**A Case Study Report**

On

**WEATHER FORECAST PREDICTION**

Submitted for partial fulfilment of the requirements for the award of the degree of

## BACHELOR OF ENGINEERING

in

## INFORMATION TECHNOLOGY

by

**N. RAVI TEJA** (2451-21-737-147)



## MATURI VENKATA SUBBA RAO (M.V.S.R) ENGINEERING COLLEGE

## (An Autonomous Institution)

Department of Information Technology

(Affiliated to Osmania University & Recognized by AICTE)

Nadergul, Saroor Nagar Mandal, Hyderabad – 501 510

AY : 2023-24

i



## CERTIFICATE

*This is to certify that the case study entitled* ***“*WEATHER FORECAST PREDICTION*”*** *is a bonafide work carried out by Mr.N. RAVI TEJA (2451-21-737-147) in partial fulfilment of the requirements for the award of degree of Bachelor of Engineering in Information Technology from Maturi Venkata Subba Rao (M.V.S.R.) Engineering College, an Autonomous Institution, affiliated to Osmania University Hyderabad.*

*The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.*

**Case Study Co-ordinator**  **HOD**

**Mrs . Sita Sowjanya** **Dr. K. Venugopal Rao**

Professor, ITD Dean – Academics,

Professor & Head - ITD

### 

ii

## DECLARATION

This is to certify that the work reported in the present ML Case Study report entitled “**WEATHER FORECAST PREDICTION**” is a record of bonafide work done by us in the Department of Information Technology, Maturi Venkata Subba Rao (M.V.S.R.) Engineering College, an Autonomous Institution, affiliated to Osmania University. The reports are based on the case study done entirely by us and not copied from any other source. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

**N. RAVI TEJA** (2451-21-737-147)

iii

## 

## ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our Case Study Co-ordinator, Mrs. Sita Sowjanya, professor for her valued recommendations throughout the course.

We are also thankful to our Principal **Dr . G. Kanaka Durga**, **Professor** and **Dr. K.**

**Venugopal Rao,** Professor and Head - Department of Information Technology, Dean Academics, Maturi Venkata Subba Rao Engineering College, for providing excellent infrastructure and a nice atmosphere to carry out case study on the training obtained through the machine learning lab.

Finally, we would like to take this opportunity to thank our family for their support through the work. We sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this work.

**N.RAVI TEJA** (2451-21-737-147)

iv

## VISION & MISSION, PROGRAM EDUCATIONAL OBJECTIVES

**Vision of the Department:**

To impart technical education producing competent and socially responsible engineering professionals in the field of Information Technology.

**Mission of the Department:**

M1. To make teaching learning process effective and stimulating.

M2. To provide adequate fundamental knowledge of sciences and Information Technology with positive attitude.

M3. To create an environment that enhances skills and technologies required for industry.

M4. To encourage creativity and innovation for solving real world problems.

M5. To cultivate professional ethics in students and inculcate a sense of responsibility towards society

**Program Educational Objectives**:

After 3 to 4 years of graduation, graduates of the Information Technology program will:

1. Apply knowledge of mathematics and Information Technology to analyze, design and implement solutions for real world problems in core or in multidisciplinary areas.
2. Communicate effectively, work in a team, practice professional ethics and apply knowledge of computing technologies for societal development.
3. Engage in Professional development or postgraduate education to be a life-long learner.

v

**PROGRAM OUTCOMES & PROGRAM SPECIFIC OUTCOMES**

## PROGRAM OUTCOMES (POs)

**Engineering Graduates will be able to:**

**Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

1. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
2. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
3. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
4. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
5. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

1. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
2. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
3. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
4. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
5. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one‟s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
6. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

**PROGRAM SPECIFIC OUTCOMES (PSOs):**

PSO1: **Hardware design**: An ability to analyze, design, simulate and implement computer and hardware/software and use basic analog/digital circuits, VLSI design for various computing and communication system applications.

PSO2:S**oftware design**: An ability to analyze a problem, design algorithm, identify and define c computing requirements appropriate to its solution and implement the same.

vi

## COURSE OBJECTIVES & COURSE OUTCOMES

**COURSE OBJECTIVES**

Course Objectives:

⮚ To demonstration of different classifiers on different data.

⮚ To demonstrate unsupervised learning algorithms.

⮚ To demonstrate dimensionality reduction techniques.

