

**Project Report**

**On**

**Unveiling-Weather-Patterns-A-Big-Data-Approach-to-Accurate-Forecasting**



Submitted in partial fulfillment for the award of

Post Graduate Diploma in Big Data Analytics (PGDBDA)

From Know IT(Pune)

**Guided by:**

Trupti Joshi Ma’am & Prasad Deshmukh Sir

**Submitted By:**

Ankit Kharwar (230943025006)

Akhil Goplani (230943025004)

Darshan Satone (230943025013)

Manish Kumar (230943025028)



**CERTIFICATE**

TO WHOMSOEVER IT MAY CONCERN

This is to certify that

Ankit Kharwar (230943025006)

Akhil Goplani (230943025004)

Darshan Satone (230943025013)

Manish Kumar (230943025028)

has successfully completed their project on

Unveiling-Weather-Patterns-A-Big-Data-Approach-to-Accurate-Forecasting

Under the guidance of Trupti Joshi Ma’am and Prasad Deshmukh sir



**ACKNOWLEDGEMENT**

This project Unveiling-Weather-Patterns-A-Big-Data-Approach-to-Accurate-Forecasting using time series Analysis was a great learning experience for us and we are submitting this work to CDAC Know IT (Pune).

We all are very glad to mention the name Trupti Joshi Ma’am and Prasad Deshmukh Sir for their valuable guidance to work on this project. Their guidance and support helped us to overcome various obstacles and intricacies during the course of project work.

We are highly grateful to Mr. Vaibhav Inamdar Manager (Know IT), CDAC, for his guidance and support whenever necessary while doing this course Post Graduate Diploma in Big Data Analytics (PGDBDA) through CDAC ACTS, Pune.

Our most heartfelt thanks go to Mr. Prasad Deshmukh (Course Coordinator, PGDBDA) who gave all the required support and kind coordination to provide all the necessities like required hardware, internet facility and extra Lab hours to complete the project and throughout the course up to the last day here in CDAC Know IT, Pune.



**TABLE OF CONTENTS**

1. ABSTRACT
2. INTRODUCTION
3. DATA COLLECTION AND FEATURES
4. SYSTEM REQUIREMENTS
   1. Software Requirements
   2. Hardware Requirements
5. FUNCTIONAL REQUIREMENTS
6. ARCHITECTURE
7. MACHINE LEARNING ALGORITHMS
8. DATA VISUALIZATION AND REPRESENTATION
9. CONCLUSION AND FUTURE SCOPE
10. REFERENCES



**Abstract**

This project presents a comprehensive approach to weather forecasting by harnessing the power of big data technologies and advanced analytical methods. Focusing on 200+ Indian cities over 20 years, we employ a sophisticated blend of time series analysis, machine learning, and Apache Spark to delve into the intricacies of weather patterns. The project begins with a robust Extract Transform & Load (ETL) process, ensuring efficient data handling and scalability. Subsequently, we explore time series analysis and machine learning techniques to unveil hidden patterns within the temporal sequences of weather data. Python serves as the primary language, offering flexibility for data manipulation, and analysis, and Tableau for visualization. By uniting machine learning models with big data technologies, our goal is to provide accurate weather predictions for selective city in our dataset.

Our project addresses this pressing need by adopting a Big Data approach to weather forecasting. In an era where technology plays a pivotal role in understanding complex systems, leveraging vast datasets through advanced analytics becomes crucial. We aim to harness the power of Big Data and employ cutting-edge machine learning algorithms to process extensive meteorological data, enabling us to unveil precise and reliable weather patterns.



**Introduction**

Weather forecasting is a complex and dynamic field that plays a pivotal role in various sectors, from agriculture to transportation. This project, titled "Unveiling Weather Patterns: A Big Data Approach to Accurate Forecasting," aims to revolutionize weather forecasting by leveraging the capabilities of big data technologies and advanced analytical methods. In this comprehensive study, we focus on over 200 Indian cities spanning two decades, employing a sophisticated blend of time series analysis, machine learning, and Apache Spark to unravel the intricacies of weather patterns.

The project commences with a robust Extract Transform & Load (ETL) process, ensuring efficient data handling and scalability. Through this initial phase, we lay the foundation for a thorough exploration of weather data, setting the stage for in-depth analysis. Subsequently, we delve into time series analysis and machine learning techniques, unveiling hidden patterns within the temporal sequences of weather data. Python serves as our primary language, offering the flexibility required for seamless data manipulation and analysis. Furthermore, we utilize Tableau for visualization, enhancing the interpretability of our findings.

