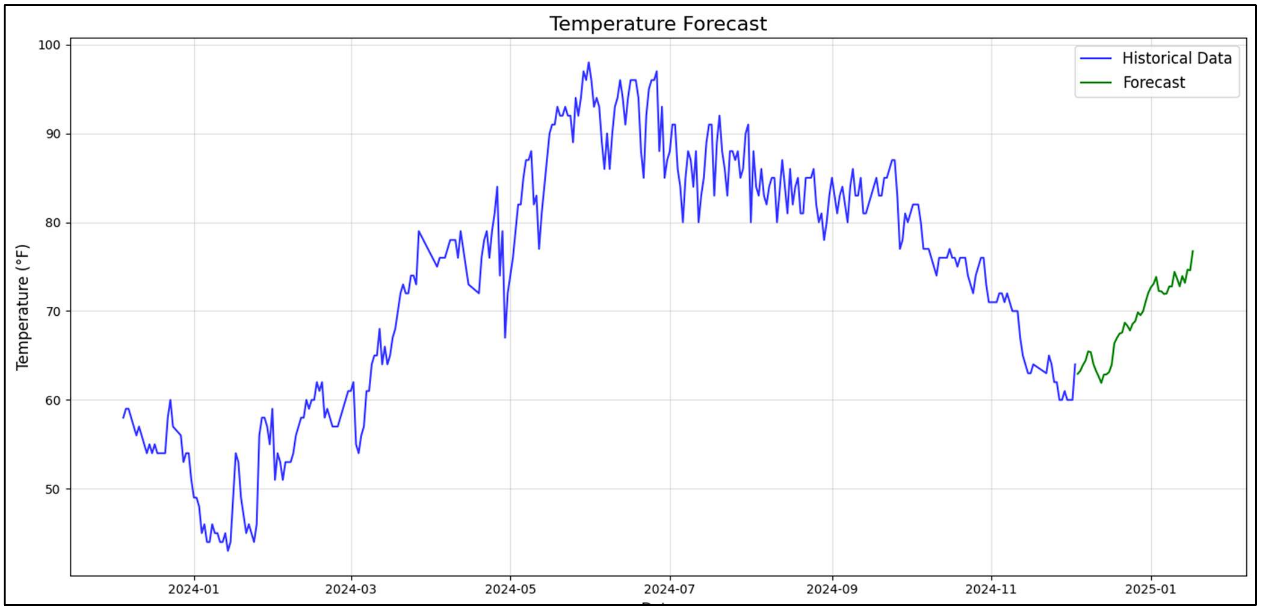
```
+==================================================+
|        Temperature Forecast Using ETS Model       |
|           A Multi-Language Implementation         |
|==================================================|
| - 1. Generate Forecast                            |
| - 2. View Results                                 |
| - 3. Compute Derivative                           |
| - 4. Exit                                         |
+==================================================+

ENTER YOUR CHOICE: |
```

