Final Project Report Template

1. Introduction

1.1. Project Overview

The **Rainfall Prediction Using Machine Learning** project focuses on creating a predictive model to forecast future rainfall based on historical meteorological data. Accurate rainfall prediction is crucial for various sectors like agriculture, water resource management, disaster preparedness, and transportation. Traditional methods of predicting rainfall can sometimes lack precision due to the complexity of weather patterns. By utilizing machine learning algorithms, this project aims to enhance the accuracy of rainfall forecasts, providing timely information for decision-making.

1.2. Objectives

The key objectives of the **Rainfall Prediction Using Machine Learning** project are:

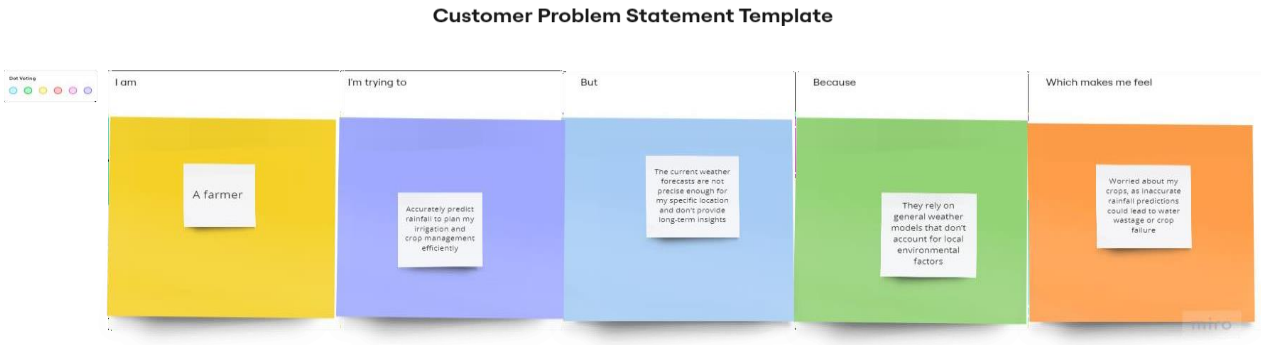
- To develop a machine learning model that predicts rainfall based on historical weather data, including temperature, humidity, wind speed, and other meteorological factors.
- To improve the accuracy of short-term and long-term rainfall forecasts using data-driven approaches.
- To assist industries like agriculture and water management in planning and preparation by providing reliable rainfall predictions.
- To reduce the impact of natural disasters like floods by improving early warning systems through accurate rainfall forecasting.
- To create a scalable and efficient solution that can be implemented in various regions, adapting to local weather conditions.

2. Project Initialization and Planning Phase

2.1 Define Problem Statement

Farmers and agricultural planners struggle with inaccurate and non-localized weather forecasts, leading to poor planning and potential crop loss. This causes anxiety and uncertainty about the best times to plant and water crops. Similarly, daily commuters and travellers face frustration and disruptions due to untimely and imprecise weather updates, impacting their travel plans and overall experience. Our project aims to address these issues by providing accurate and localized rainfall predictions, helping both groups make informed decisions and improve their productivity and convenience.

Example:



Customer Problem Statement Template

Dot Voting

I am

A farmer

I'm trying to

Accurately predict rainfall to plan my irrigation and crop management efficiently

But

The current weather forecasts are not precise enough for my specific location and don't provide long-term insights

Because

They rely on general weather models that don't account for local environmental factors

Which makes me feel

Worried about my crops, as inaccurate rainfall predictions could lead to water wastage or crop failure

Reference: https://miro.com/templates/customer-problem-statement/

Problem Statement (PS)	I am (Customer)	I’m trying to	But	Because	Which makes me feel
PS-1	A farmer	Accurately predict rainfall to plan my irrigation and crop management Efficiently.	The current weather forecasts are not precise enough for my specific location and don't provide long-term insights.	They rely on general weather models that don’t account for local environmental factors.	Worried about my crops, as inaccurate rainfall predictions could lead to water wastage or crop failure.

2.2 Project Proposal (Proposed Solution) Report

The proposal report aims to transform Rainfall Prediction using machine learning, boosting efficiency and accuracy. It tackles system inefficiencies, promising better operations, reduced risks, and happier customers. Key features include a machine learning-based credit model and real-time decision-making.

Project Overview	
Objective	The objective of this project is to develop a machine learning-based system that can accurately predict rainfall, enabling better decision-making in various industries such as agriculture, water resource management, and urban planning.
Scope	This project involves analyzing large datasets of historical weather records and utilizing machine learning algorithms to predict rainfall patterns. The goal is to provide timely and precise forecasts, improving the decision-making process in agriculture, urban planning, and disaster mitigation.
Problem Statement	
Description	Traditional methods of rainfall prediction rely heavily on statistical models that are often limited in their accuracy and adaptability. These methods struggle with complex, nonlinear patterns in weather data, leading to less precise forecasts, which can negatively impact agriculture, infrastructure planning, and disaster readiness.
Impact	Enhancing rainfall prediction accuracy will lead to better resource management in agriculture, improved urban planning, and more efficient disaster preparedness. Accurate rainfall predictions will help mitigate risks associated with flooding and drought, positively impacting local economies and public safety.
Proposed Solution	
Approach	The solution proposes using machine learning models, such as decision trees, random forests, and neural networks, to analyze historical weather data and predict rainfall. By training these models on large datasets, the system will be able to capture complex patterns and provide more reliable rainfall predictions.

Key Features	1. This solution harnesses advanced machine learning models for unparalleled rainfall prediction accuracy. 2. It dynamically updates with realtime data, ensuring continuous adaptability and precision. 3. By incorporating geographical and meteorological variables, it provides a comprehensive approach to understanding rainfall p atterns.

