

# Project Report

**On**

Weather Prediction For Pune Region



Submitted in partial fulfilment for the award of

Post Graduate Diploma in Big Data Analytics (PGDBDA)

From Know IT(Pune)

## Guided by:

Mrs. Trupti Joshi & Mr. Prasad Deshmukh

Submitted By:

Ameya Pardhi (240343025004)

Sakshi Bidwai (240343025010)

Dhanvantari Warekar (240343025012)

Gagan Bajoria (240343025014)



# CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that

Ameya Pardhi (240343025004)

Sakshi Bidwai (240343025010)

Dhanvantari Warekar (240343025012)

Gagan Bajoria (240343025014)

Have successfully completed their project on

Weather Prediction For Pune Region

Under the guidance of Mrs. Trupti Joshi and Prasad Deshmukh Sir



# ACKNOWLEDGEMENT

This project **Weather Prediction For Pune Region** 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 Mrs. Trupti Joshi and Mr. Prasad Deshmukh 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

ABSTRACT

1. INTRODUCTION
2. DATA COLLECTION AND FEATURES
3. SYSTEM REQUIREMENTS
   1. Software Requirements
   2. Hardware Requirements
4. FUNCTIONAL REQUIREMENTS
5. ARCHITECTURE
6. ALGORITHMS & MODELS
7. RESULT
8. CONCLUSION AND FUTURE SCOPE
9. REFERENCES



# ABSTRACT

The project involved developing a sophisticated weather prediction system specifically for the Pune region, employing advanced data processing and machine learning techniques. Apache Spark was utilized to handle efficient Extract, Transform, Load (ETL) processes, ensuring swift and reliable data manipulation, while the processed data was stored in Hadoop Distributed File System (HDFS) to provide scalable and fault-tolerant storage solutions. This robust framework facilitated the implementation of machine learning algorithms such as Random Forest, XGBoost, and Decision Tree to predict critical weather parameters with high accuracy. The system's reliable predictions played a crucial role in supporting various domains including agricultural planning, disaster management, and daily weather-related decision-making, thereby enhancing the region's resilience and preparedness for weather-related challenges.



# INTRODUCTION

**1. INTRODUCTION**

In an era where data is generated at unprecedented rates, accurate weather prediction has become increasingly important for various sectors, including agriculture, transportation, and disaster management. Leveraging large datasets and advanced computational tools, modern weather prediction systems aim to improve forecasting accuracy and provide timely insights.

One of the key challenges in weather prediction is handling the sheer volume and complexity of meteorological data. Traditional methods often struggle to process and analyze such large datasets efficiently. This is where Apache Spark, a distributed computing framework, comes into play. Spark allows for scalable data processing and analysis, making it well-suited for tackling large-scale weather datasets.

This project investigates the use of Apache Spark in weather prediction by analyzing historical weather data and building models to forecast future weather conditions. The project aims to demonstrate how Spark's distributed computing capabilities can be harnessed to improve the efficiency and accuracy of weather predictions.

**Objectives**: The primary objectives of this report are:

* To explore the effectiveness of Apache Spark in processing and analyzing large-scale weather datasets.
* To build and evaluate predictive models for weather forecasting using Spark's machine learning libraries.
* To assess the performance of different data processing and machine learning techniques in predicting weather patterns



# Dataset Collection and Features

## Data Sources

For our project, we utilized the data on Kaggle which is a public platform to access varieties of datasets. The dataset used for this project was sourced from weather data specific to the Pune region. This comprehensive dataset contains detailed weather records that include various weather parameters measured at hourly intervals. The dataset is particularly useful for training machine learning models aimed at predicting weather conditions and supporting decision-making processes in agriculture, disaster management, and daily planning.

## Dataset Size

The dataset comprises a total of 52,583 records, each representing weather data collected at an hourly frequency. This dataset features multiple weather parameters for each recorded time point.

