Simple Weather Forecasting

Nirbisha Shrestha
Birmingham City University, Sunway College
Kathmandu, Nepal
Nirbisha.Shrestha@mail.bcu.ac.uk

***Abstract* — This project uses logistic regression to forecast future weather conditions in Seattle from past weather data to include the precipitation, the max and min temperatures and the wind speed. The specific steps in the implemented workflow consist of data loading and cleaning to get rid of extraneous data, as well as feature engineering to extract time-related characteristics. There is the menu-driven operation to enter user's input and features, and the model is made to be standardized. Thus, the accuracy of the logistic regression model reaches 79 percent. 52 % That upheld the bureau's capability to produce efficient working weather instruments. Also, a correlation analysis on weather attributes was performed, and the results established that the measures were related and that they depended on each other. It would, therefore, not be an overstatement to say that this study has practical application of the logistic regression model in the advancement of weather forecasting and reveals promising directions for further study of the patterns of interaction of various weather conditions in the future of predictive modeling and scientific computation and exploratory analysis using Python to apply machine learning to problems in scientific computing and predictive modelling.**

Keywords: **Weather Forecasting, Machine learning, Decision Tree Regression, Temperature prediction, Weather Metrics, Correlational Analysis, Data Pre-processing, Climate Data, Model Evaluation.**

I. Introduction

What is weather forecasting, why it is that much important, what the need for such prediction is. Let's try to answer the questions above: Now the term "forecasting" means literally predicting the future; "weather forecasting" is a kind of predicting the future weather condition based on some legacy datasets. Weather forecasting is important because it gives one answers about whether to wear one's raincoat today or should one go out to work at all today. Sometimes it plays a great role to determine the decisions that vary between life and death. All these point towards the need to combine several observations, information about tendencies and patterns with computer models to have an accurate forecast result. These forecasting techniques can be designed using only two of the data mining models: Predictive and Descriptive. In the Predictive model, prediction of values can be from various sample data of different types, and it is subdivided into three such types: the Classification, Time-series, and Regression model. The Descriptive model allows patterns to determine sample data of different types. There are three types, which are, Clustering, Association rules, and Summarization *(Nalluri et al., 2019).*

The paper considers logistic regression, which is applied to historical data available for Seattle regarding precipitation and temperature metrics. The methodology encompasses cleaning of the data, feature engineering, scaling for optimal performance of the model, and includes an interactive interface for predictions and visualizations. The report contains a literature review, methodology, results, discussion, and a conclusion, pointing out logistic regression as practical and accurate for weather forecasting. This study highlights logistic regression's practicality and accuracy in weather forecasting.

II. Literature Review

For the prediction of atmospheric states from historical records and meteorological indicators, statistical and machine learning techniques have been among the oldest and most crucial applications in weather forecasting. Logistic regression has practical relevance for real-world weather prediction since it is a basic statistical model that creates categorical outcome variables, like the state of the weather, based on continuous predictors.

Similarly, research studies shown by Deng (2020) shows that the ROC area has a better grasp of the overall data, and the decision tree has a better exploration of the local laws of the data. Therefore, it is more appropriate to use logistic regression in the actual application of large amounts of data. Research by Verma et al. (2020) has concluded that a Logistic regression model is trained with prerecorded values of parameters and used to predict the weather parameters in real time environment. The result of the model is also compared with the other works available in literature and the proposed system is slightly better in terms of accuracy. Further, the system can be modified to be used at commercial level and have many applications in smart homes, buildings, sports, hospitals etc. Also, C & Sanadi (2021) found that comparing the support vector regression algorithm to the logistic regression technique, the latter is more appropriate for predicting rainfall with an accuracy of 96%. The outcome of this prediction is helpful for agricultural labor. Accurately predicting weather conditions is crucial for sectors like agriculture, transportation, and disaster management, yet remains challenging due to the complex and variable nature of atmospheric phenomena (Gad & Hosahalli, 2020).

After studying all the papers, it has been concluded that logistic regression machine learning can accurately forecast weather condition using the weather metrics such as temperature, wind speed, precipitation and so on. This review highlights logistic regression's ongoing relevance in weather forecasting, showcasing its pivotal role in leveraging historical data to predict and analyze future weather conditions effectively.