Resource Requirements

Resource Type	**Description**	**Specification/Allocation**
Hardware		
Computing Resources	CPU/GPU specifications, number of cores	T4 GPU
Memory	RAM specifications	8 GB
Storage	Disk space for data, models, and logs	1 TB SSD
Software		
Frameworks	Python frameworks	Flask
Libraries	Additional libraries	scikit-learn, pandas, numpy, matplotlib, seaborn
Development Environment	IDE, version control	Jupyter Notebook, vscode, Git
Data		
Data	Source, size, format	Kaggle dataset, 614, csv UCI dataset, 690csv, Meteorological departments, open weather datasets

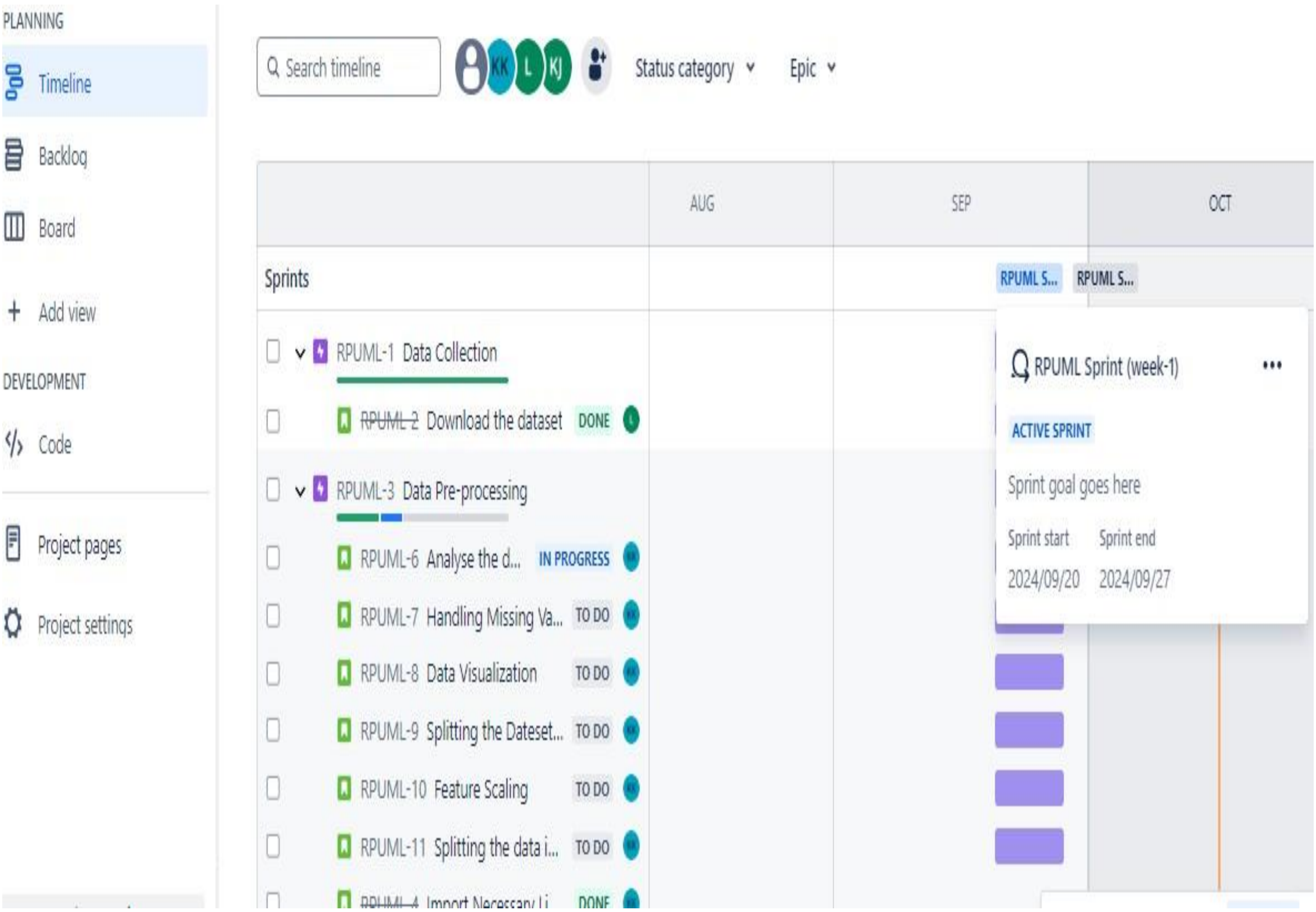
2.3 Initial Project Planning

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority	Team Members	Sprint Start Date	Sprint End Date (Planned)
Sprint -1	Data Collection	RPUML-2	Download the dataset	High	Susmitha	2024/09/20	2024/09/27
Sprint -3	Data Preprocessing	RPUML-6	Analyze the data	Medium	Anil kumar	2024/09/20	2024/09/27
Sprint -3	Data Preprocessing	RPUML-7	Handling missing values	Medium	Anil kumar	2024/09/20	2024/09/27
Sprint -3	Data Preprocessing	RPUML-8	Data visualization	Medium	Anil kumar	2024/09/20	2024/09/27
Sprint -3	Data Preprocessing	RPUML-9	Splitting the dataset	Medium	Siva Koteswara Reddy	2024/09/20	2024/09/27
Sprint -3	Data Preprocessing	RPUML-10	Feature scaling	Medium	Siva Koteswara Reddy	2024/09/20	2024/09/27
Sprint -3	Data Preprocessing	RPUML-11	Splitting the data into training/testing sets	Medium	Siva Koteswara Reddy	2024/09/20	2024/09/27
Sprint -12	Model Building	RPUML-13	Training and testing the model	High	Jaya Krishna	2024/09/27	2024/10/05
Sprint	Model Building		Model				

-12		RP-UML-14	evaluation	High	Jaya Krishna	2024/09/27	2024/10/05
Sprint -12	Model Building	RP-UML-15	Save the model	High	Jaya Krishna	2024/09/27	2024/10/05
Sprint -16	Project Initialization and Planning	RP-UML-17	Define the problem statement	High	Anil kumar	2024/09/20	2024/09/27
Sprint -16	Project Initialization and Planning	RP-UML-18	Propose a solution	Medium	Jaya Krishna	2024/09/20	2024/09/27
Sprint -16	Project Initialization and Planning	RP-UML-19	Write the planning report	High	Susmitha	2024/09/20	2024/09/27

Screenshot:



□ RPUML-1 Data Collection

□ ~~RPUML-2~~ Download the dataset DONE

□ RPUML-3 Data Pre-processing

□ RPUML-6 Analyse the d... IN PROGRESS

□ RPUML-7 Handling Missing Va... TO DO

□ RPUML-8 Data Visualization TO DO

□ RPUML-9 Splitting the Dataset... TO DO

□ RPUML-10 Feature Scaling TO DO

□ RPUML-11 Splitting the data i... TO DO

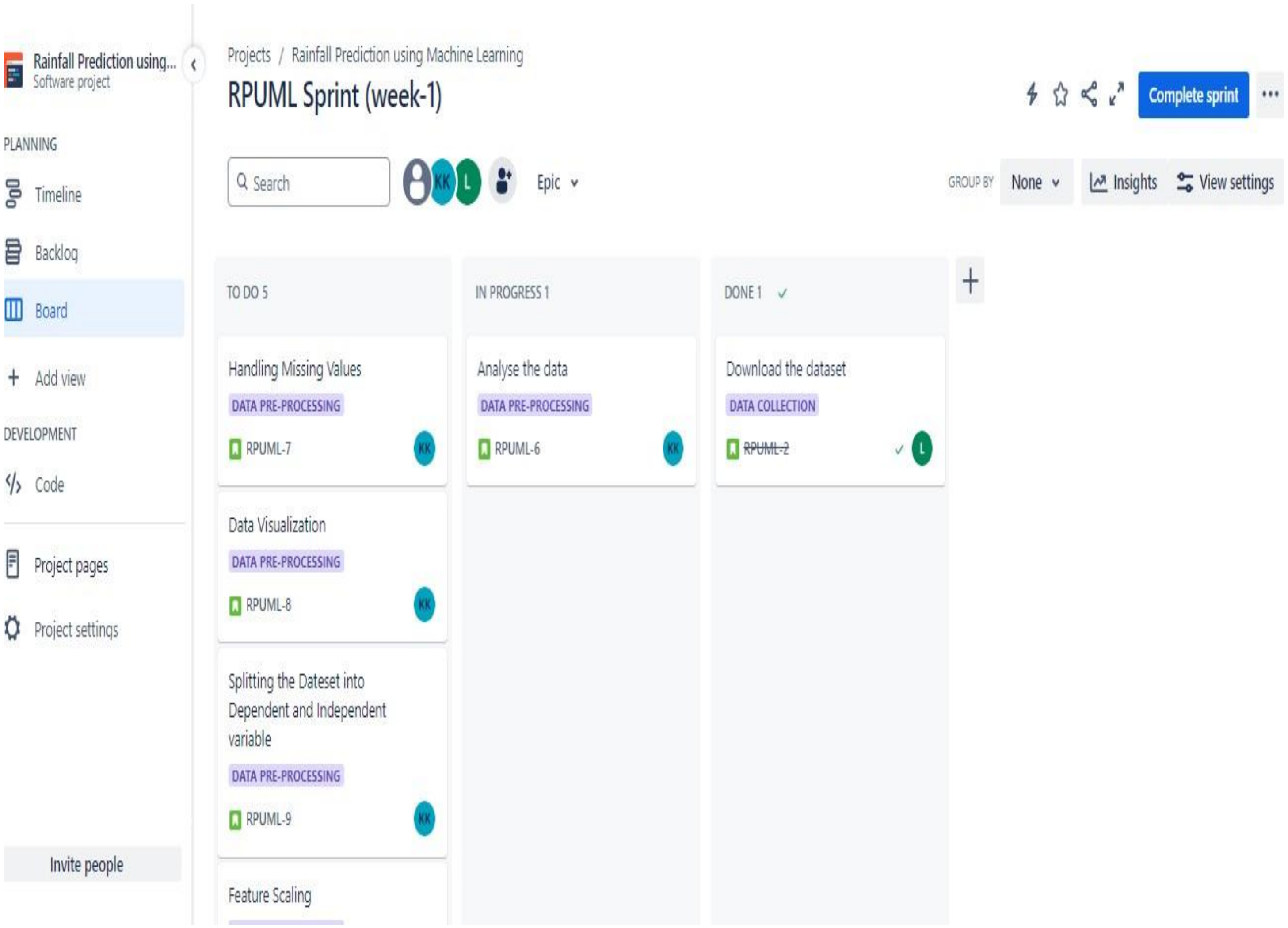
□ RPUML-4 Import Necessary Li... DONE

RPUML Sprint (week-1)

ACTIVE SPRINT

Sprint goal goes here

Sprint start	Sprint end
2024/09/20	2024/09/27



Projects / Rainfall Prediction using Machine Learning

RPUML Sprint (week-1)

TO DO 5

- Handling Missing Values
  DATA PRE-PROCESSING
  RPUML-7 KK
- Data Visualization
  DATA PRE-PROCESSING
  RPUML-8 KK
- Splitting the Dateset into Dependent and Independent variable
  DATA PRE-PROCESSING
  RPUML-9 KK
- Feature Scaling

IN PROGRESS 1

- Analyse the data
  DATA PRE-PROCESSING
  RPUML-6 KK

DONE 1 ✓

- Download the dataset
  DATA COLLECTION
  ~~RPUML-2~~ ✓ L

3. Data Collection and Preprocessing Phase

3.1 Data Collection Plan & Raw Data Sources Identified

Elevate your data strategy with the Data Collection plan and the Raw Data Sources report, ensuring meticulous data curation and integrity for informed decision-making in every analysis and decision-making endeavor.

Data Collection Plan

Section	Description
Project Overview	Rainfall prediction using machine learning entails examining historical weather data to predict future precipitation. By employing advanced algorithms such as Decision Trees, Random Forest, and Neural Networks, we achieve remarkable a ccuracy in forecasting rainfall patterns. This significantly supports agricultural p lanning, water resource management, and disaster preparedness, leading to more informed and effective decision-making.
Data Collection Plan	• Searching for Datasets: Look for datasets related to rainfall occurrence from reliable sources like meteorological departments, online databases (e.g., NOAA, OpenWeatherMap), and research institutions. Prioritize datasets that include comprehensive weather metrics over an extended period. • Prioritize dataset with various demographic information
Raw Data Sources Identified	Gather extensive historical weather data, including temperature, humidity, wind speed, and past rainfall records, from reliable sources like local meteorological s tations, national meteorological databases, and online platforms such as NOAA or OpenWeatherMap. Ensure data spans multiple years to capture seasonal and annual variations.

Raw Data Sources

Source Name	Description	Location/URL	Format	Size	Access Permissions
Dataset 1	Kaggle	https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package?select=weatherAUS.csv	CSV	14 MB	Public
Dataset 2	Kaggle	https://www.kaggle.com/datasets/rajanand/rainfall-in-india	CSV	192 KB	Public

3.2 Data Quality Report

The Data Quality Report Template will summarize data quality issues from the selected source, including severity levels and resolution plans. It will aid in systematically identifying and rectifying data discrepancies.

Data Source	Data Quality Issue	Severity	Resolution Plan
Kaggle Dataset	Missing values in the “MinTemp”,”MaxTemp”,”Rainfall”,”Evaporation”,”Sunshine”,”WindGustDir”,”WindGustSpeed”,”WindDir9am”,”WindDir3pm”,”WindSpeed9am”,”WindSpeed3pm”, ”Humidity9am”,”Humidity3pm”,“Pressure9am”,”Pressure3pm”,”Cloud9am”, ”Cloud3pm”, ”Temp9am”, ”Temp3pm”, ”RainToday”, ”RainTomorrow”	Moderate	Use mean/mode Imputation
Kaggle Dataset	Categorical data in the dataset	Moderate	Encoding has to be done in the data.