* Date\_time
* maxtempC & mintempC
* sunHour
* uvIndex
* moon\_illumination
* moonrise & moonset
* sunrise & sunset
* WindChillC
* WindGustKmph & WindSpeedKmph
* Cloudcover
* Humidity
* precipMM
* pressure
* tempC
* visibility
* windrrDegree

The dataset is split into two parts: train, test

* Train: 42,066 rows
* Test: 10,517 rows



## Features/Attributes

## • Maximum Temperature (maxtempC): Indicates the highest temperature reached.

## • Minimum Temperature (mintempC): Indicates the lowest temperature recorded.

## • Temperature (tempC): Current temperature at the time of recording.

## • Wind Chill (WindChillC): Perceived temperature based on wind.

## • Wind Gust (WindGustKmph): Speed of wind gusts.

## • Wind Direction (winddirDegree): Direction from which the wind is blowing.

## • Wind Speed (windspeedKmph): Speed of the wind.

## • Visibility (visibility): Distance at which an object can be clearly discerned.

## • Cloud Cover (cloudcover): The fraction of the sky covered by clouds.

## • Precipitation (precipMM): Amount of precipitation.

## • Humidity (humidity): Amount of water vapor present in the air.

## • Pressure (pressure): Atmospheric pressure.

## • UV Index (uvIndex): Intensity of ultraviolet radiation.

## • Sunshine Hours (sunHour): Duration of sunshine.

## • Sunrise/Sunset: Times of sunrise and sunset.

## • Moonrise/Moonset: Times of moonrise and moonset.

## • Moon Illumination (moon\_illumination): Percentage of the moon illuminated



# 3. SYSTEM REQUIREMENTS

**Hardware Requirements**

1. **Computer**: A machine with adequate processing power and memory is essential for handling large-scale data processing and analysis tasks using Apache Spark. A modern multicore processor with at least 16 GB of RAM is recommended to efficiently run Spark jobs and manage distributed data processing.
2. **Storage**: Sufficient storage space is necessary for storing weather datasets, intermediate files, and results. Given the potential volume of data, an SSD (Solid State Drive) is preferable for faster data access and processing.
3. **Internet Connection**: A stable internet connection is crucial for downloading and installing Spark, related libraries, and any necessary datasets from online sources. It also ensures smooth connectivity if working with cloud-based Spark clusters.

**Software Requirements**

1. **Operating System**: Windows 10 or higher, macOS, or Linux. Spark is cross-platform, so ensure your OS supports the necessary software installations.
2. **Apache Spark**: Install Apache Spark for distributed data processing and analysis. Ensure you have Java installed as it is required by Spark.
3. **Python**: Python is used for scripting and data manipulation. Ensure Python is installed on your system, ideally version 3.7 or higher.
4. **Python Libraries**: Install the following Python libraries and dependencies using package managers like pip:
   * **PySpark**: To interface with Apache Spark using Python.
   * **NumPy**: For numerical computing.
   * **Pandas**: For data manipulation and preparation before and after Spark processing.
   * **Matplotlib and Seaborn**: For data visualization.
   * **SciPy**: For scientific computing tasks that complement Spark's capabilities.
5. **Jupyter Notebook**: For interactive coding, data exploration, and project documentation. Ensure Jupyter is configured to work with Spark.