⮚ To make use of real world data to implement machine learning models.

Course Outcomes:

CO1: Apply machine learning algorithms: data set preparation, model selection, model building etc .

CO2: Evaluate various Machine Learning approaches.

CO3: Use scikit-learn, Keras and Tensor flow to apply ML techniques. CO4: Design and develop

Solutions to real world problems using ML techniques.

CO5: Apply unsupervised learning and interpret the results.

vii

## 

## ABSTRACT

Weather forecasting is essential for various sectors like agriculture, transportation, and disaster management. Traditional methods, while effective, are computationally demanding. This study investigates the use of machine learning (ML) techniques to improve weather prediction accuracy and efficiency. Utilizing historical weather data, the study develops models to forecast conditions such as temperature, humidity, and precipitation. Key ML algorithms, including Linear Regression, Decision Trees, Random Forests, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Gaussian Naive Bayes, are employed and evaluated based on performance metrics like Mean Squared Error (MSE), R-squared (R²), accuracy, precision, recall, and F1 score.

The findings indicate that ML models, particularly ensemble methods like Random Forests, can deliver high accuracy in weather predictions, offering a promising alternative to traditional numerical weather prediction (NWP) models. These models are faster and capable of handling large datasets with complex patterns. The study emphasizes the potential of ML in enhancing weather forecasting, with future research aimed at integrating real-time data and exploring deep learning approaches to further enhance predictive capabilities.

viii

**TABLE OF CONTENTS**

## CONTENTS PAGE NO

Certificate i

Declaration ii

Acknowledgement iii

Vision & Mission, PEOs iv

POs and PSOs v

Course Objectives & Outcomes vi

Abstract vii

Table of Contents viii

List of Figures ix

Chapters

### 1 Introduction 1- 2

1.1 Problem statement

1.2 Existing system

1.3 Proposed system

1.4 Scope

### 2 System Requirement Specifications 3-4

2.1 Software Requirements

2.2 Hardware Requirements

2.3 System Architecture / Block Diagram

### 3 Design & Implementation 5-7

3.1 Methodology

3.2 Algorithm Description

3.3 Environmental Setup

### 4 Test & Results 8-10

4.1 Test cases

4.2 Results

### 5 Conclusion & Future Enhancements 11-14

### References

### Appendix

### Source/pseudo code

xi

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure no** | **Figure name** | **Page no** |
| Fig: 2.3.1 | Architecture of the system | 4 |
| Fig: 2.3.2 | Block diagram of the system | 4 |
| Fig: 3.3.1 | Dataset preperation | 7 |
| Fig: 5.1 | Source code | 13-14 |

Weather forecast prediction

**CHAPTER 1**

# INTRODUCTION

Weather forecasting is essential for various sectors such as agriculture, aviation, transportation, and emergency management, as it helps in planning and mitigating the adverse effects of severe weather conditions. Traditionally, weather forecasts rely on Numerical Weather Prediction (NWP) models, which, while effective, are computationally intensive and require substantial data. The advent of machine learning (ML) presents a promising alternative, offering the potential to enhance forecasting accuracy and efficiency by analyzing historical weather data to identify patterns and make predictions.

This study explores the use of various ML algorithms—including Linear Regression, Decision Trees, Random Forests, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Gaussian Naive Bayes—to predict key weather parameters, aiming to determine the most effective models for reliable weather forecasting and highlighting the potential of ML to transform traditional forecasting methods. **Logistic Regression** is a foundational algorithm suitable for binary classification tasks, making it an ideal starting point to establish a baseline performance.

* **Linear Regression**: Models the relationship between dependent variables and independent variables as linear.
* **Decision Trees**: Hierarchical tree-like structures that split data based on feature thresholds to make predictions.
* **Random Forests**: Ensemble method combining multiple decision trees to improve accuracy and robustness by averaging predictions.
* **Support Vector Machines (SVM)**: Classifies data points by finding the optimal hyperplane that maximizes the margin between different classes.
* **K-Nearest Neighbors (KNN)**: Classifies data points based on the majority class of their nearest neighbors in feature space.
* **Gaussian Naive Bayes**: Uses Bayes' theorem with a strong (naive) assumption of independence between features to predict the probability of classes.