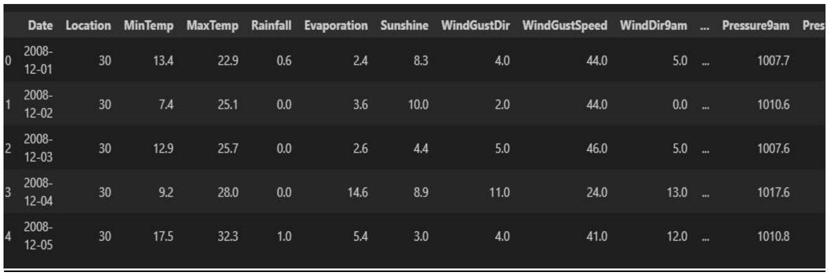
3.3 Data Exploration and Preprocessing

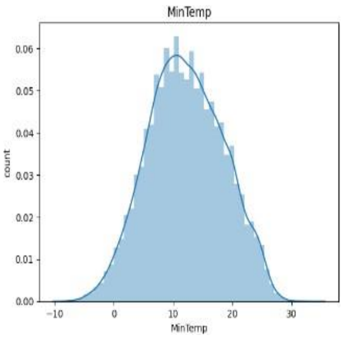
Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

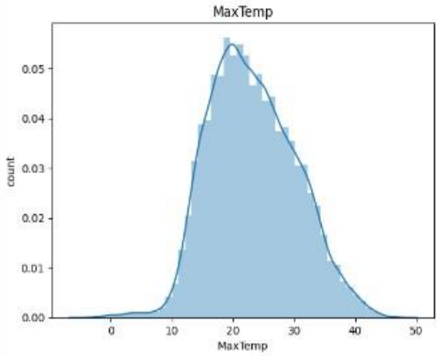
<table>
<tr><th>Section</th><th>Description</th></tr>
<tr><td>Data Overview</td><td>
<u>Dimension:</u>

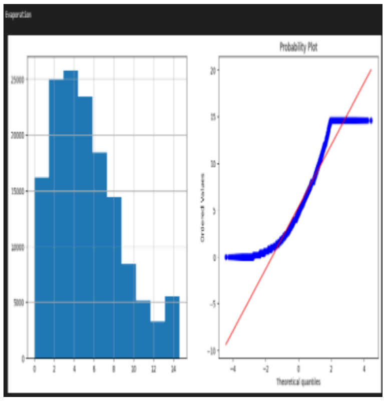
145460 rows × 23 columns

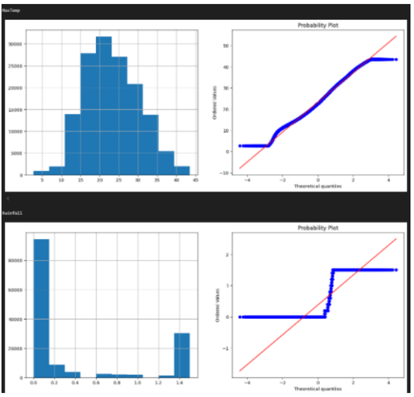
<u>Descriptive statistics:</u>



<table>
<tr><th></th><th>Date</th><th>Location</th><th>MinTemp</th><th>MaxTemp</th><th>Rainfall</th><th>Evaporation</th><th>Sunshine</th><th>WindGustDir</th><th>WindGustSpeed</th><th>WindDir9am</th><th>...</th><th>Pressure9am</th><th>Pres</th></tr>
<tr><td>0</td><td>2008-12-01</td><td>30</td><td>13.4</td><td>22.9</td><td>0.6</td><td>2.4</td><td>8.3</td><td>4.0</td><td>44.0</td><td>5.0</td><td>...</td><td>1007.7</td><td></td></tr>
<tr><td>1</td><td>2008-12-02</td><td>30</td><td>7.4</td><td>25.1</td><td>0.0</td><td>3.6</td><td>10.0</td><td>2.0</td><td>44.0</td><td>0.0</td><td>...</td><td>1010.6</td><td></td></tr>
<tr><td>2</td><td>2008-12-03</td><td>30</td><td>12.9</td><td>25.7</td><td>0.0</td><td>2.6</td><td>4.4</td><td>5.0</td><td>46.0</td><td>5.0</td><td>...</td><td>1007.6</td><td></td></tr>
<tr><td>3</td><td>2008-12-04</td><td>30</td><td>9.2</td><td>28.0</td><td>0.0</td><td>14.6</td><td>8.9</td><td>11.0</td><td>24.0</td><td>13.0</td><td>...</td><td>1017.6</td><td></td></tr>
<tr><td>4</td><td>2008-12-05</td><td>30</td><td>17.5</td><td>32.3</td><td>1.0</td><td>5.4</td><td>3.0</td><td>4.0</td><td>41.0</td><td>12.0</td><td>...</td><td>1010.8</td><td></td></tr>
</table>
</td></tr>
<tr><td>Univariate Analysis</td><td>