2. Running ETS Model in Python within the C Interface

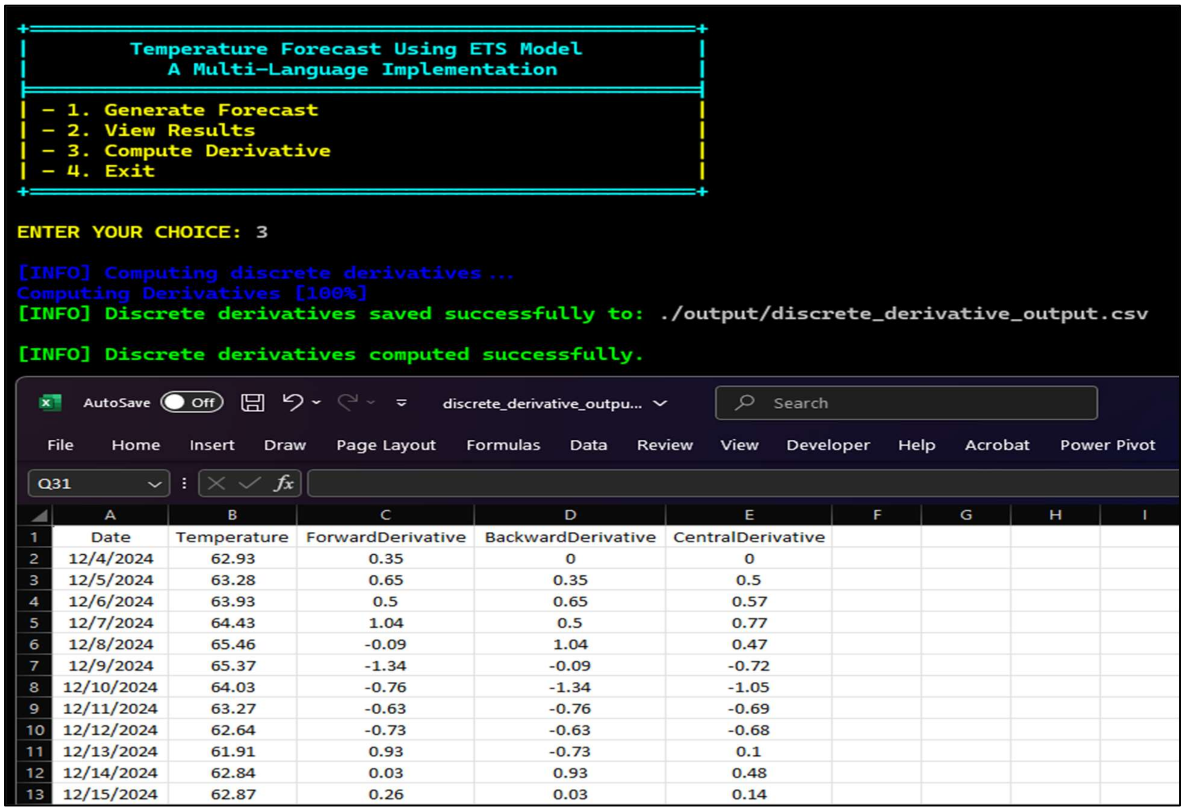


```
+==================================================+
|        Temperature Forecast Using ETS Model        |
|           A Multi-Language Implementation          |
|==================================================|
| - 1. Generate Forecast                             |
| - 2. View Results                                  |
| - 3. Compute Derivative                            |
| - 4. Exit                                          |
+==================================================+

ENTER YOUR CHOICE: 1

[INFO] Generating temperature forecast using ETS model ...
[INFO] Starting ETS Model Using Statsmodels in Python [100%]
[INFO] Loading dataset ...
[INFO] Dataset loaded successfully.
[INFO] Preprocessing data ...
[INFO] Data preprocessing complete.
[INFO] Training ETS model ...
Model Training Progress: 100%|██████████████████████████████████| 1000/1000 [31:38<00:00,  1.90s/it]
[INFO] Model training complete.
[INFO] Generating forecasts ...
[INFO] Forecast saved to ./output/forecast_output.csv
[INFO] Plotting results ...
|
```

3. Graph Generated by our Program



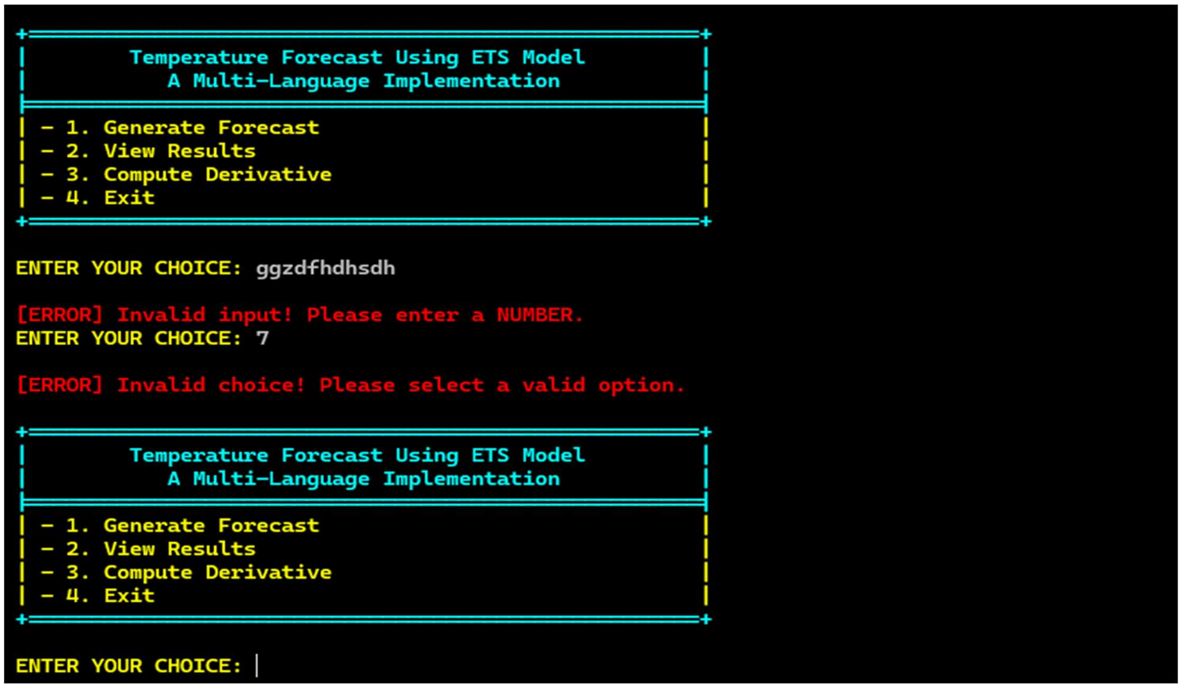
4. Discrete Derivative Calculated using C Language



```
+==========================================================+
|          Temperature Forecast Using ETS Model            |
|             A Multi-Language Implementation              |
|==========================================================|
| - 1. Generate Forecast                                   |
| - 2. View Results                                        |
| - 3. Compute Derivative                                  |
| - 4. Exit                                                |
+==========================================================+

ENTER YOUR CHOICE: 3

[INFO] Computing discrete derivatives ...
Computing Derivatives [100%]
[INFO] Discrete derivatives saved successfully to: ./output/discrete_derivative_output.csv

[INFO] Discrete derivatives computed successfully.
```