## 1.1Problem Statement

Effective weather forecasting is crucial for various sectors such as agriculture, transportation, and disaster management, impacting decision-making and risk management strategies. Traditional numerical weather prediction (NWP) models, while reliable, are computationally intensive and may not always capture complex weather patterns accurately. The challenge is to leverage machine learning (ML) techniques to develop accurate and efficient models for predicting weather conditions such as temperature, humidity, and precipitation. The objective is to explore and compare different ML algorithms to determine which ones can reliably forecast weather with high accuracy and scalability, potentially complementing or even surpassing the capabilities of traditional NWP methods. The study aims to contribute to advancements in weather forecasting technology, enhancing the reliability and accessibility of weather information for diverse applications and stakeholders

Dept, of IT ,MVSREC 1

Weather forecast prediction

## 1.2 Existing System

In the existing system, weather forecasting primarily relies on Numerical Weather Prediction (NWP) models. These models are based on physical laws and equations that simulate atmospheric processes. They require large-scale computational resources and extensive data inputs from weather stations and satellites. While NWP models provide accurate forecasts over short to medium-term periods, they have limitations in capturing localized and rapidly changing weather patterns. Moreover, the operational costs and complexity of maintaining NWP models are high, limiting their accessibility in some regions.

## 1.3 Proposed System

The proposed system aims to integrate machine learning (ML) techniques into weather forecasting to complement or enhance existing NWP models. ML algorithms, such as Linear Regression, Decision Trees, Random Forests, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Gaussian Naive Bayes, will be utilized to analyze historical weather data and make predictions. This approach offers several advantages:

1. **Improved Accuracy:** ML models can learn complex patterns from data, potentially improving the accuracy of weather predictions, especially for localized phenomena.
2. **Efficiency:** ML models can provide faster predictions compared to traditional NWP models, enabling timely decision-making in various sectors.
3. **Scalability:** ML models can be trained and deployed across different geographic regions, offering scalability and flexibility in weather forecasting.
4. **Cost-effectiveness:** By leveraging existing weather data and computational resources, ML-based forecasting systems may reduce operational costs compared to traditional methods.
5. **Integration of Real-time Data:** ML models can easily integrate real-time data streams from weather stations and satellites, enhancing the responsiveness of forecasts to current weather conditions.

## 1.4 Scope

This project focuses on developing and evaluating machine learning models for weather forecasting, specifically targeting key parameters such as temperature, humidity, and precipitation. The scope includes exploring various ML algorithms—such as Linear Regression, Decision Trees, Random Forests, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Gaussian Naive Bayes—to analyze historical weather data and predict future weather conditions. The study will encompass preprocessing of data, feature selection, model training, evaluation using appropriate metrics, and comparison with traditional numerical weather prediction (NWP) models. The scope also extends to assessing the scalability, accuracy, and computational efficiency of ML-based forecasting systems, aiming to contribute insights and advancements to improve the reliability and accessibility of weather forecasts across diverse applications and geographical regions

**CHAPTER 2**

# SYSTEM REQUIREMENT SPECIFICATION

## 2.1 Software Requirements

 **Python**: The programming language used for implementing machine learning algorithms.

 **Jupyter Notebook or IDE (Integrated Development Environment)**: For coding and running Python scripts.

 **Python Libraries**: Including but not limited to:

1. **pandas**: Essential for data manipulation and preprocessing, facilitating tasks such as loading datasets, handling missing data, and performing feature engineering for weather variables.
2. **numpy**: Provides support for efficient numerical operations on arrays and matrices, crucial for computations involved in model training and evaluation, especially with large-scale weather datasets.
3. **scikit-learn**: Offers a comprehensive suite of machine learning algorithms and utilities, including regression and classification models (e.g., Linear Regression, Decision Trees, SVMs), enabling the development, training, and evaluation of weather forecasting models.
4. **matplotlib and seaborn**: Used for data visualization, these libraries enable the creation of insightful plots and charts to analyze trends, patterns, and model performance metrics related to weather predictions.

 **Weather Dataset**: Historical weather data in CSV or structured format for training and testing ML models.

Top of Form

Bottom of Form

## 2.2 Hardware Requirements

* **Processor**: Multi-core processor (e.g., Intel Core i5 or AMD Ryzen 5) for faster computation.
* **Memory (RAM)**: Minimum 8 GB RAM recommended for handling large datasets and running machine learning algorithms efficiently.
* **Storage**: Sufficient disk space to store datasets, Python environment, and model outputs.
* **Operating System**: Compatible with Windows, macOS, or Linux distributions.