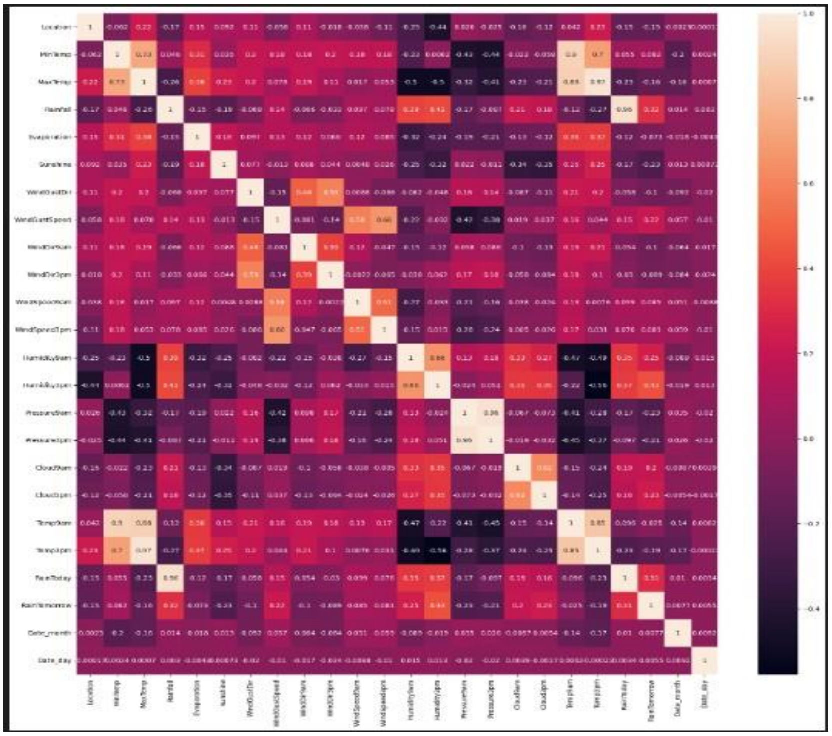


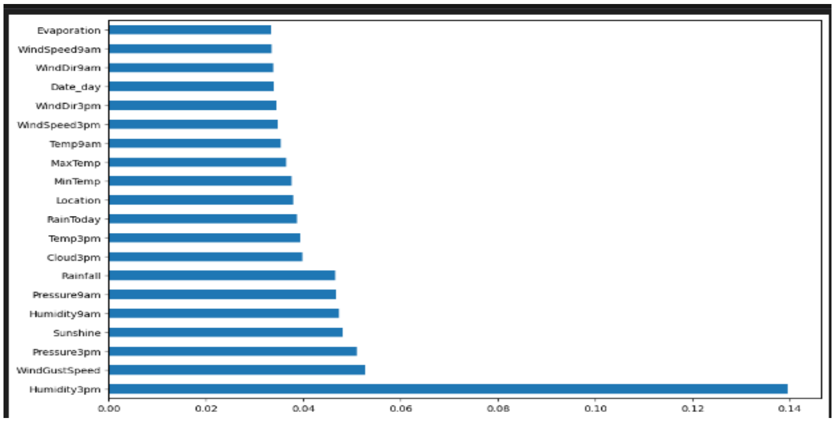
</td></tr>
<tr><td>Bivariate Analysis</td><td>




</td></tr>
</table>

Multivariate Analysis



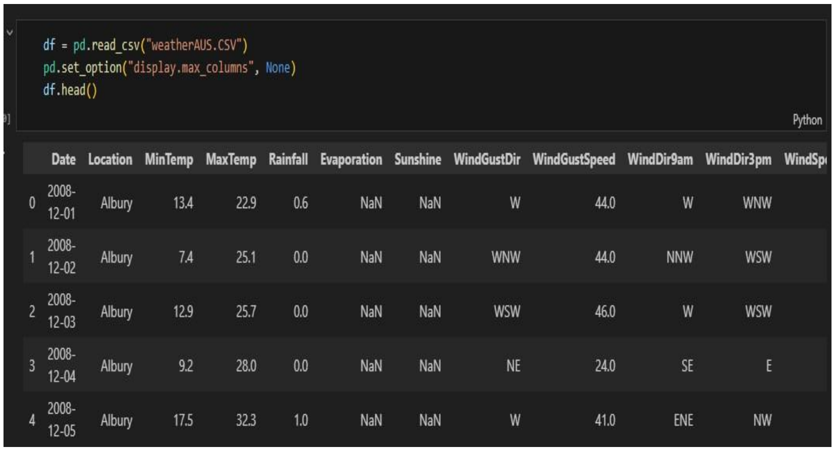


Outliers and Anomalies

-

Data Preprocessing Code Screenshots

Loading Data



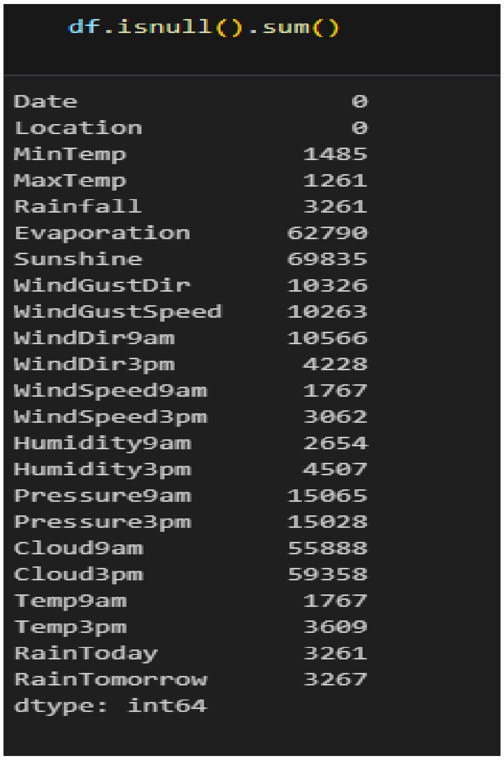
```
df = pd.read_csv("weatherAUS.CSV")
pd.set_option("display.max_columns", None)
df.head()
```

Python

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSp
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	WNW	
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	WSW	
2	2008-12-03	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	WSW	
3	2008-12-04	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	E	
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	NW	

Handling Missing Data

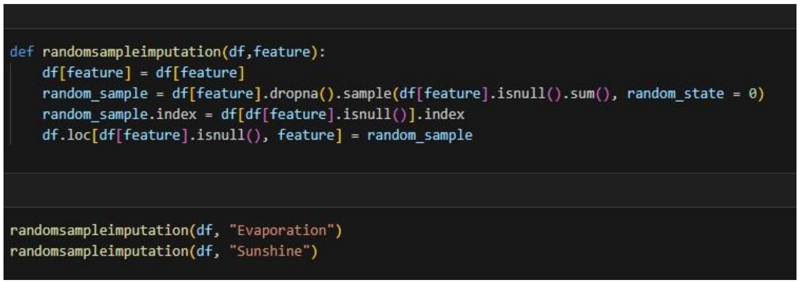
Identifying missing values.



```
df.isnull().sum()
```

```
Date                  0
Location              0
MinTemp            1485
MaxTemp            1261
Rainfall           3261
Evaporation       62790
Sunshine          69835
WindGustDir       10326
WindGustSpeed     10263
WindDir9am        10566
WindDir3pm         4228
WindSpeed9am       1767
WindSpeed3pm       3062
Humidity9am        2654
Humidity3pm        4507
Pressure9am       15065
Pressure3pm       15028
Cloud9am          55888
Cloud3pm          59358
Temp9am            1767
Temp3pm            3609
RainToday          3261
RainTomorrow       3267
dtype: int64
```