By uniting machine learning models with cutting-edge big data technologies, our overarching goal is to provide accurate weather predictions for selected cities within our extensive dataset. This project not only contributes to the advancement of weather forecasting methodologies but also underscores the significance of integrating data science approaches to enhance prediction accuracy in a field crucial to numerous industries.

**Dataset Collection and Features**



We successfully collected a substantial dataset from NASA's prediction of worldwide energy resources through their data access viewer (<https://power.larc.nasa.gov/data-access-viewer/>). This dataset spans a 20-year period for over 200+ cities in India and is an extensive compilation of valuable information totaling approximately 16 gigabytes. The collected data is rich in meteorological features, providing a comprehensive understanding of the atmospheric conditions.

The inclusion of key parameters such as air temperature, dew/frost temperature, wet bulb temperature, surface pressure, wind speed, specific humidity, relative humidity, and corrected precipitation allows for a holistic analysis of atmospheric conditions. These insights are crucial for various applications, including energy resource planning, climate research, and environmental impact assessments.

Additionally, we express our gratitude to NASA for providing this valuable dataset, acknowledging their significant role in advancing scientific understanding and fostering collaborative efforts for the greater good.

Here is a brief overview of the key features present in the dataset:

* Date & Time (YYYYMMDDHH):

Represents the timestamp in the format of Year, Month, Day, and Hour, facilitating time-series analysis.

* T2M (Average Air Temperature at 2 meters):

Signifies the mean air temperature at 2 meters above the Earth's surface, a critical parameter for assessing overall climate conditions.

* T2MDEW (Average Air DEW/Frost Temperature at 2 meters):

Represents the average air dew or frost temperature at 2 meters above the Earth's surface, providing insights into humidity and potential frost formation.

* T2MWET (Average Air WET Bulb Temperature at 2 meters):

Indicates the average wet bulb temperature at 2 meters above the Earth's surface, offering a measure of humidity and potential cooling effects.

* PS (Surface Pressure):

Reflects the atmospheric pressure at the Earth's surface, a crucial parameter for understanding weather patterns and conditions.

* PSC (Corrected Atmospheric Pressure):

Represents the corrected atmospheric pressure, providing a more accurate measure for atmospheric analysis.



* WS2M (Wind Speed at 2 meters):

Denotes the wind speed at 2 meters above the Earth's surface, essential for assessing local wind patterns and potential energy resources.

* QV2M (Specific Humidity at 2 meters):

Represents the specific humidity at 2 meters above the Earth's surface, aiding in understanding moisture content in the atmosphere.

* RH2M (Relative Humidity at 2 meters):

Indicates the relative humidity at 2 meters above the Earth's surface, a critical factor in assessing comfort levels and potential precipitation.

* PRECTOTCORR (Precipitation Corrected):

Signifies the corrected precipitation values, accounting for any discrepancies in the initial precipitation data.

This extensive dataset, gathered through the challenging process of utilizing NASA's API, provides a valuable resource for researchers and analysts interested in studying and predicting meteorological conditions for energy resource planning in the diverse regions of India over a significant time span.





**System Requirement**

* Hardware Requirements:

Computer: Ensure your computer has sufficient processing power and memory to execute data processing and analysis tasks effectively. A modern multicore processor and at least 16 GB of RAM are recommended for handling the complexities of our project.

* Storage: Allocate ample storage space for storing the generated dataset and any additional datasets if required. Utilizing an SSD (Solid State Drive) is advisable to facilitate faster data access, particularly when managing large volumes of structured and unstructured data.
* Internet Connection: A stable internet connection is critical for downloading and installing necessary software packages, libraries, and accessing online resources essential throughout our project.
* Software Requirements:
* Operating System: Our project relies on Python (3.10.12) as the primary programming language. It's also essential to mention that the operating system used is Linux.
* Python: Python is the backbone of our project, serving as the main programming language for data generation, analysis, and machine learning tasks.
* PySpark: PySpark (3.4.0) plays a crucial role in preprocessing our downloaded data, initially in a complex JSON unstructured format. We use PySpark to convert this data into a structured format by defining a schema. The processed data is then stored in MongoDB.
* MongoDB: MongoDB (6.0.13) is employed as a database to store the structured data resulting from the preprocessing phase. MongoDB's flexibility with JSON-like documents aligns well with the nature of our project's data.
* Airflow: Airflow (2.8.0) is utilized for scheduling workflows (DAGs) in our project. This automation and orchestration tool enhances the efficiency and reliability of our data processing pipeline.
* Gradio: Gradio (4.19.1) is included in our project to simplify the deployment and interaction with machine learning models through user interfaces. Specific details about how Gradio is utilized in our project need to be added.
* Machine Learning Libraries (stats models 0.14.1): Machine learning components in our project, such as ARIMA and SARIMAX, are implemented using the stats models library (0.14.1). SARIMAX is particularly chosen for training due to the seasonal nature of our data.