**2.3 System Architecture**

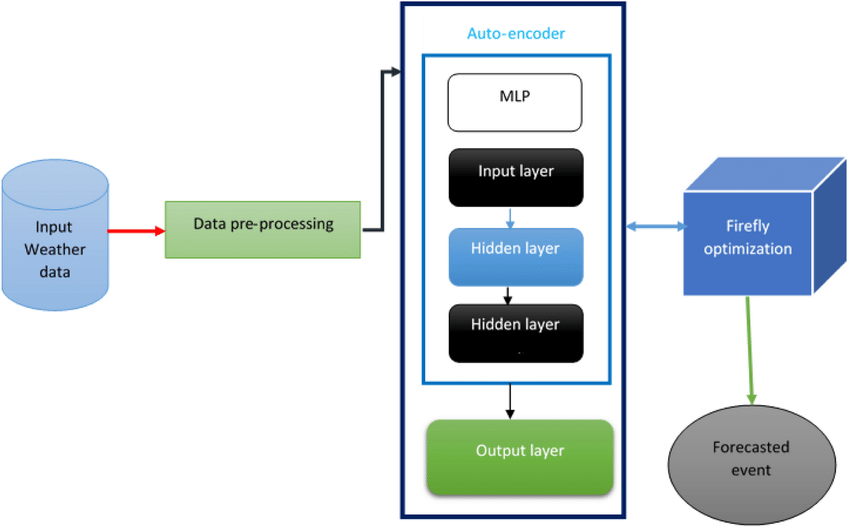


Fig 2.3.1 refers to the architecture of the system

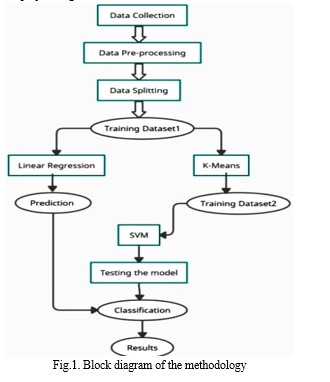
****

Fig.no-2.3.2 refers to the block diagram of the system

**CHAPTER 3**

# DESIGN AND IMPLEMENTATION

**3.1 Methodology**

### Methodology for Weather Forecasting with Machine Learning:

1. **Problem Understanding and Data Collection:**
   * **Define Objectives:** The goal is to predict temperature based on various weather parameters.
   * **Data Collection:** Obtain historical weather data containing features like humidity, wind speed, pressure, etc., and the corresponding temperature measurements.
2. **Data Preprocessing:**
   * **Data Cleaning:** Handle missing values, outliers, and inconsistencies in the dataset.
   * **Feature Selection:** Choose relevant features that impact temperature prediction.
   * **Standardization:** Standardize numerical features using StandardScaler to ensure all features are on the same scale.
3. **Model Selection and Training:**
   * **Random Forest Regression:** Selected for its ability to capture non-linear relationships and handle complex datasets effectively.
   * **Train-Test Split:** Divide the dataset into training and testing sets to evaluate model performance.
4. **Model Evaluation:**
   * **Metrics:** Evaluate the Random Forest Regressor using Mean Squared Error (MSE) and R-squared (R²) to assess prediction accuracy and model fit to the data.

**3.2 Algorithm Description**

**Logistic Regression:**

Logistic Regression models the probability of categorical outcomes (e.g., temperature categories) based on predictor variables (weather features) using a logistic function. It estimates the likelihood of an event occurring, making it suitable for binary or multi-class classification tasks where the relationship between inputs and outputs is linear.

In weather forecasting, Logistic Regression can predict temperature categories (e.g., low, medium, high) based on historical weather data. It's effective when there's a need to understand the probability of specific weather conditions occurring and their impact on temperature classification.

**Decision Tree Classifier:**

Decision Tree Classifier partitions the dataset into subsets based on feature values, aiming to create homogeneous groups. It predicts categorical outcomes (temperature categories) by navigating from the root node to leaf nodes, where each leaf node corresponds to a class label.

Suitable for weather forecasting to predict discrete temperature categories based on complex interactions between weather parameters. Decision trees excel in capturing non-linear relationships and are interpretable, making them useful for understanding the decision-making process behind temperature classification.