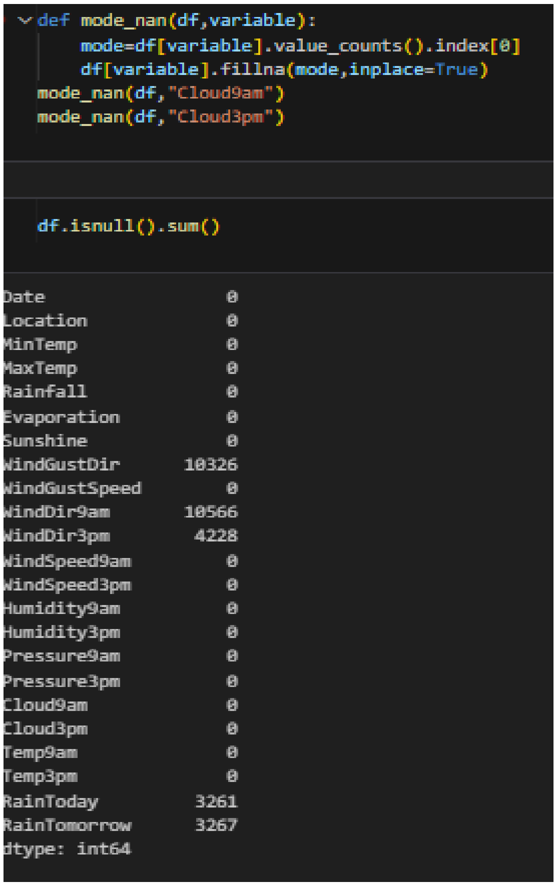
Handling missing values



```
def randomsampleimputation(df,feature):
    df[feature] = df[feature]
    random_sample = df[feature].dropna().sample(df[feature].isnull().sum(), random_state = 0)
    random_sample.index = df[df[feature].isnull()].index
    df.loc[df[feature].isnull(), feature] = random_sample
```

```
randomsampleimputation(df, "Evaporation")
randomsampleimputation(df, "Sunshine")
```

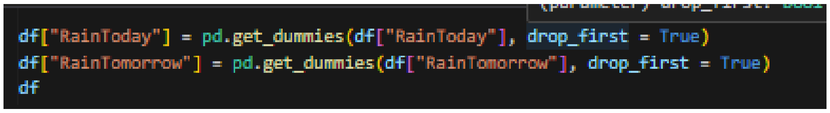
Data Transformation



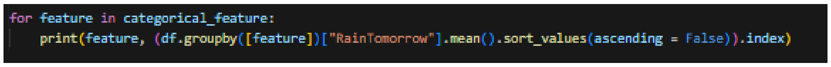
```
def mode_nan(df,variable):
    mode=df[variable].value_counts().index[0]
    df[variable].fillna(mode,inplace=True)
mode_nan(df,"Cloud9am")
mode_nan(df,"Cloud3pm")
```

```
df.isnull().sum()
```

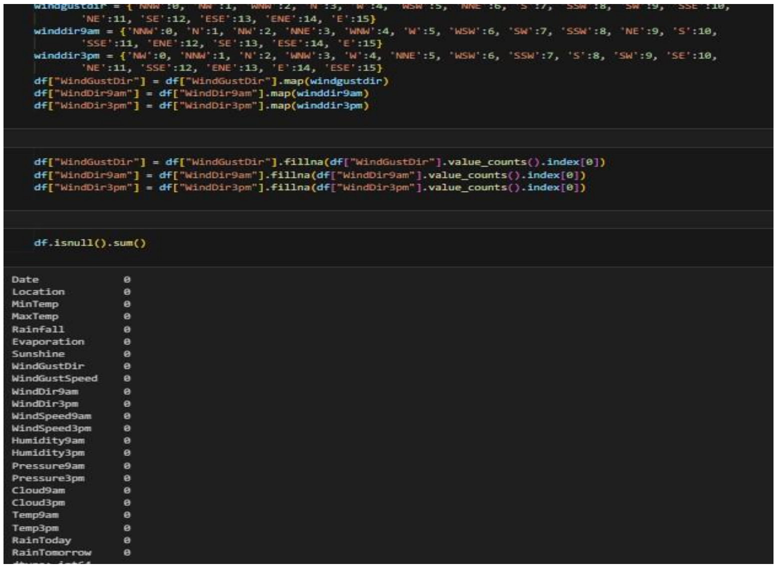
```
Date                 0
Location             0
MinTemp              0
MaxTemp              0
Rainfall             0
Evaporation          0
Sunshine             0
WindGustDir      10326
WindGustSpeed        0
WindDir9am       10566
WindDir3pm        4228
WindSpeed9am         0
WindSpeed3pm         0
Humidity9am          0
Humidity3pm          0
Pressure9am          0
Pressure3pm          0
Cloud9am             0
Cloud3pm             0
Temp9am              0
Temp3pm              0
RainToday         3261
RainTomorrow      3267
dtype: int64
```



```
df["RainToday"] = pd.get_dummies(df["RainToday"], drop_first = True)
df["RainTomorrow"] = pd.get_dummies(df["RainTomorrow"], drop_first = True)
df
```



```
for feature in categorical_feature:
    print(feature, (df.groupby([feature])["RainTomorrow"].mean().sort_values(ascending = False)).index)
```

