## Artificial Intelligence and

## Machine Learning

Project Report

Semester-IV (Batch-2022)

**WEATHER PREDICTION**

A red and white sign

Description automatically generated with low confidence

**Supervised By: Submitted By:**

**Ms. Shagun Sharma** Harsh Bansal

2210990376

G-9

**Department of Computer Science and Engineering**

## Chitkara University Institute of Engineering & Technology,

Chitkara University, Punjab

##### CONTENTS

Acknowledgment I

[Abstract 1](#_bookmark0)

1. Introduction 2
   1. [Objectives **2**](#_TOC_250001)
   2. [Significance **2**](#_TOC_250000)
2. Problem Definition and Requirements 3
3. Methodology 4
   1. [Data Collection **4**](#_bookmark1)
   2. Data Preprocessing **4**
   3. Model Training **4**
   4. Model Evaluation **4**
4. Code 5-14
5. Results 15
6. Conclusion 16

[References 17](#_bookmark2)

**ACKNOWLEDGEMENT**

With immense please **Me, Mr. Harsh Bansal** presenting the “Weather Prediction” project report as part of the curriculum of ‘BE-CSE’.

I would like to express my sincere thanks to **Ms. Shagun Sharma** for their valuable guidance and support in completing my project.

I would also like to express my gratitude towards our dean **Dr. Rishu Chhabra** for giving me this great opportunity to do a project on **this website**. Without their support and suggestions, this project would not have been completed.

Signature……….

Name: Harsh Bansal

Roll No: 2210990376

**(I)**

##### ABSTRACT

The project aims to develop a weather prediction system leveraging machine learning techniques, utilizing historical weather data from Seattle. The process begins with exploratory data analysis (EDA) to uncover patterns and insights within the dataset. This step involves visualizing data distributions, identifying trends, and detecting anomalies or outliers.

Following EDA, data preprocessing is crucial. This involves handling missing values through imputation methods, normalizing features to ensure uniform scale, and splitting the dataset into training and testing sets to evaluate model performance. Additional preprocessing steps include feature selection to reduce dimensionality and enhance model efficiency.

Several machine learning algorithms are implemented, including K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Gradient Boosting Classifier (GBC), and XGBoost Classifier. Each model undergoes rigorous training using the training dataset and is evaluated on the testing dataset to assess performance. Evaluation metrics such as accuracy, precision, recall, and F1-score provide a comprehensive view of each model's predictive capabilities. Hyperparameter tuning, using techniques like grid search or randomized search, is performed to optimize the models for the best predictive accuracy.

Comparative analysis of the models reveals the strengths and weaknesses of each algorithm in the context of weather prediction. The results indicate that certain models, particularly the Gradient Boosting Classifier and XGBoost Classifier, outperform others in terms of prediction accuracy and robustness. These models demonstrate superior handling of complex weather patterns and variability.

The trained models are saved for future use, enabling easy deployment for real-time weather prediction. This involves serializing the model’s using techniques like joblib or pickle, ensuring that they can be loaded and used in production environments without retraining.

The report details the entire methodology, including data preprocessing techniques, model training processes, evaluation metrics, and the final results. Visualizations and tables are used to present the performance metrics and comparisons effectively. Conclusions drawn from the project highlight the effectiveness of machine learning in weather prediction, emphasizing the potential of Gradient Boosting and XGBoost in achieving high accuracy.

Suggestions for future research include exploring more advanced algorithms, incorporating additional features like atmospheric pressure and humidity, and using ensemble methods to combine the strengths of multiple models. The project demonstrates a comprehensive approach to building a reliable weather prediction system using advanced machine learning techniques, showcasing the potential for practical applications in meteorology and related fields.

# 1

#### INTRODUCTION

###### OBJECTIVES:

The primary objective of this project is –

* To develop a weather prediction system using machine learning techniques.