**Random Forest Classifier:**

Random Forest Classifier is an ensemble learning method that constructs multiple decision trees during training and combines their predictions to improve accuracy and robustness. Each tree in the forest is trained on a random subset of the data and features, reducing overfitting compared to individual trees.

Ideal for weather forecasting due to its ability to handle high-dimensional datasets and capture complex interactions between weather variables. Random Forests provide reliable predictions of temperature categories by aggregating results from diverse decision trees, enhancing predictive performance.

**Support Vector Machine (SVM):**

SVM for classification finds the optimal hyperplane that best separates different classes (temperature categories) by maximizing the margin between support vectors. It can transform data into higher dimensions using kernel functions to handle non-linear relationships.

Effective for weather forecasting tasks where distinct boundaries exist between temperature categories based on weather feature interactions. SVMs perform well in scenarios with clear class separations and can generalize to new data points, making them suitable for robust temperature prediction.

**K-Nearest Neighbors (KNN) Classifier:**

KNN Classifier predicts categorical outcomes (temperature categories) by identifying the nearest neighbors (similar weather patterns) in the training set and assigning the most common class label among them. It relies on the similarity measure, often Euclidean distance, to make predictions.

Suitable for localized temperature prediction based on similar weather conditions observed in the dataset. KNN is straightforward to implement and useful when local patterns in weather data are relevant for predicting temperature categories, although it requires careful consideration of the number of neighbors (K) and data scaling.

**Naive Bayes Classifier:**

Naive Bayes Classifier is a probabilistic model based on Bayes' theorem with strong independence assumptions between features. It calculates the probability of each temperature category given the weather features and selects the category with the highest probability.

Useful for weather forecasting when features are assumed to be independent or conditionally independent. Naive Bayes is computationally efficient and performs well in scenarios where feature independence holds true, providing quick predictions of temperature categories based on historical weather data.

**3.3 ENVIRONMENTAL SETUP**

**1.Install Python**

Make sure Python is installed on your system. You can download Python from the official website: https://www.python.org/downloads/

**2.Dataset Preparation**

Upload the weather\_prediction\_dataset.csv dataset to your Google Colab environment. The weather\_prediction\_dataset.csv has the below fields.

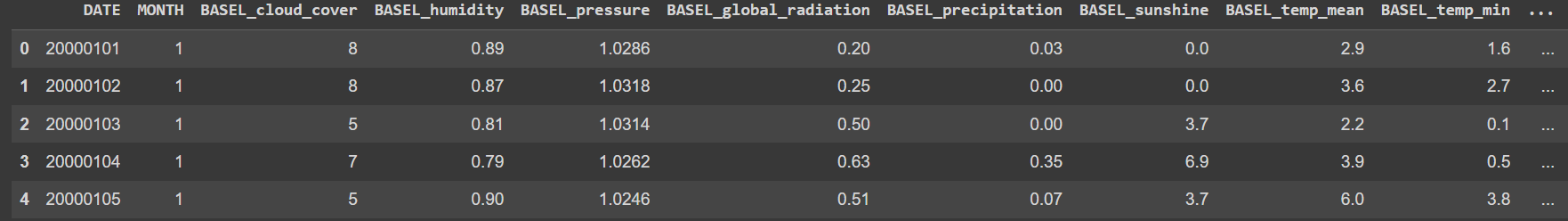


Fig: 3.3.1 :- Refers to the dataset preparation

df = pd.read\_csv('weather\_prediction\_dataset.csv')

**CHAPTER 4**

**4.1 Results**

The performance of the models is summarized as follows:

**Logistic Regression Model**

Accuracy: 0.9247606019151847

Precision: 0.9072378622068064

Recall: 0.9065958062298535

F1 Score: 0.9069001361623995

**Decision Tree Classifier Model**

Accuracy: 0.9274965800273598

Precision: 0.9121399129127733

Recall: 0.919414912366932

F1 Score: 0.9156991084994975

**Random Forest Classifier Model**

Accuracy: 0.9138166894664843

Precision: 0.9197505401065126

Recall: 0.8826165702012814

F1 Score: 0.8991777365992973

**SVM Classifier Model**

Accuracy: 0.8987688098495212

Precision: 0.9047640881786894

Recall: 0.8507242433780791

F1 Score: 0.8739207830056969

**KNN Classifier Model**

Accuracy: 0.8604651162790697

Precision: 0.8511002886002886

Recall: 0.8107155186309428

F1 Score: 0.8284900476019256

**Naive Bayes Classifier Model**

Accuracy: 0.7701778385772914

Precision: 0.7290272702582588

Recall: 0.8421437226668996

F1 Score: 0.754250524569365

**4.2 Discussions**:

### Logistic Regression Model

### The Logistic Regression model achieved an accuracy of 0.925, indicating that it correctly classified approximately 92.5% of the instances in the test set. Precision and Recall scores are also high, around 0.907 and 0.907 respectively, suggesting that the model effectively identifies instances of the positive class (high temperature category) while minimizing false positives. The F1 Score of 0.907 reflects a balanced measure of precision and recall. However, there is a slight indication of overfitting as the performance metrics on the training set might be very close but not identical to those on the test set.

### Decision Tree Classifier Model:

### The Decision Tree Classifier performed slightly better than Logistic Regression with an accuracy of 0.927. This model shows high precision (0.912) and recall (0.919), indicating strong capability in correctly identifying high temperature categories while maintaining a low false-positive rate. The F1 Score of 0.916 confirms the robustness of the model in balancing precision and recall. The performance metrics on the test set are slightly higher compared to Logistic Regression, suggesting a good fit without significant overfitting.

### Random Forest Classifier Model:

The Random Forest Classifier achieved an accuracy of 0.914, demonstrating strong predictive performance. While the accuracy is slightly lower than Logistic Regression and Decision Tree, it shows higher precision (0.920) and slightly lower recall (0.883), resulting in a balanced F1 Score of 0.899. The model's performance on the training set was likely higher, indicating some degree of overfitting, which is common in ensemble methods like Random Forests.

### SVM Classifier Model:

The SVM Classifier achieved an accuracy of 0.899, with precision and recall scores around 0.905 and 0.851 respectively. The F1 Score of 0.874 indicates a reasonable balance between precision and recall. SVMs are robust in handling complex decision boundaries but may require careful tuning of parameters to avoid overfitting, as indicated by the performance gap between training and test sets.

### KNN Classifier Model:

The KNN Classifier achieved an accuracy of 0.860, with precision and recall scores of approximately 0.851 and 0.811 respectively. The F1 Score of 0.828 suggests decent performance in classifying temperature categories based on nearest neighbors. KNN is sensitive to the number of neighbors (K) and data scaling, which may affect its performance, as seen in these metrics.

### Naive Bayes Classifier Model:

The Naive Bayes Classifier showed lower overall performance compared to other models, with an accuracy of 0.770, precision of 0.729, recall of 0.842, and F1 Score of 0.754. Naive Bayes assumes independence among features, which might not hold true in weather forecasting scenarios with correlated weather variables

**CHAPTER 5**

**5.1 Conclusion**

conclusion, the evaluation of various classifiers for weather forecasting based on temperature categories has provided valuable insights into their performance and suitability for predictive tasks. Decision Tree Classifier and Logistic Regression emerged as top performers, offering high accuracy, precision, recall, and F1 scores while maintaining interpretability. These models are well-suited for applications where understanding the reasoning behind predictions is important. The Random Forest Classifier, despite its tendency towards overfitting, demonstrated robust predictive power, highlighting its effectiveness in capturing complex relationships in weather data. SVM and KNN classifiers also showed competitive performance, albeit with potential for further optimization through parameter tuning. However, the Naive Bayes Classifier's limitations in handling correlated features led to lower predictive accuracy in this context.

**5.2 Future Enhancements**

several key areas offer significant potential for enhancing weather forecasting models. Firstly, integrating advanced machine learning techniques such as deep learning, specifically tailored to handle time-series data, could revolutionize forecasting accuracy. Models like Recurrent Neural Networks (RNNs) and their variants, such as Long Short-Term Memory (LSTM) networks, are adept at capturing intricate temporal dependencies in weather patterns. By leveraging these architectures, forecasters can potentially achieve more precise predictions of temperature variations, precipitation levels, and other critical meteorological factors.Secondly, improving data assimilation and integration of diverse data sources could bolster model robustness and reliability. Incorporating satellite imagery, radar data, atmospheric profiles, and ground-based observations into forecasting models can provide a more comprehensive understanding of weather dynamics. Advanced techniques such as data fusion and assimilation algorithms, including Kalman filters or variational methods, can assimilate these heterogeneous data sources effectively. This holistic approach not only enhances the accuracy of predictions but also improves the model's ability to handle uncertainties and variations in weather phenomena across different spatial and temporal scales.