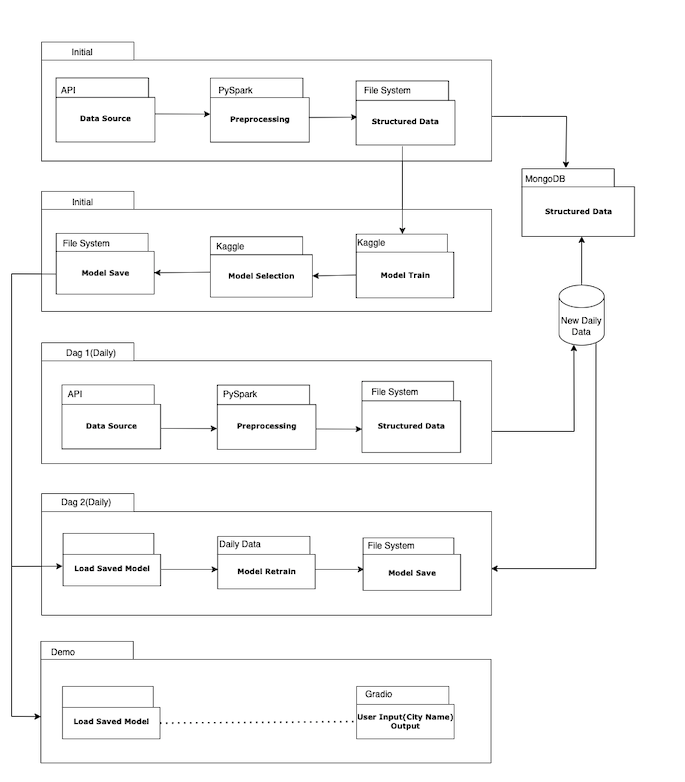


* Kaggle: Our machine learning models are trained on Kaggle, leveraging its platform to make efficient use of resources. The collaborative community and diverse datasets available on Kaggle enhance the training process.
* Tableau Public: Tableau Public remains part of our project's software stack, likely used for data visualization and creating interactive dashboards to convey insights derived from our analysis.

This comprehensive set of hardware and software requirements showcases the robust infrastructure and tools employed in our project, emphasizing efficiency and scalability in handling large datasets and complex data processing tasks. The adjustment to 16 GB of RAM reflects the optimization made during the project, maintaining effective performance while conserving resources.



**Architecture**





The architectural workflow for our system is designed to efficiently handle data retrieval, preprocessing, and analysis, ensuring a seamless flow of information from user input to meaningful insights. The various components collaborate to create a robust and dynamic system:

* User Input and Initial API Request:

The process begins with the API receiving a request containing user input, typically a city name. This request is sent to the data source.

* Data Retrieval and Initial File System Storage:

The data source retrieves relevant information based on the user input and saves it in the initial file system. This serves as the foundational dataset for subsequent analysis.

* Machine Learning Model Training and File System Storage:

The initial dataset is utilized to train a machine learning model, and the trained model is saved in the file system. This model acts as the baseline for generating predictions.

* Daily Data Retrieval and PySpark Preprocessing:

On a daily basis, the API retrieves the latest data from the data source. Using PySpark, the data is preprocessed to ensure it aligns with the required structure for further analysis.

* Kaggle-Based Model Training:

The preprocessed data is then used to train a machine learning model on Kaggle, harnessing the platform's resources and collaborative capabilities for effective model training.

* Model Loading and Prediction Generation:

The trained model is loaded, and predictions are generated on the preprocessed data. These predictions form the basis of the insights provided to the user.