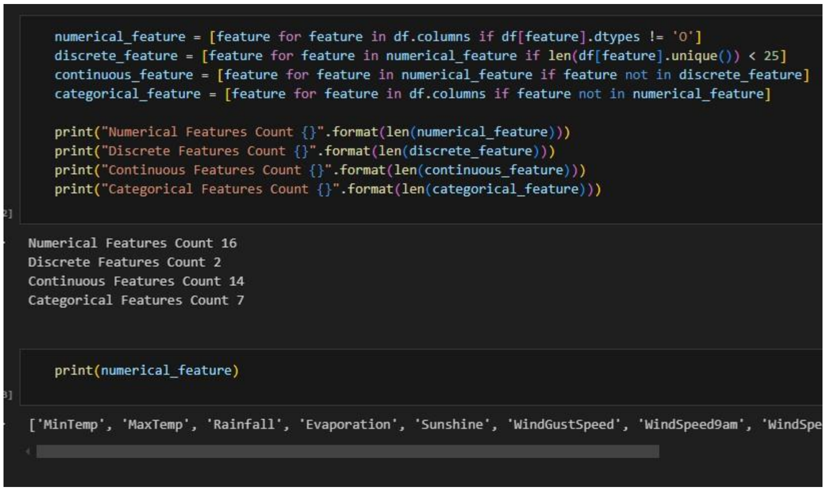
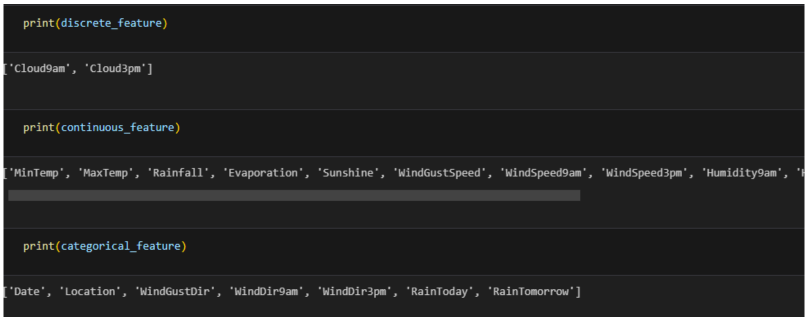


```
windgustdir = {'NNW':0, 'NW':1, 'WNW':2, 'N':3, 'W':4, 'WSW':5, 'NNE':6, 'S':7, 'SSW':8, 'SW':9, 'SSE':10,
        'NE':11, 'SE':12, 'ESE':13, 'ENE':14, 'E':15}
winddir9am = {'NNW':0, 'N':1, 'NW':2, 'NNE':3, 'WNW':4, 'W':5, 'WSW':6, 'SW':7, 'SSW':8, 'NE':9, 'S':10,
        'SSE':11, 'ENE':12, 'SE':13, 'ESE':14, 'E':15}
winddir3pm = {'NW':0, 'NNW':1, 'N':2, 'WNW':3, 'W':4, 'NNE':5, 'WSW':6, 'SSW':7, 'S':8, 'SW':9, 'SE':10,
        'NE':11, 'SSE':12, 'ENE':13, 'E':14, 'ESE':15}
df["WindGustDir"] = df["WindGustDir"].map(windgustdir)
df["WindDir9am"] = df["WindDir9am"].map(winddir9am)
df["WindDir3pm"] = df["WindDir3pm"].map(winddir3pm)
```

```
df["WindGustDir"] = df["WindGustDir"].fillna(df["WindGustDir"].value_counts().index[0])
df["WindDir9am"] = df["WindDir9am"].fillna(df["WindDir9am"].value_counts().index[0])
df["WindDir3pm"] = df["WindDir3pm"].fillna(df["WindDir3pm"].value_counts().index[0])
```

```
df.isnull().sum()
```

```
Date             0
Location         0
MinTemp          0
MaxTemp          0
Rainfall         0
Evaporation      0
Sunshine         0
WindGustDir      0
WindGustSpeed    0
WindDir9am       0
WindDir3pm       0
WindSpeed9am     0
WindSpeed3pm     0
Humidity9am      0
Humidity3pm      0
Pressure9am      0
Pressure3pm      0
Cloud9am         0
Cloud3pm         0
Temp9am          0
Temp3pm          0
RainToday        0
RainTomorrow     0
```

Feature Engineering	
	`numerical_feature = [feature for feature in df.columns if df[feature].dtypes != 'O']`
	`discrete_feature = [feature for feature in numerical_feature if len(df[feature].unique()) < 25]`
	`continuous_feature = [feature for feature in numerical_feature if feature not in discrete_feature]`
	`categorical_feature = [feature for feature in df.columns if feature not in numerical_feature]`
	
	`print("Numerical Features Count {}".format(len(numerical_feature)))`
	`print("Discrete Features Count {}".format(len(discrete_feature)))`
	`print("Continuous Features Count {}".format(len(continuous_feature)))`
	`print("Categorical Features Count {}".format(len(categorical_feature)))`
	
	`Numerical Features Count 16`
	`Discrete Features Count 2`
	`Continuous Features Count 14`
	`Categorical Features Count 7`
	
	`print(numerical_feature)`
	
	`['MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustSpeed', 'WindSpeed9am', 'WindSpe`
	
	
	`print(discrete_feature)`
	
	`['Cloud9am', 'Cloud3pm']`
	
	`print(continuous_feature)`
	
	`['MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustSpeed', 'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'H`
	
	`print(categorical_feature)`
	
	`['Date', 'Location', 'WindGustDir', 'WindDir9am', 'WindDir3pm', 'RainToday', 'RainTomorrow']`

Save Processed Data	-

4. Model Development Phase

4.1 Feature Selection Report

In the forthcoming update, each feature will be accompanied by a brief description. Users will indicate whether it's selected or not, providing reasoning for their decision. This process will streamline decision-making and enhance transparency in feature selection.

Feature	Description	Selected (Yes/No)	Reasoning
Date	The date of the recorded observation.	No	Lower correlation with target column.
Location	The geographic location of the observation.	No	Explanation of why it was selected or excluded
MinTemp	The minimum temperature recorded for the day	Yes	Influences daily weather predictions.
MaxTemp	The maximum temperature recorded for the day	No	Lower correlation with target column.
Rainfall	The amount of rainfall recorded for the day.	Yes	Direct measure of precipitation.
Evaporation	The amount of evaporation measured for the day.	No	Lower correlation with target column.

Sunshine	The number of sunshine hours recorded for the day.	No	Lower correlation with target column.
WindGustDir	The direction of the strongest wind gust recorded.	Yes	Gust direction indicates storm paths**.**
WindGustSpeed	The speed of the strongest wind gust recorded.	Yes	Indicates potential for extreme weather.
WindDir9am	The wind direction recorded at 9 AM	No	Lower correlation with target column.
WindDir3pm	The wind direction recorded at 3 PM.	No	Lower correlation with target column.
WindSpeed9am	The wind speed recorded at 9 AM.	Yes	Morning wind patterns influence daily weather.
WindSpeed3pm	The wind speed recorded at 3 PM.	Yes	Afternoon wind patterns provide for existing data.
Humidity9am	The Humidity percentage recorded at 9 AM.	Yes	Morning humidity influences daily weather.
Humidity3pm	The Humidity percentage recorded at 3 PM.	Yes	Directly affects precipitation predictions.
Pressure9am	The atmospheric pressure recorded at 9 AM.	No	Lower correlation with target column.
Pressure3pm	The atmospheric pressure recorded at 3 PM.	No	Lower correlation with target column.

Cloud9am	The cloud cover recorded at 9 AM.	Yes	Lower correlation with target column.
Cloud3pm	The cloud cover recorded at 3 PM	Yes	Morning cloud cover affects weather outcomes.
Temp9am	The temperature recorded at 9 AM.	No	Lower correlation with target column.
Temp3pm	The temperature recorded at 3 PM.	No	Lower correlation with target column.
RainToday	Indicates if it rained today.	No	High correlation but redundant with Rainfall column already providing relevant data
RainTomorrow	Predicts if it will rain tomorrow.	Yes	The target variable for predictive modelling – is essential for project goals.