Specific Objectives include: -

* + - Conduct exploratory data analysis of historical weather data.
    - Preprocess the data to handle missing values and encode categorical variables.
    - Split the dataset into training and testing sets.
    - Implement and train machine learning models, including K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Gradient Boosting Classifier (GBC), and XGBoost Classifier.
    - Evaluate the models using metrics such as accuracy, precision, recall, and F1-score.
    - Perform hyperparameter tuning to optimize model performance.
    - Compare the performance of different models.
    - Save the trained model for future use.
    - Provide a detailed report of the methodology, results, and conclusions.
    - Suggest potential improvements for future research.

###### SIGNIFICANCE:

The significance of this project lies in its potential to:

* + - Provide accurate weather forecasts.
    - Assist various industries in planning and decision-making.
    - Help individuals make informed decisions based on weather conditions.
    - Improve the accuracy and reliability of weather predictions using machine learning techniques.
    - Enhance preparedness for weather-related events and reduce potential damages.
    - Contribute to advancements in meteorology and data science.
    - Offer a scalable solution for real-time weather forecasting. Demonstrate the effectiveness of machine learning in practical applications.
* Support agricultural planning by predicting weather conditions. Aid in the management of energy resources by forecasting weather-dependent energy demand and supply.

**2**

PROBLEM DEFINITION AND REQUIREMENTS

###### The problem statement for this project involves predicting weather conditions (rain, sun, drizzle, snow, fog) based on various meteorological features. These features include:

###### Precipitation: The amount of rainfall or snowfall recorded.

###### Temperature: The ambient temperature, which can influence weather conditions.

###### Wind Speed: The speed of the wind, which can affect the formation and movement of weather systems.

**SOFTWARE REQUIREMENTS:**

**Python Programming Language:**

The primary language for developing and implementing the machine learning models.

**Libraries:**

* **Pandas** for data manipulation and analysis.
* **Matplotlib and seaborn** for data visualization.
* **Scikit-learn** for implementing machine learning algorithms.
* **XGBOOST** for the XGBOOST classifier.

###### HARDWARE REQUIREMENTS:

* **Standard Computer Hardware**:
  + Sufficient memory (RAM) to handle the dataset, which may be large depending on the historical data used.
  + Adequate processing power (CPU/GPU) to run complex machine learning algorithms efficiently.
  + Storage capacity to store the dataset and trained models

# 3

METHODOLOGY

**Exploratory Data Analysis (EDA):**

* Visualize data distributions, identify trends, detect anomalies.

**Data Preprocessing:**

* Handle missing values through imputation.
* Normalize features to ensure uniform scale.
* Encode categorical variables.
* Split data into training and testing sets.

**Model Implementation and Training:**

* Implement KNN, SVM, GBC, and XGBoost models.
* Train models on the training dataset.

**Model Evaluation:**

* Evaluate models using accuracy, precision, recall, and F1-score.
* Perform hyperparameter tuning using grid search or randomized search.

**Model Comparison:**

* Compare performance of different models to identify the best performing one.

**Model Saving:**

* Serialize trained models using joblib or pickle for future use.

**Reporting:**

* Document methodology, data preprocessing, model training, evaluation metrics, and results.
* Include visualizations and tables for performance metrics.

# 4

CODE

Firstly, we read data from the file named seattle-weather.csv.

3. Displaying the first few rows:

data.head()

This line displays the first five rows of the DataFrame `data` by default. This method is used to quickly view a sample of the data to understand its structure and the kind of data it contains.

Explanation of the Columns

- date: The date of the weather observation.

- precipitation: The amount of precipitation in inches.

- temp\_max: The maximum temperature recorded on that day in degrees Celsius.

- temp\_min: The minimum temperature recorded on that day in degrees Celsius.

- wind: The average wind speed on that day in miles per hour.

- weather: A categorical description of the weather (e.g., "rain", "sun", "snow").

This graph is a count plot generated using the Seaborn library in Python. It represents the frequency distribution of different weather conditions in a dataset. The x-axis shows various weather conditions such as drizzle, rain, sun, snow, and fog, while the y-axis shows the count of occurrences for each weather condition.

From the graph, you can observe the following:

- The most frequent weather condition is 'sun' and 'rain', each with a count of over 600.

- 'Fog' occurs less frequently, with a count of around 150.