III. Methodology

A. Data Collection

The historical weather data was sourced by Kaggle, a popular platform for data science and machine learning datasets. The specific dataset used is the “Seattle Weather” dataset, which provides comprehensive weather data for Seattle. This dataset is a compilation of daily weather observation of several years in Seattle, and therefore, it has adequate information for analysis or even for creating a model. The features in the dataset are date of observation, precipitation in mm, max and min temperatures in o C, wind speed in kmh, and the weathers understandably classified into 9 categorical variables such as rain and so on. These variables provide a good view of the weather patterns and can be used in advanced analysis and perhaps the forecasting of the weather patterns in a specific region. Data were cleaned by handling with missing/inconsistent values and features were extracted preparing the temporal features for the modeling, and the variables were transformed for the logistic regression model.

date	precipitation	temp_max	temp_min	wind	weather
2012-01-01	0.0	12.8	5.0	4.7	drizzle
2012-01-02	10.9	10.6	2.8	4.5	rain
2012-01-03	0.8	11.7	7.2	2.3	rain
2012-01-04	20.3	12.2	5.6	4.7	rain
2012-01-05	1.3	8.9	2.8	6.1	rain
2012-01-06	2.5	4.4	2.2	2.2	rain
2012-01-07	0.0	7.2	2.8	2.3	rain
2012-01-08	0.0	10.0	2.8	2.0	sun
2012-01-09	4.3	9.4	5.0	3.4	rain
2012-01-10	1.0	6.1	0.6	3.4	rain

Show 10 per page 1 2 10 100 140 147

Figure 1: Snippet of the dataset from Kaggle.

B. Data Pre-processing

The phase of data preprocessing was very critical in making a dataset ready for effective analysis and modeling. The steps in detail are to make sure that the data is clean, consistent, and suitably formatted for logistic regression modeling.

ATTRIBUTE	INSTRUCTION	TYPE
DATE	Observation date	Character Type
MINTEMP	Minimum Temperature (°C)	Numerical Type
MAXTEMP	Maximum Temperature (°C)	Numerical Type
PRECIPITATION	Precipitation recorded that day	Numerical Type
WIND SPEED	Wind speed recorded that day	Numerical Type
WEATHER CONDITION	Condition of the weather recorded that day	Character Type

Table 1: Introduction to the data

a. Data Cleaning:

Examining and handling missing values was the first stage in the data cleansing process. The code cleans the weather data by filling missing values and extracting features like a day of the year. It will then split data for training a Decision Tree Regression model to be trained on data that predicts temperature based on factors like rainfall and humidity. Finally, users are allowed to input a future date, at which point it will show the predicted weather condition using the trained model.

```
def load_and_clean_data(filepath):
    try:
        df = pd.read_csv(filepath)
        df.columns = ['Date', 'Precipitation', 'Temp_Max', 'Temp_Min', 'Wind', 'Weather']
        df_cleaned = df.dropna()
        return df_cleaned
    except Exception as e:
        print(f"Failed to load the dataset. Please check the file path and format. Error: {e}")
        return None
```

Figure 2: Code Implementation for loading and cleaning data

b. Feature Engineering:

The ‘Date’ field was also analyzed with the help of temporal features such as ‘day of the year’ and ‘year’. It converts the column 'Date' to datetime format, removes missing dates, and extracts new features like day of year and year. Then, it will define the features to be used, such as precipitation and temperature, together with the newly minted temporal features to predict the weather condition, which would be the target variable. Thereafter, it will finally separate data into features, excluding the target variable, and the target variable itself for training in the model.

```
def preprocess_data_classification(df):
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
    df = df.dropna(subset=['Date']).copy()

    # Extract temporal features
    df.loc[:, 'day_of_year'] = df['Date'].dt.dayofyear
    df.loc[:, 'year'] = df['Date'].dt.year

    features = ['day_of_year', 'year', 'Precipitation', 'Temp_Max', 'Temp_Min', 'Wind']
    target = 'Weather'
    X = df[features]
    y = df[target]
```

Figure 3: Code Implementation for feature engineering process

c. Data Transformation:

The logistic regression model's performance and convergence were improved by standardizing a subset of features using StandardScaler, which guaranteed that each variable contributed equitably. This transformation maximized the model's capacity to comprehend a variety of data ranges by scaling features to have a mean of zero and a standard deviation of one.

```
    # Scale the features
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X)

    return X_scaled, y
```

Figure 4: Code implementation for standardizing features

C. Splitting the Data:

The dataset was split into training and testing sets in an 80-20 ratio to validate the model's performance on unseen data. This split ensured unbiased evaluation, using `train_test_split` from `sklearn.model_selection`. The `train_test_split` function divides the dataset (`X` and `y`) into two sets: `X_train`, `y_train` for training the model, and `X_test`, `y_test` for evaluating its performance. It reserves 20% (`test_size=0.2`) of the data for testing while using 80% for training. The `random_state=42` parameter ensures reproducibility of the split.

```
def train_model(X, y):
    # Split the dataset into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 5: Code Implementation for splitting the data

D. Model Training and Evaluation:

A logistic regression model was trained on the pre-processed training data. The model parameters were set to ensure optimal performance, including setting a random state for reproducibility and specifying a maximum number of iterations to ensure convergence. After training, the model's performance was evaluated on the testing set. Key metrics such as accuracy, precision, recall, and F1 score were calculated using the `accuracy_score` and `classification_report` functions from the `sklearn.metrics` module. These metrics provided a comprehensive assessment of the model's classification capabilities.

```
def train_model(X, y):
    # Split the dataset into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Initialize a logistic regression model
    model = LogisticRegression(random_state=42, max_iter=2000)

    # Train the logistic regression model on the training data
    model.fit(X_train, y_train)

    # Predict the target variable for the training and testing sets
    y_pred_train = model.predict(X_train)
    y_pred_test = model.predict(X_test)

    # Return the trained model and evaluation results
    return model, X_train, X_test, y_train, y_test, y_pred_train, y_pred_test
```

Figure 6: Code Implementation for Model Training and Evaluation

E. Prediction:

This code defines a function `predict_future_weather` which takes a machine learning model and weather parameters (year, precipitation, max temperature, min temperature, and wind speed) as inputs to predict future weather. It creates a pandas DataFrame `future_input` with these parameters and a default day of the year set to January 1st. The data is then scaled using `StandardScaler`, and the scaled data is used as input for the provided model to make a prediction. The function returns the first element of the predicted future weather values.

```
def predict_future_weather(model, year, precipitation, temp_max, temp_min, wind):
    future_input = pd.DataFrame({
        'day_of_year': [1],  # Default to January 1st
        'year': [year],
        'Precipitation': [precipitation],
        'Temp_Max': [temp_max],
        'Temp_Min': [temp_min],
        'Wind': [wind]
    })