* User Interface with Gradio:

Gradio serves as the user interface, allowing users to conveniently input their requests and visualize the predictions in an accessible manner.

* Structured Data Storage in MongoDB:

The structured data, including both input and output, is saved in MongoDB. This database provides a scalable and flexible solution for storing data in a JSON-like format.



* Daily Data Processing and File System Update:

On a daily basis, new data is processed and saved in the file system. This ensures that the system is constantly updated with the latest information for ongoing analysis and model training.

* Model Retraining and File System Update:

The saved data is used to retrain the machine learning model, and the retrained model is stored in the file system. This iterative process ensures that the model stays current and relevant.

* Structured Data Storage in File System:

The structured data resulting from the daily processing is saved in the file system, facilitating easy access and future analysis.

* Trained Model Storage in File System:

The trained model, now updated through regular retraining, is stored in the file system, ensuring its availability for predictions and analysis.

This comprehensive workflow underscores the synergy of APIs, data sources, file systems, machine learning models, and user interfaces in creating a system that efficiently processes and analyzes vast amounts of data to provide meaningful and up-to-date insights to the user. The integration of MongoDB and file systems ensures robust data storage and accessibility for ongoing analysis and model training. The daily updates and retraining processes contribute to the system's adaptability and responsiveness to the ever-changing data landscape.



**Algorithms**

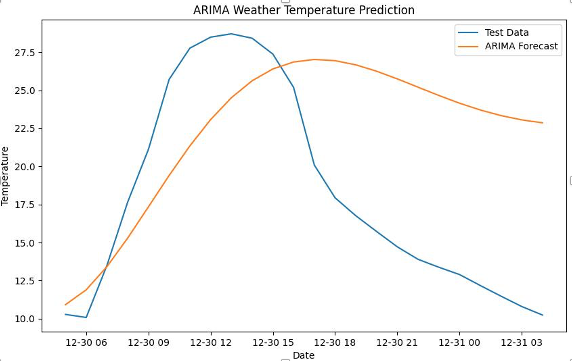
**AutoRegressive Integrated Moving Average (ARIMA) :**

The AutoRegressive Integrated Moving Average (ARIMA) model is a widely used statistical tool for time-series forecasting, including weather temperature prediction. The model combines autoregression, differencing, and moving average components to analyze and predict future values based on historical data.

However, the ARIMA model does have certain limitations. One major disadvantage is its restricted ability to capture long-term trends and seasonal variations, particularly in complex and nonlinear weather systems. Weather patterns can change over time due to various factors such as global warming, El Niño, or other climate phenomena, which may not be adequately accounted for in the ARIMA model.

Additionally, ARIMA models assume linearity in the relationships between variables, while weather systems can be highly nonlinear and dynamic. This limitation may result in inaccurate predictions, especially when attempting to forecast extreme weather events or significant shifts in temperature patterns.

Another challenge with ARIMA models is the need for a stationary time series, which means that the statistical properties, such as mean and variance, should remain constant over time. In practice, weather data can be non-stationary, requiring differencing or other transformations to stabilize the series. These transformations can introduce additional complexity and potential errors in the model.





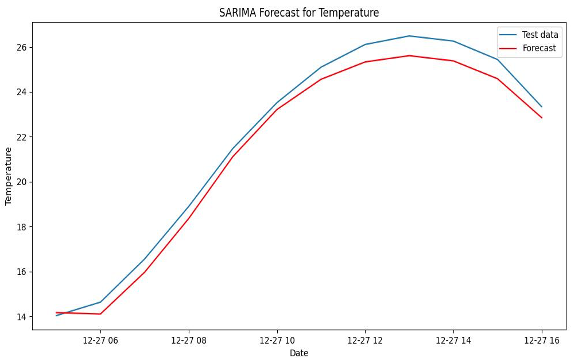
The Root Mean Square Error (RMSE) is a crucial metric for evaluating the accuracy of a forecasting model, including the ARIMA model for weather temperature prediction. An RMSE of 8.1337 indicates that, on average, the model's predictions are approximately 8.1337 units away from the actual observed values.

In the context of weather temperature prediction, an RMSE of 8.1337 implies that the model's forecasts may deviate from the true temperatures by around 8 degrees Celsius. While this might be acceptable for some applications, it is essential to aim for lower RMSE values to improve the model's accuracy.