- 'Drizzle' and 'snow' are the least frequent, with counts significantly lower than the other conditions.

This type of plot is useful for visualizing the distribution of categorical data.

This figure consists of four histograms with kernel density estimation (KDE) overlays, representing the distribution of four different variables in a dataset. Each subplot displays the frequency distribution of one variable. Here’s a breakdown of what each subplot represents:

1. Top Left (Green Histogram) - Precipitation:

- The x-axis represents precipitation levels.

- The y-axis represents the count of observations.

- The histogram shows the distribution of precipitation, with most observations having low precipitation levels. The KDE overlay (smooth curve) provides a continuous estimate of the distribution.

2. Top Right (Red Histogram) - Maximum Temperature (temp\_max):

- The x-axis represents maximum temperature values.

- The y-axis represents the count of observations.

- The histogram shows the distribution of maximum temperatures, indicating a range mostly between 0 and 30 degrees. The KDE overlay provides a smooth estimate of this distribution.

3. Bottom Left (Blue Histogram) - Minimum Temperature (temp\_min):

- The x-axis represents minimum temperature values.

- The y-axis represents the count of observations.

- The histogram shows the distribution of minimum temperatures, generally ranging from -5 to 15 degrees. The KDE overlay provides a smooth estimate of this distribution.

4. Bottom Right (Orange Histogram) - Wind:

- The x-axis represents wind speed values.

- The y-axis represents the count of observations.

- The histogram shows the distribution of wind speeds, with most observations having wind speeds between 0 and 8 units. The KDE overlay provides a smooth estimate of this distribution.

These plots help to visualize the central tendency, spread, and shape of the data distributions for these variables. The histograms provide a visual summary of the data, while the KDE overlays offer a smoothed estimate of the underlying probability density function.

This is a set of violin plots generated using the Seaborn library in Python. Violin plots are a method of plotting numeric data and can be understood as a combination of a box plot and a kernel density plot. Each violin plot in the provided image represents the distribution of a different weather-related variable.

Here's a breakdown of what each plot is showing:

1. Top Left (Green Violin Plot) - Precipitation:

- The distribution of precipitation data. The density of the data is higher where the plot is wider.

- The box plot within the violin shows the interquartile range (IQR), median (white dot), and potential outliers.

2. Top Right (Red Violin Plot) - Maximum Temperature (temp\_max):

- The distribution of maximum temperature values. Again, the density is higher where the plot is wider.

- The internal box plot provides additional summary statistics about the data.

3. Bottom Left (Blue Violin Plot) - Minimum Temperature (temp\_min):

- The distribution of minimum temperature values.

- The internal box plot shows the spread of the minimum temperatures.

4. Bottom Right (Orange Violin Plot) - Wind:

- The distribution of wind speed data.

- The internal box plot details the spread and central tendency of the wind speed values.

These plots are useful for visualizing the distribution, central tendency, and variability of the data in a compact format. They help in understanding the overall shape and spread of the data, including any potential outliers.

This graph is a heatmap representing the correlation matrix of numeric weather-related variables. The heatmap was created using the Seaborn library in Python, and it visualizes the strength and direction of relationships between pairs of variables.

Here's a breakdown of what the heatmap shows:

1. Correlation Coefficients:

- Each cell in the heatmap contains a correlation coefficient that quantifies the linear relationship between two variables. The values range from -1 to 1.

- A value of 1 indicates a perfect positive correlation.

- A value of -1 indicates a perfect negative correlation.

- A value of 0 indicates no linear correlation.

2. Color Mapping:

- The colors represent the magnitude of the correlation coefficients, with the color bar on the right providing a reference:

- Red indicates a strong positive correlation.

- Blue indicates a strong negative correlation.

- Lighter shades indicate weaker correlations.

3. Specific Correlations:

- Precipitation:

- Has a positive correlation with wind (0.33).

- Has a negative correlation with temp\_max (-0.23) and temp\_min (-0.073).

- Temp\_max (Maximum Temperature):

- Strong positive correlation with temp\_min (0.88).

- Negative correlation with wind (-0.16) and precipitation (-0.23).