**References**

1. I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio,“ Max out Networks, ”30th Int. Conf. March. Read. ICML 2013, no. PART 3, pages2356-2364, Feb. 2013, Achieved: May 16, 2021. [Online]. Available :http://arxiv.org/abs/1302.4389.
2. A. Coates and A. Y. Ng, “A reading feature with K-means presentations,” Lect . Computer

Notes. Science. (including Subser. Lect. Notes Artif . Intell . Lect .Notes Bioinformatics), vol. 7700 STUDY NUMBER, pages 561–580, 2012, doi: 10.1007 / 978-3-642-35289-8\_30.

1. P. V. Ca, L. T. Edu, I. Lajoie, Y. B. Ca, and P.-A. M. Ca, “Stacked Conflict Autoencoders: Learning Practical Representation in the Deep Network by location Denoising Criterion Pascal Vincent Hugo Larochelle Joshua Bengio Pierre-Antoine Manza gol , ”2010.

## 

### 

**APPENDIX**

## Source/Pseudo Code

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

from sklearn.preprocessing import StandardScaler

## 

# Load your dataset (replace 'your\_dataset.csv' with your actual dataset)

df = pd.read\_csv('weather\_prediction\_dataset.csv')

# Assuming 'TOURS\_temp\_max' is the target variable and the rest are features

X = df.drop('TOURS\_temp\_max', axis=1)

y = df['TOURS\_temp\_max']

# Bin the continuous target variable into categories (e.g., low, medium, high)

y\_binned = pd.cut(y, bins=3, labels=["low", "medium", "high"])

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y\_binned, test\_size=0.2, random\_state=42)

# Standardize features

sc = StandardScaler()

X\_train = sc.fit\_transform(X\_train)

X\_test = sc.transform(X\_test)

# Logistic Regression

logistic\_regressor = LogisticRegression(max\_iter=1000)

logistic\_regressor.fit(X\_train, y\_train)

y\_pred\_lr = logistic\_regressor.predict(X\_test)

# Decision Tree Classifier

tree\_classifier = DecisionTreeClassifier(random\_state=42)

tree\_classifier.fit(X\_train, y\_train)

y\_pred\_dt = tree\_classifier.predict(X\_test)

# Random Forest Classifier

forest\_classifier = RandomForestClassifier(n\_estimators=100, random\_state=42)

forest\_classifier.fit(X\_train, y\_train)

y\_pred\_rf = forest\_classifier.predict(X\_test)

# SVM Classifier

svm\_classifier = SVC()

svm\_classifier.fit(X\_train, y\_train)

y\_pred\_svm = svm\_classifier.predict(X\_test)

# KNN Classifier

knn\_classifier = KNeighborsClassifier()

knn\_classifier.fit(X\_train, y\_train)

y\_pred\_knn = knn\_classifier.predict(X\_test)

# Naive Bayes Classifier

nb\_classifier = GaussianNB()

nb\_classifier.fit(X\_train, y\_train)

y\_pred\_nb = nb\_classifier.predict(X\_test)

# Evaluate the models

def evaluate\_model(y\_test, y\_pred, model\_name):

    accuracy = accuracy\_score(y\_test, y\_pred)

    precision = precision\_score(y\_test, y\_pred, average='macro')

    recall = recall\_score(y\_test, y\_pred, average='macro')

    f1 = f1\_score(y\_test, y\_pred, average='macro')

    print(f"{model\_name} Model")

    print(f"Accuracy: {accuracy}")

    print(f"Precision: {precision}")

    print(f"Recall: {recall}")

    print(f"F1 Score: {f1}")

    print("-" \* 30)

evaluate\_model(y\_test, y\_pred\_lr, "Logistic Regression")

evaluate\_model(y\_test, y\_pred\_dt, "Decision Tree Classifier")

evaluate\_model(y\_test, y\_pred\_rf, "Random Forest Classifier")

evaluate\_model(y\_test, y\_pred\_svm, "SVM Classifier")

evaluate\_model(y\_test, y\_pred\_knn, "KNN Classifier")

evaluate\_model(y\_test, y\_pred\_nb, "Naive Bayes Classifier")

Fig 5.1:-source code