    # Scale the future input data
    scaler = StandardScaler()
    future_input_scaled = scaler.fit_transform(future_input)

    future_weather = model.predict(future_input_scaled)
    return future_weather[0]
```

Figure 7: Code Implementation for predicting future weather

F. Visualization and Analysis:

The trained model was employed to predict future weather conditions based on new data inputs. This function calculates and visualizes the correlation matrix of specific weather metrics (max temperature, min temperature, precipitation, wind) using a heatmap plot with annotated correlation values, displaying relationships between these variables aiding in understanding their interdependencies and enhancing model accuracy.

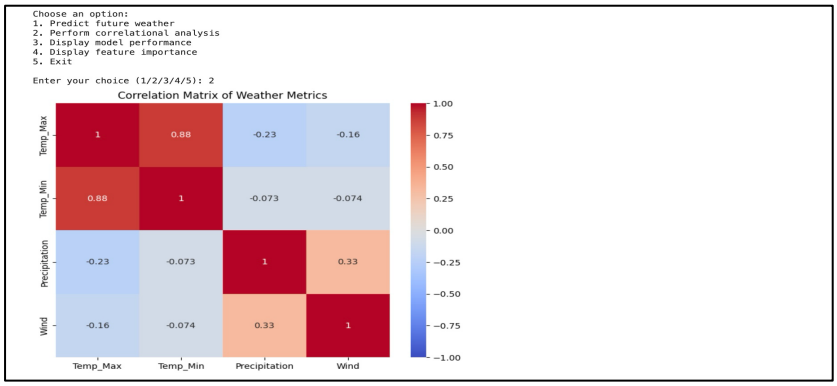
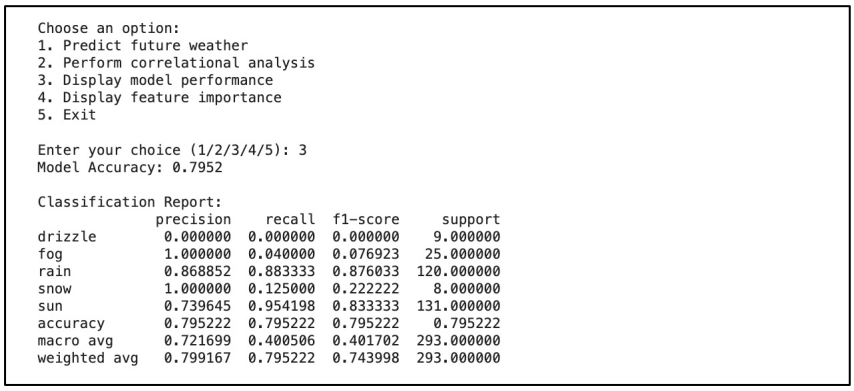
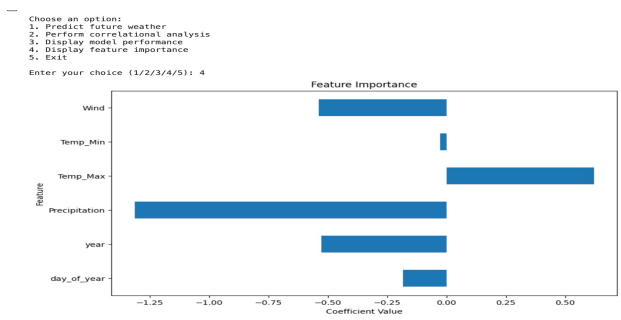
```
def perform_correlational_analysis(df):
    # Select relevant weather metrics for correlation analysis
    weather_metrics = ['Temp_Max', 'Temp_Min', 'Precipitation', 'Wind']