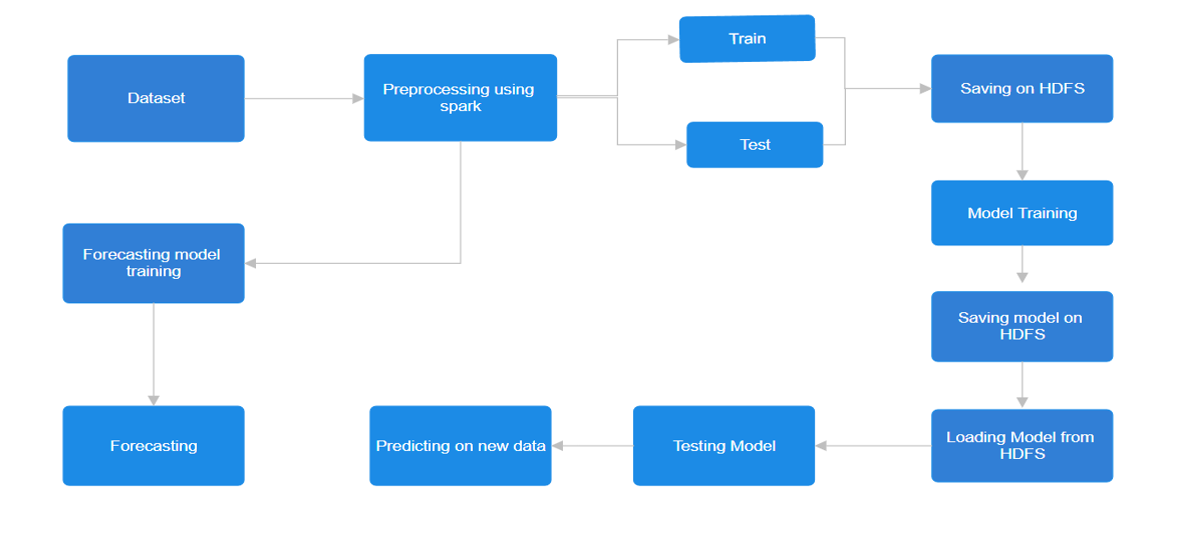
# 4. FUNCTIONAL REQUIREMENTS

**Python 3**:

* **Python** is a versatile, high-level programming language that plays a crucial role in scripting and data manipulation within the project.
* In this project, Python is primarily used to interface with Apache Spark via the PySpark library, enabling large-scale data processing and analysis for weather prediction.
* Python's extensive library ecosystem allows the project to easily integrate various data processing, machine learning, and visualization tasks, ensuring the focus remains on core functionality.
* **PySpark** enables Python to leverage Spark's distributed computing capabilities, making it possible to handle vast amounts of weather data efficiently.
* Python's simplicity and readability, combined with the power of Spark, make it an ideal choice for this project's computational needs.



# 5. ARCHITECTURE:



**Fig: System Architecture of Weather Prediction**

* Extracting data from Public Source such as Kaggle for the train, test and prediction dataset.
* Data preprocessing by using Spark.
* Saving data on HDFS.
* Training the model using train data.
* Saving the trained model on HDFS.
* Trained the model using ML algorithms such as Linear Regression, Decision Tree,Random Forest.
* Predicted outputs.



# 6. ALGORITHMS & MODELS

**6.1 Linear Regression:**

**Algorithm Overview:**

Linear regression attempts to model the relationship between a dependent variable y and one or more independent variables X=[x1,x2,...,xn] by fitting a linear equation to the observed data. The goal is to minimize the difference between the predicted and actual values of y, often using the least squares method.

## Mathematical Formulation:

The linear regression model can be expressed as: y = β0 + β1x1 + β2x2 +...+ βnxn + ϵ​, where β0​ is the intercept, β1,β2,...,βn​ are the coefficients for the features, and ϵ epresents the error term. The objective is to minimize the sum of squared residuals (SSR):

SSR=∑i=1n​(yi​−y^​i​)2

**Technical Aspects:**

* **Gradient Descent:** To find the optimal coefficients, gradient descent can be used, which iteratively adjusts the coefficients to minimize the cost function (e.g., MSE).
* **Regularization:** Regularization techniques like Lasso (L1) and Ridge (L2) are often employed to prevent overfitting by adding a penalty term to the cost function.

**Hyperparameters Used:**

* **regParam (λ\lambda):** Controls the regularization strength, balancing the trade-off between fitting the training data and generalizing to new data.
* **elasticNetParam (α\alpha):** Determines the mix of L1 and L2 regularization. When α=0, the model behaves like Ridge regression, and when α=1, it behaves like Lasso regression.



**6.2 Decision Tree**

**Algorithm Overview:**

Decision Trees make predictions by recursively splitting the data into subsets based on the value of the input features. Each internal node of the tree represents a decision rule on a feature, and each leaf node represents the predicted outcome. The model uses a top-down, greedy approach known as Recursive Binary Splitting.

**Mathematical Formulation:**

The tree is built by selecting the best feature to split on at each node. The "best" split is determined by minimizing a criterion such as:

• Gini Impurity (for classification):

Gini Impurity = 1 - Σ(pᵢ)², where pᵢ is the proportion of samples belonging to class i.