	A	B	C	D	E
1	Date	Temperature	ForwardDerivative	BackwardDerivative	CentralDerivative
2	12/4/2024	62.93	0.35	0	0
3	12/5/2024	63.28	0.65	0.35	0.5
4	12/6/2024	63.93	0.5	0.65	0.57
5	12/7/2024	64.43	1.04	0.5	0.77
6	12/8/2024	65.46	-0.09	1.04	0.47
7	12/9/2024	65.37	-1.34	-0.09	-0.72
8	12/10/2024	64.03	-0.76	-1.34	-1.05
9	12/11/2024	63.27	-0.63	-0.76	-0.69
10	12/12/2024	62.64	-0.73	-0.63	-0.68
11	12/13/2024	61.91	0.93	-0.73	0.1
12	12/14/2024	62.84	0.03	0.93	0.48
13	12/15/2024	62.87	0.26	0.03	0.14

5. Proper Error Handling



```
+==========================================================+
|          Temperature Forecast Using ETS Model            |
|             A Multi-Language Implementation              |
|==========================================================|
| - 1. Generate Forecast                                   |
| - 2. View Results                                        |
| - 3. Compute Derivative                                  |
| - 4. Exit                                                |
+==========================================================+

ENTER YOUR CHOICE: ggzdfhdhsdh

[ERROR] Invalid input! Please enter a NUMBER.
ENTER YOUR CHOICE: 7

[ERROR] Invalid choice! Please select a valid option.

+==========================================================+
|          Temperature Forecast Using ETS Model            |
|             A Multi-Language Implementation              |
|==========================================================|
| - 1. Generate Forecast                                   |
| - 2. View Results                                        |
| - 3. Compute Derivative                                  |
| - 4. Exit                                                |
+==========================================================+

ENTER YOUR CHOICE: |
```

6. Forecast Output

date	avtemp
12/4/2024	62.93467392
12/5/2024	63.28119474
12/6/2024	63.9279314
12/7/2024	64.42753213
12/8/2024	65.46405529
12/9/2024	65.36957089
12/10/2024	64.03239006
12/11/2024	63.2738247
12/12/2024	62.644751
12/13/2024	61.91357602
12/14/2024	62.84499541
12/15/2024	62.87205725
12/16/2024	63.13333786
12/17/2024	63.95974737
12/18/2024	66.37052089
12/19/2024	66.97952878
12/20/2024	67.46286759
12/21/2024	67.58271424
12/22/2024	68.69516348
12/23/2024	68.34541669
12/24/2024	67.81489367
12/25/2024	68.5698773
12/26/2024	68.85966341
12/27/2024	69.8571294
12/28/2024	69.53721209
12/29/2024	69.985998
12/30/2024	71.0873552
12/31/2024	72.06217206
1/1/2025	72.68148179
1/2/2025	73.05718843
1/3/2025	73.85280149
1/4/2025	72.26255342
1/5/2025	72.21814838
1/6/2025	71.92733001
1/7/2025	71.97986276
1/8/2025	72.77286118
1/9/2025	72.76967282
1/10/2025	74.4143711
1/11/2025	73.6530036
1/12/2025	72.78723078
1/13/2025	73.94672095
1/14/2025	73.18487855
1/15/2025	74.66543344
1/16/2025	74.58650387
1/17/2025	76.74574884