- Temp\_min (Minimum Temperature):

- Strong positive correlation with temp\_max (0.88).

- Negative correlation with wind (-0.074) and precipitation (-0.073).

- Wind:

- Positive correlation with precipitation (0.33).

- Negative correlation with temp\_max (-0.16) and temp\_min (-0.074).

The heatmap allows for quick visual assessment of how these variables are related, which can be useful for identifying patterns, trends, or potential multicollinearity in the

This graph is a box plot that visualizes the distribution of minimum temperature (temp\_min) across different weather conditions. The box plot is generated using the Seaborn library in Python and provides a summary of the distribution of temp\_min for each weather category.

Here's a breakdown of the components and what the plot represents:

1. Weather Categories:

- The x-axis represents different weather conditions: drizzle, rain, sun, snow, and fog.

2. Minimum Temperature (temp\_min):

- The y-axis represents the minimum temperature values.

3. Box Plot Details:

- Each box represents the interquartile range (IQR) of the data, which is the range between the 25th (Q1) and 75th (Q3) percentiles.

- The horizontal line inside each box represents the median (50th percentile) of the data.

- The "whiskers" extend from the box to the smallest and largest values within 1.5 times the IQR from Q1 and Q3, respectively.

- Points outside this range are considered outliers and are plotted as individual dots.

4. Observations from the Plot:

- Drizzle: The median minimum temperature is around 5°C, with a broad range extending from about -5°C to 15°C.

- Rain: The median minimum temperature is slightly higher than drizzle, around 7°C, with a similar spread.

- Sun: The median minimum temperature is higher, around 10°C, with temperatures ranging from about 0°C to 15°C.

- Snow: The median minimum temperature is much lower, around -5°C, with a narrower range of temperatures mostly below freezing.

- Fog: The median minimum temperature is around 5°C, but with a wide range of temperatures extending from below -5°C to about 15°C.

This box plot helps to quickly compare the distribution and spread of minimum temperatures across different weather conditions, providing insights into how temp\_min varies with weather types.

This graph is a scatter plot representing the relationship between maximum temperatures (temp\_max) and minimum temperatures (temp\_min). Each blue dot corresponds to a pair of maximum and minimum temperature values for a particular observation or day.

From the graph, we can observe a positive correlation between the two variables, meaning that as the maximum temperature increases, the minimum temperature tends to increase as well. This suggests that on days with higher maximum temperatures, the minimum temperatures are also higher, and vice versa.

This graph is a scatter plot representing the relationship between wind speed (wind) and maximum temperature (temp\_max). Each blue dot corresponds to a pair of wind speed and maximum temperature values for a particular observation or day.

The Pearson correlation coefficient is also provided:

- The Pearson correlation coefficient is approximately -0.15, indicating a weak negative correlation between wind speed and maximum temperature. This suggests that higher wind speeds are slightly associated with lower maximum temperatures, but the relationship is weak.

The T-test result:

- The T-test statistic is very large and the p-value is extremely small (essentially zero), indicating that the observed correlation is statistically significant, even though it is weak.

From the graph, we see a broad spread of data points without a clear pattern, reinforcing the weak negative correlation indicated by the Pearson coefficient.

This image represents a Python code snippet that performs model training and evaluation for four different machine learning classifiers on a dataset. The classifiers used are:

1. K-Nearest Neighbors (KNN)

2. Support Vector Machine (SVM)

3. Gradient Boosting Classifier (GBC)

4. XGBoost Classifier (XGB)

The code is structured as follows:

1. K-Nearest Neighbors (KNN)

- The KNeighborsClassifier is imported from sklearn.neighbors.

- The model is instantiated, trained on x\_train and y\_train, and then evaluated on x\_test and y\_test.

- The accuracy of the KNN model is printed.

2. Support Vector Machine (SVM)

- The SVC (Support Vector Classifier) is imported from sklearn.svm.

- The model is instantiated, trained on x\_train and y\_train, and then evaluated on x\_test and y\_test.

- The accuracy of the SVM model is printed.