• Mean Squared Error (for regression):

MSE = (1/n) \* Σ(yi - ŷi)², where n is the number of samples, yi is the actual value, and ŷi is the predicted value.

**Technical Aspects:**

* Greedy Splitting: The tree splits the dataset at each node based on the feature that results in the most significant information gain, measured by the reduction in impurity (e.g., Gini or MSE).
* Pruning: To prevent overfitting, decision trees often employ pruning techniques that limit the depth of the tree or remove branches that add little predictive power.

**Hyperparameters:**

* maxDepth: Limits the maximum depth of the tree, controlling the model's complexity and preventing overfitting by constraining the number of splits.
* minInfoGain: Sets the minimum threshold for information gain required to split a node. This ensures that only significant splits are considered.

.



**6.3 Random Forest**

**Algorithm Overview:**

Random Forest is an ensemble learning method that builds multiple decision trees and combines their predictions through averaging (for regression) or voting (for classification). It leverages both bagging and random feature selection to reduce variance and improve generalization.

**Mathematical Formulation:**

For regression tasks, the prediction is the average of the predictions from all the trees:

ŷ = (1/T) \* Σ (from t=1 to T), where T is the number of trees, and ŷ is the prediction from the tree.

**Technical Aspects:**

* **Bagging:** Random Forest uses bootstrap samples, which involve training each tree on a different subset of the data, thereby reducing overfitting by introducing variability.
* **Random Feature Selection:** At each node, a random subset of features is selected, and the best split is chosen from this subset. This reduces the correlation between trees, enhancing the ensemble's robustness.

**Hyperparameters:**

* **numTrees:** Specifies the number of trees in the forest. A larger number of trees typically results in better performance, but increases computational cost.
* **maxDepth:** Controls the maximum depth of each tree in the forest. Balancing depth and the number of trees is crucial for optimizing both accuracy and efficiency.



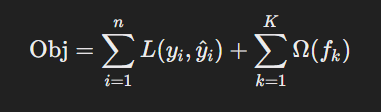
**6.4 XGBoost**

**Algorithm Overview:**

XGBoost (Extreme Gradient Boosting) is an advanced implementation of gradient boosting that focuses on speed and performance. It sequentially adds trees to the model, where each tree attempts to correct the errors made by the previous ones. The model optimizes a differentiable loss function and includes regularization terms to prevent overfitting.

**Mathematical Formulation:**

The objective function for XGBoost is:



Here:

* L(yi,y^i) is the loss function, which measures the difference between the actual value yi ​ and the predicted value y^i ​. For regression tasks, this could be the Mean Squared Error (MSE).
* Ω\Omega is a regularization term that penalizes the complexity of the model, helping to prevent overfitting.

**Technical Aspects:**

* **Tree Additive Model:** XGBoost builds trees sequentially, where each new tree models the residual errors of the previous trees.
* **Shrinkage:** A learning rate is applied to the contribution of each tree, reducing the risk of overfitting by making smaller updates to the model.
* **Regularization:** XGBoost introduces L1 and L2 regularization terms to control the complexity of the trees and improve generalization.

**Hyperparameters:**

* **max\_depth:** Defines the maximum depth of the trees, controlling the model's complexity and ability to capture intricate patterns.
* **learning\_rate (η\etaη):** Controls the step size at each iteration. Lower values reduce overfitting but require more trees to converge.
* **gamma:** A regularization parameter that specifies the minimum loss reduction required to make a split. Higher values prevent the model from making unnecessary splits, thus simplifying the model



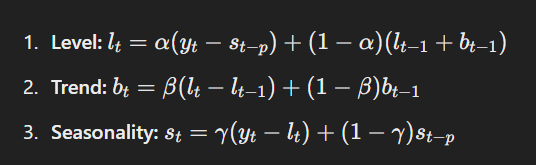
**6.5 Holt-Winters**

**Algorithm Overview:**

The Holt-Winters method is a time series forecasting technique that extends exponential smoothing by incorporating components for level, trend, and seasonality. It is particularly effective for data with seasonal patterns and trends. The method can handle both additive and multiplicative seasonality.