7. Derivative Output

Date	Temperature	Forward Derivative	Backward Derivative	Central Derivative
12/4/2024	62.93	0.35	0	0
12/5/2024	63.28	0.65	0.35	0.5
12/6/2024	63.93	0.5	0.65	0.57
12/7/2024	64.43	1.04	0.5	0.77
12/8/2024	65.46	-0.09	1.04	0.47
12/9/2024	65.37	-1.34	-0.09	-0.72
12/10/2024	64.03	-0.76	-1.34	-1.05
12/11/2024	63.27	-0.63	-0.76	-0.69
12/12/2024	62.64	-0.73	-0.63	-0.68
12/13/2024	61.91	0.93	-0.73	0.1
12/14/2024	62.84	0.03	0.93	0.48
12/15/2024	62.87	0.26	0.03	0.14
12/16/2024	63.13	0.83	0.26	0.54
12/17/2024	63.96	2.41	0.83	1.62
12/18/2024	66.37	0.61	2.41	1.51
12/19/2024	66.98	0.48	0.61	0.55
12/20/2024	67.46	0.12	0.48	0.3
12/21/2024	67.58	1.11	0.12	0.62
12/22/2024	68.7	-0.35	1.11	0.38
12/23/2024	68.35	-0.53	-0.35	-0.44
12/24/2024	67.81	0.75	-0.53	0.11
12/25/2024	68.57	0.29	0.75	0.52
12/26/2024	68.86	1	0.29	0.64
12/27/2024	69.86	-0.32	1	0.34
12/28/2024	69.54	0.45	-0.32	0.06
12/29/2024	69.99	1.1	0.45	0.78
12/30/2024	71.09	0.97	1.1	1.04
12/31/2024	72.06	0.62	0.97	0.8
1/1/2025	72.68	0.38	0.62	0.5
1/2/2025	73.06	0.8	0.38	0.59
1/3/2025	73.85	-1.59	0.8	-0.4
1/4/2025	72.26	-0.04	-1.59	-0.82
1/5/2025	72.22	-0.29	-0.04	-0.17
1/6/2025	71.93	0.05	-0.29	-0.12
1/7/2025	71.98	0.79	0.05	0.42
1/8/2025	72.77	0	0.79	0.39
1/9/2025	72.77	1.64	0	0.82
1/10/2025	74.41	-0.76	1.64	0.44
1/11/2025	73.65	-0.87	-0.76	-0.81
1/12/2025	72.79	1.16	-0.87	0.15
1/13/2025	73.95	-0.76	1.16	0.2
1/14/2025	73.18	1.48	-0.76	0.36
1/15/2025	74.67	-0.08	1.48	0.7
1/16/2025	74.59	2.16	-0.08	1.04
1/17/2025	76.75	0	2.16	0

6.3 Code Implementation

Our study includes three main code implementations first Main Menu code written in C Language, Secondly ETS Model in Python and Discrete Derivative in C language

The implemented system, which includes a C-based program for computing discrete derivatives of the predicted temperature data and an Exponential Smoothing (ETS) model for temperature forecasting based on Python, is described in this part.

In order to fit an ETS model to historical temperature data, estimate future temperatures, and produce predictions, the Python code makes use of the statsmodels library. After reading the predicted data, the C-based program computes the forward, backward, and central discrete derivatives and outputs the findings for additional examination.

6.3 (a) Data Preprocessing and Model Training

After completing the required preprocessing, the Python code loads a temperature dataset from a CSV file. The average temperature (avtemp) is scaled to a standardized form, and missing values are filled in by interpolating the data. The scaled temperature data is then used to initialize and train the ETS model, taking trend and seasonal components into account.

```
# Data preprocessing: interpolate missing values and scale temperature
data
df['avtemp'] = df['avtemp'].interpolate(method='linear')
mean_temp, std_temp = df['avtemp'].mean(), df['avtemp'].std()
df['TAVG_scaled'] = (df['avtemp'] - mean_temp) / std_temp
```

A 365-day seasonal period was used in the model's configuration, which indicated additive error, trend, and seasonality components using the error='add', trend='add', and seasonal='add' choices. Daily temperature data that shows annual seasonal patterns can benefit from this.