**Seasonal Autoregressive Integrated Moving Average (SARIMAX)**

SARIMAX (Seasonal Autoregressive Integrated Moving Average with exogenous variables) is a powerful time series forecasting model that builds upon the ARIMA model by incorporating seasonal components and exogenous variables. This extension makes SARIMAX particularly effective for capturing complex seasonal patterns and accounting for external factors that may influence the variable of interest, such as weather temperature.

In the context of weather temperature prediction, SARIMAX can account for various seasonal patterns, such as daily, weekly, or annual cycles, and incorporate external factors like humidity, pressure, or wind speed. By considering these additional components, SARIMAX can provide more accurate and reliable forecasts compared to the basic ARIMA model.



The Root Mean Square Error (RMSE) of 0.6172 for the SARIMAX model indicates that, on average, the model's predictions are approximately 0.6172 units away from the actual observed values. This RMSE value is relatively low, suggesting

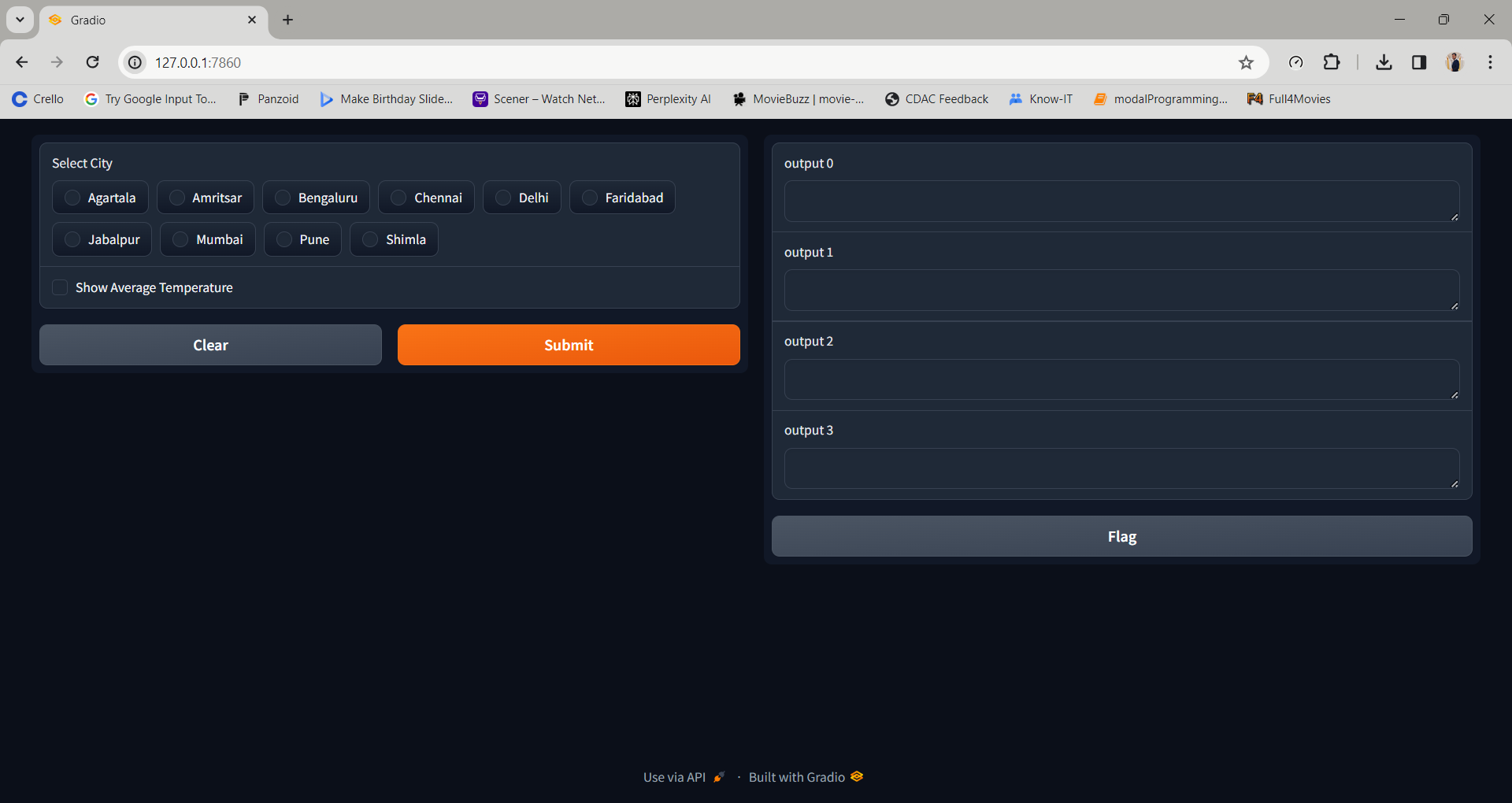


that the SARIMAX model's forecasts are quite accurate in predicting weather temperatures.