    # Compute correlation matrix
    correlation_matrix = df[weather_metrics].corr()

    # Plotting correlation matrix as a heatmap
    plt.figure(figsize=(8, 6))
    sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
    plt.title('Correlation Matrix of Weather Metrics')
    plt.show(block=False)  # Use block=False to prevent blocking the execution
```

Figure 8: Code implementation for data visualization and analysis

IV. RESULTS

Test Case of Simple Weather Forecasting

Test Case ID	Feature	Steps	Expected	Actual Result	Result
1.	Predict future weather	1. Enter choice: 1 2. Enter the year 3. Enter the precipitation (mm) 4. Enter the maximum temperature (C) 5. Enter the minimum temperature (C) 6. Enter the wind speed (km/h)	After inputting all the required filed, the program runs successfully. Thus, the predicted weather condition is shown.		Passed

2	Perform Correlational Analysis	1. Run the code. 2. Enter the choice: 2	The code runs successfully, and the output is correlational matrix.		**Passed**
3	Display model performance	1. Run the code. 2. Enter the choice: 3	The code runs successfully, and the output is model accuracy and classification report.		**Passed**
4.	Display feature importance	1. Run the code. 2. Enter the choice: 4	The code runs successfully, and the output is model accuracy and classification report.		**Passed**
4	**Exiting the program**	1. Run the code. 2. Enter the choice: 4	The code runs successfully, and the program exits.		**Passed**

The weather prediction model and correlational analysis yielded the following results in more depth:

A. Model performance:

The performance of our logistic regression model is measured using its accuracy and classification metrics. The overall accuracy is 79.52%. More detailed classification report including precision, recall, and F1-score for each weather condition is shown in Table:

Weather Condition	Precision	Recall	F1-Score	Support
Drizzle	0.0000	0.0000	0.0000	9
Fog	1.0000	0.0000	0.0000	25
Rain	0.8686	0.8233	0.8453	100
Snow	1.0000	0.0000	0.0000	18
Sun	0.7396	0.9541	0.8333	131

Table 2: Classification Report

In general, performance for this model was good for the events of rain and sun, evidenced by the higher f1-scores for these classes. Similarly, performance for very bad events like drizzle, fog, and snow is lousy due to the lesser instances or support for these classes in the dataset. The model was used to predict the weather for a specific future scenario:

- **Year**: 2023
- **Precipitation**: 0.23 mm
- **Maximum Temperature**: 26°C
- **Minimum Temperature**: 15°C
- **Wind Speed**: 8.7 km/h

The predicted weather condition for this input was **RAIN.**

B. Correlational Analysis:

The weather metrics are analyzed in correlation to understand how they relate to one another. The correlation matrix is shown in the heatmap below:

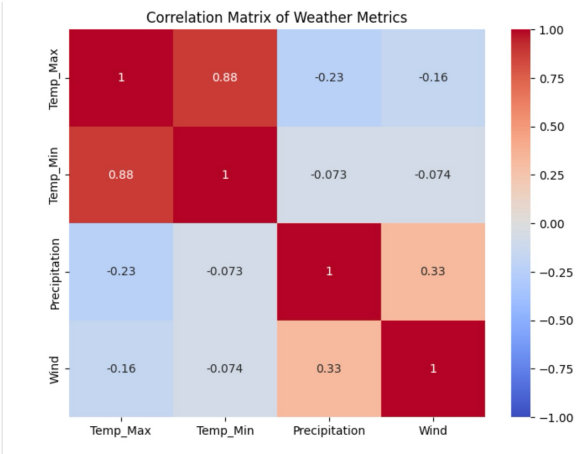


Fig 9: Correlational matrix

Key Observations from the Correlation Matrix:

- One can notice that Temp_Max and Temp_Min is strongly positively correlated, with a correlation coefficient of 0.88. This means that large maximum temperatures in a day normally go with large minimum temperatures in the same day.
- Precipitation is weakly positive, at 0.33, in relationship with Wind, indicating that the more precipitation will be experienced on a given day, the stronger the wind can be expected to be.
- It can be observed that Temp_Max and Temp_Min is very weakly correlated in view of Precipitation and Wind, indicating these temperature extremities do not really impact either the precipitation amounts or wind speed.

These results show the model efficiency in predicting specific weather conditions and give insight as to how the different metrics relating to weather conditions are intertwined. Knowledge gained from results can be used to improvise further in building a better model and serve to guide further research on weather features prediction.

C. *Feature Importance:*