6.3 (b) ETS Model Training

The ETSModel.fit() function is used to train the model on the preprocessed data, and forecasts for the upcoming 45 days are produced. The mean and standard deviation from the training data is then used to unscaled the predicted values to their initial temperature units.

```
forecast_scaled = fitted_model.forecast(forecast_days)
forecast = (forecast_scaled * std_temp) + mean_temp
```

The forecast is saved to a CSV file, and a plot is generated to visualize the forecasted temperature alongside historical data.

6.3 (c) Model Evaluation

The model performed its best of capturing the temperature data's seasonal trend. The forecast figure, which is included in the report, shows how the expected temperature matches past trends, proving that the model was successful in capturing the underlying patterns. However, the quality of the input data determines how accurate the forecast is, and more advanced models using Machine Learning can improve the forecast's accuracy further.

6.3 (d) C Code: Discrete Derivative Calculation

The forecast data produced by the Python model is processed by the C-based program. It extracts the relevant data into a Forecast Data structure after reading the CSV file with the temperature forecast. The program reads the input file line by line and stores the temperature data using simple file handling techniques.

```
// Read forecast data from the CSV file
int count = read_forecast_data(input_filename, &data);
```

6.3 (e) Derivative Computation

The program calculates the forward, backward, and central derivatives for every data point after the data has been loaded. These derivatives provide information about how quickly the temperature is rising or falling by showing the rate of change between consecutive time points.

```
// Compute discrete derivatives
if (i < count - 1) { // Forward difference
    forward_derivative = (data[i + 1].temperature - data[i].temperature);
}
if (i > 0) { // Backward difference
    backward_derivative = (data[i].temperature - data[i - 1].temperature);
}
if (i > 0 && i < count - 1) { // Central difference
    central_derivative = (data[i + 1].temperature - data[i -
1].temperature)/ 2.0;
}
```

An output CSV file contains the derivatives' results as well as the associated temperature readings. The program offers a user-friendly experience by displaying the state of the derivative computing process using a straightforward progress bar.

```
// Save results to output file
fprintf(output_file, "%s,%.2f,%.2f,%.2f,%.2f\n", data[i].date,
data[i].temperature,
        forward_derivative, backward_derivative, central_derivative);
```

6.3 (f) Integration of Python and C Code

Through the use of system calls in the main menu program, the Python and C code were integrated. The following function in the C-based program allows the user to calculate the derivatives and then generate the forecast utilizing a Python script:

```
// Call Python script to generate forecast
int result = system("python ./scripts/forecast_ets.py ./data/dataset.csv
./output/forecast_output.csv");
```

After generating the forecast, the program proceeds to compute the discrete derivatives by calling the C-based executable:

```
// Call C program to compute derivatives
snprintf(command, sizeof(command), ".\\scripts\\discrete_derivative.exe %s
%s", input_file, output_file);
```

The system's overall usability and functionality are improved by this smooth integration between Python and C, which enables the user to carry out forecasting and derivative analysis in a single, unified process.

6.3 (g) Conclusion

The ETS model is used by the implemented system to reliably predict temperature trends, and discrete derivatives are computed to examine the rate of temperature change. A comprehensive solution for both analysis and prediction is offered by the well-integrated Python and C code. Exploring advanced models for forecasting and refining the derivative computations to better manage extreme circumstances are potential future advancements.

Chapter 7

FUTURE SCOPE & RECOMMENDATIONS

7.1 Future Scope

The system in use shows that it is feasible to predict temperature using Exponential Smoothing (ETS) and analyze rate-of-change behavior using discrete derivatives. To improve its utility, scalability, and accuracy, this system can be improved in a variety of areas.

7.1 (a) Integration of Advanced Models

Even while the ETS model works well for many time-series problems, it could miss complex patterns in temperature data, particularly when there are large non-linear trends, unexpected shifts, or different datasets. More powerful time-series forecasting models could be explored in future research, such as**:**

- **Machine Learning Techniques**: Neural networks, like Transformer models or LSTM (Long Short-Term Memory) networks, can be used to simulate extensive non-linear relationships in temperature measurements.
- **XGBoost/LightGBM:** These gradient boosting algorithms could also be explored for temperature prediction by treating it as a regression problem.