Achieving an RMSE of 0.6172 demonstrates the model's ability to effectively capture both seasonal patterns and the influence of external factors on temperature. This improvement in accuracy can significantly benefit various applications, such as energy management, agriculture, or transportation planning, by providing more precise and reliable weather forecas

Here's how it works:



[](http://127.0.0.1:7860/)



DATA VISUALIZATION AND REPRESENTATION



**Conclusion And Future Scope**

In conclusion, our project represents a sophisticated and comprehensive system for predicting and analyzing meteorological data across 200+ cities in India. Leveraging NASA's prediction of worldwide energy resources, we successfully collected and processed a vast dataset spanning two decades. The integration of Python, PySpark, MongoDB, Kaggle, Airflow, Gradio, and Tableau Public facilitated a robust and scalable infrastructure for data retrieval, preprocessing, machine learning modeling, and user interaction.

The combination of PySpark and Kaggle played a pivotal role in efficiently handling the complex and unstructured initial data, enabling us to derive valuable insights through machine learning models. Gradio provided an intuitive interface for users to interact with the system, making it accessible and user-friendly. The use of MongoDB ensured structured data storage and retrieval, while Tableau Public enhanced the visualization of our findings.

* Future Scope:

The project lays a solid foundation for future enhancements and expansions. Here are some potential avenues for future development:

* Enhanced Machine Learning Models:

Explore advanced machine learning algorithms and models to further improve prediction accuracy and robustness. Investigate deep learning approaches to capture intricate patterns in the meteorological data.

* Real-time Data Processing:

Integrate real-time data processing capabilities to provide users with the most up-to-date information. This could involve adopting streaming analytics technologies and continuous model training.

* Geospatial Analysis:

Incorporate geospatial analysis to enhance insights by considering the geographical context of the meteorological data. This could lead to more localized and accurate predictions.

* User Personalization:

Implement user personalization features to tailor the presented insights based on individual preferences or specific areas of interest. This could enhance user engagement and satisfaction.

* Integration with Additional Data Sources:

Expand the project's scope by integrating data from additional sources, such as satellite imagery or other meteorological databases. This diversification could lead to a more comprehensive understanding of environmental conditions.

* Optimization of Workflow:

Continuously optimize the workflow, considering advancements in technology and methodologies. This includes exploring new tools and techniques for data processing, machine learning, and user interaction.

* Collaborative Features:

Introduce collaborative features, allowing users to share insights, analyses, and visualizations. This could foster a community-driven approach to understanding and interpreting meteorological data.

* Mobile Application Development:

Develop a mobile application to extend the accessibility of the system, enabling users to access predictions and insights on the go. Consider user-friendly design principles for a seamless mobile experience.

* Environmental Impact Assessment:

Extend the project to include features for assessing the environmental impact of meteorological conditions. This could be particularly useful for industries and policymakers in making informed decisions.

* Incorporate Feedback Mechanisms:

Implement mechanisms for users to provide feedback on predictions and analyses. This feedback loop could be valuable for continuously improving the accuracy and relevance of the system.

In essence, the project not only provides a comprehensive solution for meteorological data analysis but also serves as a dynamic platform with numerous possibilities for future enhancements and innovations. The evolving



landscape of technology and data science presents exciting opportunities to further refine and expand the system's capabilities

**References**



Apache Spark. [https://spark.apache.org/]

MongoDB. [https://[www.mongodb.com/]](http://www.mongodb.com/)

Kafka. [https://kafka.apache.org/]

Python. [[https://ww](http://www.python.org/)w.py[thon.](http://www.python.org/)org/]

scikit-learn. [https://scikit-learn.org/]

The data has been taken from NASA prediction of worldwide Energy Resources <https://power.larc.nasa.gov/data-access-viewer/>