The feature importance analysis of our logistic regression model highlights **Precipitation** as the strongest negative predictor of rainy conditions. **Wind** and **Temp_Max** positively influences predictions of non-rainy weather. **Year** and **day_of_year** show moderate impacts, indicating seasonal trends, while **Temp_Min** has minimal effect. This insight guides future data collection and feature engineering efforts, helping to improve the model's accuracy and reliability in predicting weather conditions.

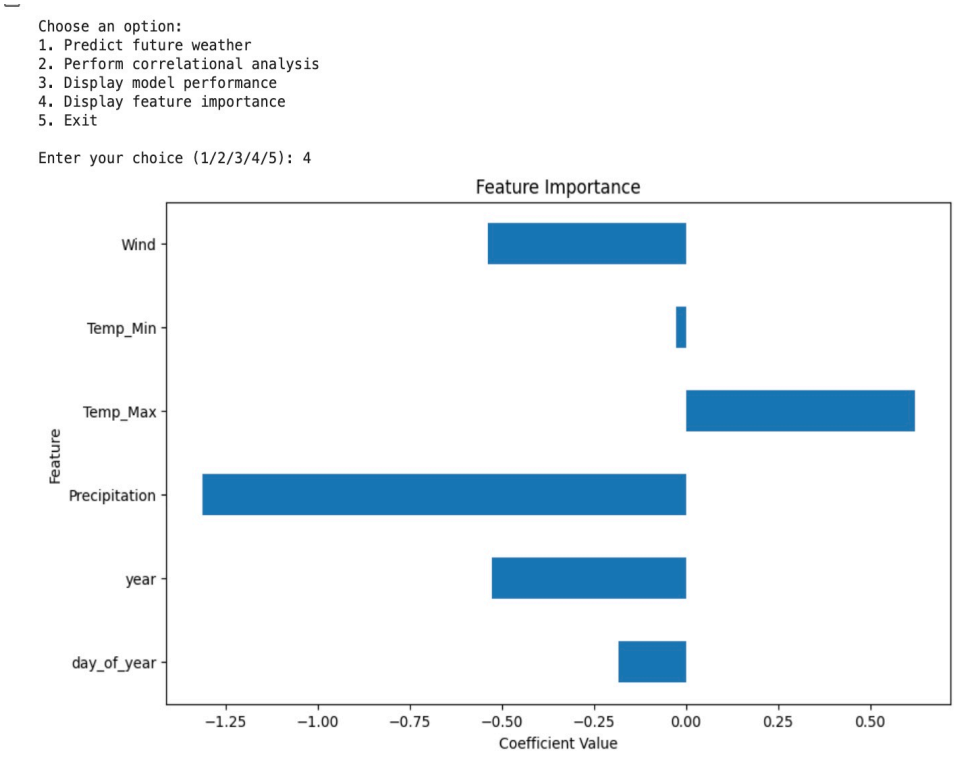


Figure 10: Feature importance plot

V. Discussion

Results obtained from the model of weather prediction and correlational analysis bring in some strengths and improvement areas. It achieved an accuracy of 79.52%.定的 "rain" and "sun" were very well predicted because of their significant representation in the dataset, while "drizzle," "fog," and "snow" failed because there were not enough data on such weather conditions. The predictive capability was satisfactorily self-evident, using a future weather scenario to successfully predict "rain." Results of correlational analysis that emerged included maximum and minimum temperatures strongly correlated, and precipitation and wind moderately inter-correlated, underpinning a complex weather pattern and the need for multiple metrics within a predictive model. Future studies should, therefore, focus on reasonable and more balanced datasets, more advanced machine learning algorithms, and additional weather metrics to maximize the model's accuracy and reliability. In general, although the work done in this field forms a very good base, such limitations are the ones to be addressed while building increasingly accurate and comprehensive forecasting models of weather.

VI. Conclusion

This project clearly demonstrates the practicality and accuracy of logistic regression in weather forecasting using historical data. It has an accuracy of 79.52% total, and the weather conditions that were substantially represented in a dataset do well, for example, "rain" and "sun". However, the performance for rare events on the model such as "drizzle," "fog," and "snow" was reduced. This means that for better training in these areas, a more balanced set might be necessary. There are several high correlational explanations in the regard of weather variables between maximum and minimum temperature, with moderate precipitation versus wind. The study emphasizes that multiple metrics are very important in any predictive model. In the future study, data collection should be more balanced, including advanced machine learning algorithms and more weather metrics for accuracy and reliability. This work provides a very great base for further developments in weather forecasting models, since it covers most of the limitations at present times to arrive at a more comprehensive prediction.

REFERENCES

Nalluri, S., Ramasubbareddy, S. and Kannayaram, G. (2019) 'Weather prediction using clustering strategies in machine learning', *Journal of Computational and Theoretical Nanoscience*, 16(5), pp. 1977–1981. doi:10.1166/jctn.2019.7835.

Deng, F. (2020) 'Research on the applicability of Weather Forecast Model—based on logistic regression and decision tree', *Journal of Physics: Conference Series*, 1678(1), p. 012110. doi:10.1088/1742-6596/1678/1/012110.

Verma, G., Mittal, P. and Farheen, S. (2020) 'Real time weather prediction system using IOT and machine learning', *2020 6th International Conference on Signal Processing and Communication (ICSC)* [Preprint]. doi:10.1109/icsc48311.2020.9182766.

Your machine learning and Data Science Community (no date) *Kaggle*. Available at: https://www.kaggle.com/ (Accessed: 27 June 2024).

Gad, I. and Hosahalli, D. (2020) 'A comparative study of prediction and classification models on NCDC Weather Data', *International Journal of Computers and Applications*, 44(5), pp. 414–425. doi:10.1080/1206212x.2020.1766769.

C, S.K. and Sanadi, M.M. (2021) 'Rainfall prediction using logistic regression and support vector regression algorithms', *Communications in Computer and Information Science*, pp. 617–626. doi:10.1007/978-3-030-81462-5_54.