7.1 (b) Integration with Real-Time Data

The present system forecasts based on previous temperature readings. It would be useful to continuously update projected temperatures in real-world applications when new temperature data becomes available. The system could be integrated with real-time weather data APIs (like AccuWeather, OpenWeatherMap, or local meteorological data sources) to do this. Through this integration, the system would be able to adjust to shifting circumstances and produce predictions that are more accurate.

7.1 (c) Integration of External Factors

Location, altitude, and air pressure are only a few of the external variables that frequently affect temperature trends. Further developments of the system could implement the following extra features in the forecasting model:

- **Environmental factors:** Air pressure, humidity, and wind speed.

- **Geospatial data:** Latitude, longitude, and altitude

By adding these more variables, the model may be more capable to manage complex weather patterns and regional temperature changes.

7.1 (d) Visualization Improvements

Basic line plots are produced by the present system to show the past and predicted temperatures. Using interactive plots using libraries like **Plotly** or **Bokeh**, further improvements could improve the visualization. These libraries facilitate interactive features that facilitate data analysis at different levels of detail, such as zooming, panning, and tooltips.

7.2 Recommendations

7.2 (a) Model Tuning and Optimization

The ETS model can be further improved by fine tuning its parameters. Following strategies are recommended:

- **Grid Search:** Implement grid search to check different combinations of seasonal periods, trend models to find the optimal set of parameters

- **Ensemble Methods:** Combining multiple models (e.g., ETS, ARIMA, and Machine Learning Models) to improve accuracy and performance

7.2 (b) Real-Time Integration and Automation

The ability of the system to automatically retrieve and interpret new data as it becomes available is essential for real-world applications, such as those in energy management or agriculture. This can be accomplished by combining automated forecast creation with scheduled data retrieval.

Scheduled Data Fetching: The system can periodically retrieve real-time weather data by putting in place an automated pipeline, such as cron jobs or other scheduling tools. This guarantees that the most up-to-date data is constantly accessible for analysis

Forecast Generation Automation: Every step of the process, including retrieving data, creating forecasts, and calculating derivatives, can be automated.

This would eliminate the need for manual intervention and enable users to receive regular updates. By ensuring that the system can produce precise and timely predictions, automated forecasting assists industries in making well-informed decisions based on up-to-date information.

Industries in areas like agricultural and energy management may benefit from more effective, responsive, and data-driven decision-making by incorporating real-time data collecting and automating the forecasting process.

7.2 (c) User Interface Improvements

Although the command line is the primary means of interaction in the current system, a graphical user interface (GUI) could improve usability in future versions. This would improve the system's accessibility and usability, particularly for non-technical users who might find command-line interfaces difficult to use.

Graphical User Interface (GUI): Without having to use the terminal, a GUI would make it simple for users to upload datasets, view forecasts, and modify model settings. This interface might be developed with tools like **Tkinter** for Python or **Qt** and **GTK** for C, giving users an easy-to-use method of interacting with the system.

Web-Based Interface: By providing users with access to forecasting and analytical tools from anyplace with an internet connection, a web-based front-end could further increase accessibility. This could allow for integration with cloud-based systems for real-time data processing, storing, and retrieval, increasing the system's adaptability.

The system might provide a more practical, adaptable, and scalable solution for users in a variety of industries by including a graphical user interface (GUI) or a web-based interface, improving accessibility and usability.

7.3 Proposed Hardware Design

While this project primarily focused on software-driven solutions for temperature prediction, a conceptual hardware design was also considered to complement the system. The inclusion of hardware components aims to create a more integrated and practical implementation of the predictive model in real-world scenarios.

7.3 (a)Overview of the Hardware System

The proposed hardware design involves a combination of sensors, a processing unit, and a communication module to facilitate the collection, processing, and visualization of temperature data. This system can enhance the project's applicability in various domains, such as weather monitoring, climate studies, and industrial applications.

7.3 (b) Key Components

1. **Temperature Sensors**
   Sensors such as the **DHT22** or **DS18B20** are proposed for real-time temperature data acquisition. These sensors are highly accurate and can measure a wide range of temperatures, making them suitable for diverse environmental conditions.