**Mathematical Formulation:**

The Holt-Winters method consists of three equations for updating the level, trend, and seasonal components:



**Technical Aspects:**

* **Initialization:** The initial values for level, trend, and seasonality must be estimated, often using historical data.
* **Exponential Smoothing:** The method applies exponential smoothing to the level, trend, and seasonal components, ensuring that recent observations are given more weight.

**Hyperparameters:**

* **trend (additive/multiplicative):** Determines how the trend component is modeled. Additive trends assume a constant change over time, while multiplicative trends assume proportional changes.
* **seasonal (additive/multiplicative):** Defines the seasonality type. Additive seasonality assumes constant seasonal effects, while multiplicative seasonality adjusts the seasonal effects relative to the level of the series.
* **seasonal\_periods:** Specifies the length of the seasonal cycle, such as 12 for monthly data with yearly seasonality.



1. **RESULT**

**7.1 Predictive Model :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Before Hyperparameters** | | | **After Hyperparameters** | | |
| **MAE** | **RMSE** | **R-Sqr** | **MAE** | **RMSE** | **R-Sqr** |
| **Linear Regression** | **0.960** | **2.020** | **0.800** | **0.960** | **2.030** | **0.800** |
| **Decision Tree** | **0.900** | **1.910** | **0.820** | **0.900** | **1.910** | **0.820** |
| **Random Forest** | **0.830** | **1.890** | **0.830** | **0.610** | **1.760** | **0.850** |
| **XGBoost** | **0.620** | **1.820** | **0.840** | **0.590** | **1.730** | **0.857** |

**7.2 Forecasting Model :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Before Hyperparameters** | | | **After Hyperparameters** | | |
| **AIC** | **MAE** | **RMSE** | **AIC** | **MAE** | **RMSE** |
| **Holt-Winter(Additive)** | **-25596** | **6.3** | **7.4** | **-25589** | **6.7** | **7.9** |
| **Holt-Winter(Multiplicative)** | **-26151** | **6.5** | **7.8** | **-26157** | **5** | **6.1** |



**8. CONCLUSION AND FUTURE SCOPE**

**Conclusion:**

* **Successful Model Development**: Developed a machine learning model that forecasts temperatures with reasonable accuracy based on historical weather data.
* **Data Processing and Analysis**: Demonstrated effective data preprocessing, feature selection, and model training techniques.
* **Valuable Insights**: Provides valuable insights into future temperature trends in Pune.
* **Impact on Stakeholders**: Aims to assist stakeholders in climate-sensitive sectors such as agriculture, tourism, and infrastructure development with improved decision-making.
* **Addressing Climate Challenges**: Contributes to efforts in mitigating the impacts of climate variability and advancing understanding of regional climate dynamics.

Overall, the Pune temperature prediction project lays the groundwork for ongoing research and collaboration in the field of climate science and machine learning, contributing to efforts aimed at addressing the challenges posed by climate variability and change in the region.



**Future Scope:**

* **High-Resolution Data Integration**: Integrate high-resolution data from various sources for improved prediction accuracy.
* **ETL Processing with Apache Airflow**: Use Apache Airflow to streamline ETL processes and improve data pipeline management.
* **Advanced Feature Engineering and Deep Learning**: Explore further feature engineering and implement deep learning architectures for enhanced model performance.
* **Uncertainty Estimation and Localized Predictions**: Develop methods for estimating uncertainty and making localized predictions.
* **Decision Support Integration**: Integrate the model with decision support systems for urban planning and disaster management.



# 9.References

1. Weather Forecast Prediction: An Integrated Approach for Analyzing and Measuring Weather Data, 10.5120/ijca2018918265, Munmun Biswas, Tanni Dhoom, Sayantanu Baru

[https://www.researchgate.net/publication/329922758\_Weather\_Forecast\_Prediction\_An\_Integrated\_Approach\_for\_Analyzing\_and\_Measuring\_Weather\_Data]

1. Weather Prediction Using Machine Learning Algorithms,

10.1109/ICICCSP53532.2022.9862337, Aishwaria Shaji, A. R. Amritha, V. R. Rajalakshmi

[https://ieeexplore.ieee.org/document/9862337]

1. Kaggle

[https://www.kaggle.com/datasets/tony16ishere/pune-weather-dataset]