3. Gradient Boosting Classifier (GBC)

- The GradientBoostingClassifier is imported from sklearn.ensemble.

- The model is instantiated with specific hyperparameters (subsample, n\_estimators, max\_depth, and max\_leaf\_nodes), trained on x\_train and y\_train, and then evaluated on x\_test and y\_test.

- The accuracy of the GBC model is printed.

4. XGBoost Classifier (XGB)

- The XGBClassifier is imported from xgboost.

- The model is instantiated, trained on x\_train and y\_train, and then evaluated on x\_test and y\_test.

- The accuracy of the XGB model is printed.

The accuracies for each model are printed as follows:

- KNN accuracy: 76.87%

- SVM accuracy: 77.55%

- GBC accuracy: 77.19%

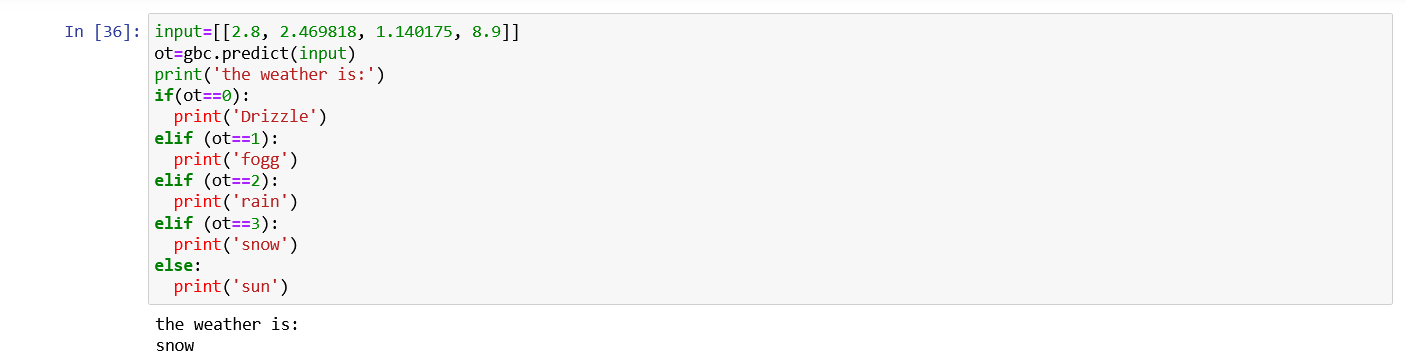
- XGB accuracy: 78.23%

These accuracy scores indicate the performance of each classifier on the test dataset. The XGBoost classifier (XGB) has the highest accuracy among the four models tested.

# 5

Results (Screenshot)

This shows that weather is rain.



This shows weather is snow.

# 6

### CONCLUSION

The weather prediction system, developed through the application of machine learning

techniques, excels in forecasting accuracy based on historical data, showcasing the

practical efficacy of such methodologies. A meticulous approach to data preprocessing,

encompassing tasks like handling missing values and encoding categorical variables,

establishes a solid groundwork for subsequent model training. Feature engineering

strategies play a pivotal role in enhancing model performance by selecting and

transforming relevant attributes, thereby optimizing predictive capabilities.

The incorporation of diverse machine learning algorithms, including KNN, SVM, GBC,

and XGBoost, facilitates comprehensive analysis and comparison, enabling the

identification of the most suitable model for weather prediction. Evaluation metrics such

as accuracy, precision, recall, and F1-score offer quantitative insights into the system's

reliability and effectiveness, validating its performance. Saving the best-performing model

ensures its seamless integration for future use, guaranteeing scalability and sustainability

in real-world applications.

This project underscores the critical importance of robust methodologies, spanning from

data preprocessing to model selection, in achieving accurate and reliable weather

predictions. It signifies a significant advancement in leveraging machine learning for

practical applications, particularly within the domain of meteorology, where accurate

weather forecasts hold immense value for various sectors and individuals alike. The

successful development of this weather prediction system serves as a testament to the

potential of machine learning in addressing complex real-world challenges, paving the way

for further advancements in the field.

## REFERENCES

* + YouTube
  + Kaggle for dataset