2. **Microcontroller Unit**
   A microcontroller such as the **Arduino Uno** or a single board computer like **Raspberry Pi 5** is recommended for processing the data collected by the sensors. The unit will handle tasks such as sensor integration, data preprocessing, and transmission to the software system for further analysis.

3. **Communication Module**
   To enable real-time data transfer, a Wi-Fi or Bluetooth module (e.g., ESP8266 or HC-05) is integrated into the design. This allows seamless communication between the hardware and the software system running the prediction model.

4. **Power Supply**
   The system requires a stable power source, which can be achieved using a rechargeable battery or direct power connection. This ensures uninterrupted operation, even in remote locations.

5. **Data Storage and Logging**
   An SD card module is proposed for temporary local storage of collected data. This ensures data redundancy and provides a backup in case of communication failures.

6. **Output Display (Optional)**
   An optional addition to the hardware system is an LCD or OLED display for real-time temperature readings. This feature can offer immediate feedback to users in on-site applications.

7.3 (c) Integration with Software System

The hardware design complements the predictive model developed in the software system by providing real-time input data. The collected data can be transmitted to the software via a network or physical connection, where it undergoes preprocessing and analysis. The integration ensures a continuous flow of data, enhancing the accuracy and relevance of the predictions.

7.3 (d) Applications of the Proposed Design

The proposed hardware design broadens the scope of the project, enabling its application in the following areas:

- **Smart Agriculture**: Monitoring soil and ambient temperatures for optimizing crop growth conditions.

- **Climate Monitoring**: Establishing low-cost weather stations for localized climate data collection.

- **Industrial Applications**: Ensuring temperature regulation in manufacturing processes.

7.3 (e) Future Considerations

To further refine the hardware design, the following considerations can be explored:

- Implementing energy-efficient components for extended operation in remote areas.

- Exploring IoT platforms like **MQTT** or **ThingSpeak** for cloud-based data storage and visualization.

- Incorporating additional environmental sensors (e.g., humidity or air quality sensors) to expand the system’s functionality.

REFERENCES

[1]. Agaj, T., Budka, A., Janicka, E., & Bytyqi, V. (2024). Using ARIMA and ETS models for forecasting water level changes for sustainable environmental management. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-024-73405-9

[2]. Elseidi, M. (2022). Forecasting temperature data with complex seasonality using time series methods. *Modeling Earth Systems and Environment*, *9*(2), 2553–2567. https://doi.org/10.1007/s40808-022-01632-y

[3]. Khan, S., Wang, H., Nauman, U., Dars, R., Boota, M. W., & Wu, Z. (2025). Climate Impact on Evapotranspiration in the Yellow River Basin: Interpretable Forecasting with Advanced Time Series Models and Explainable AI. *Remote Sensing*, *17*(1), 115. https://doi.org/10.3390/rs17010115

[4]. Kinast, K. B., & Fokoué, E. (2024). An In-Depth Look at Rising Temperatures: Forecasting with Advanced Time Series Models in Major US Regions. *Forecasting*, *6*(3), 815-838. https://doi.org/10.3390/forecast6030041

[5]. Panigrahi, S., & Behera, H. (2017). A hybrid ETS–ANN model for time series forecasting. *Engineering Applications of Artificial Intelligence*, *66*, 49–59. https://doi.org/10.1016/j.engappai.2017.07.007

[6]. https://www.geeksforgeeks.org/c-language-introduction/

[7]. https://developers.google.com/edu/python/introduction

[8]. https://medium.com/@williamblanzeisky/predicting-temperature-data-using-arima-model-time-series-b85eb6a754cb

[9]. https://docs.python.org/3/extending/extending.html

[10]. https://docs.python.org/3/library/csv.html

[11]. https://stackoverflow.com/questions/61078280/how-to-read-a-csv-file-in-c

[12]. https://www.statsmodels.org/dev/examples/notebooks/generated/ets.html

[13]. https://www.statsmodels.org/dev/examples/notebooks/generated/statespace_sarimax_stata.htmlhttps://www.statsmodels.org/dev/examples/notebooks/generated/exponential_smoothing.html

[14]. https://www.ncei.noaa.gov/access/search/index

[15]. https://www.geeksforgeeks.org/parallel-processing-in-python/

[16]. https://www.ibm.com/think/topics/parallel-computing

[17]. https://spssanalysis.com/descriptive-statistics-in-spss/