4.2 Model Selection Report

In the forthcoming Model Selection Report, various models will be outlined, detailing their descriptions, hyperparameters, and performance metrics, including Accuracy or F1 Score. This comprehensive report will provide insights into the chosen models and their effectiveness.

Model Selection Report:

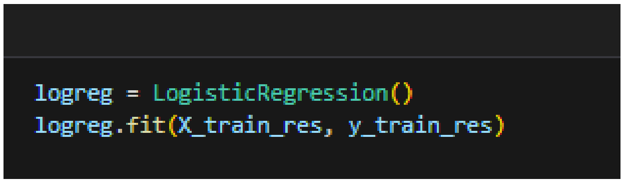
Model	Description	Hyperparameters	Performance Metric (e.g., Accuracy, F1 Score)
Random Forest	Builds multiple decision trees and averages them for robust predictions.	-	Accuracy Score = 82%
Decision Tree	Uses a tree-like structure to make decisions based on feature splits.	-	Accuracy Score = 80%
K Nearest Neighbour	Predicts the class based on the majority vote of the 'k' nearest neighbors.	-	Accuracy Score = 75%
Logistic Regression	Applies regression techniques to classify binary targets.	-	Accuracy Score = 76%
XGBoost	Efficient, high-performance	-	Accuracy Score = 84%

	gradient boosting classifier.		
SVC	Finds the best hyperplane for separating classes in n-dimensional space.	-	Accuracy Score = 76%
CatBoost	Gradient boosting optimized for handling categorical features without much preprocessing.	-	Accuracy Score = 85%

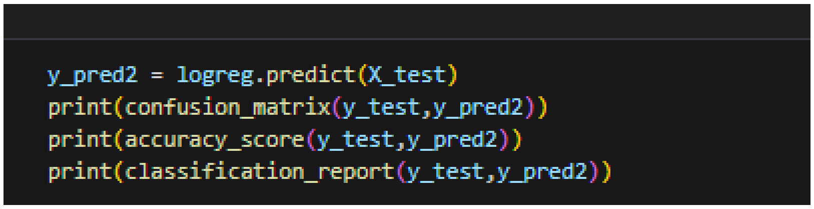
4.3 Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

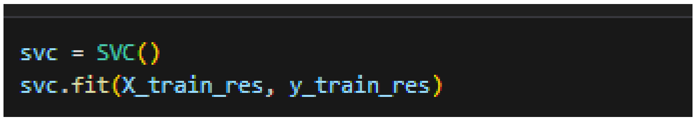
Initial Model Training Code:



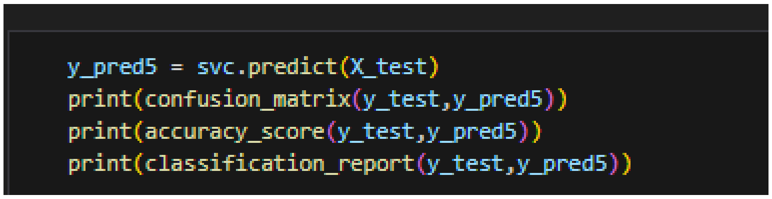
```
logreg = LogisticRegression()
logreg.fit(X_train_res, y_train_res)
```



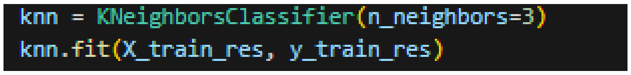
```
y_pred2 = logreg.predict(X_test)
print(confusion_matrix(y_test,y_pred2))
print(accuracy_score(y_test,y_pred2))
print(classification_report(y_test,y_pred2))
```



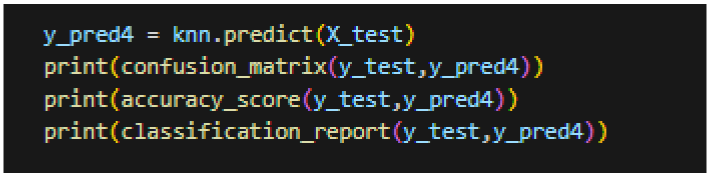
```
svc = SVC()
svc.fit(X_train_res, y_train_res)
```



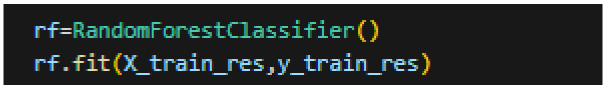
```
y_pred5 = svc.predict(X_test)
print(confusion_matrix(y_test,y_pred5))
print(accuracy_score(y_test,y_pred5))
print(classification_report(y_test,y_pred5))
```



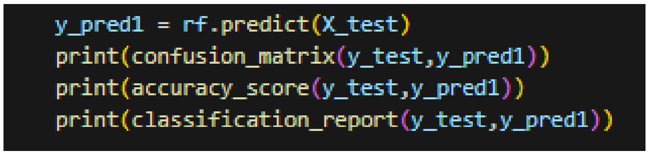
```
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train_res, y_train_res)
```



```
y_pred4 = knn.predict(X_test)
print(confusion_matrix(y_test,y_pred4))
print(accuracy_score(y_test,y_pred4))
print(classification_report(y_test,y_pred4))
```



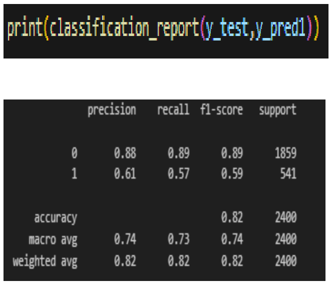
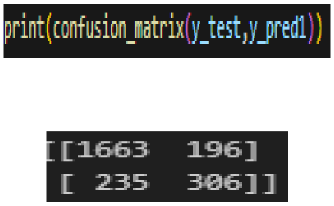
```
rf=RandomForestClassifier()
rf.fit(X_train_res,y_train_res)
```

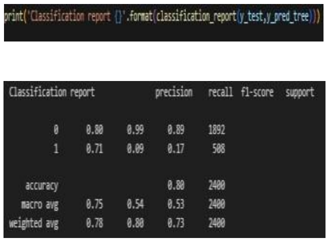
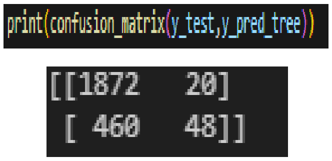


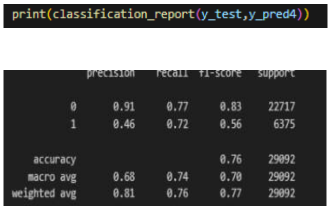
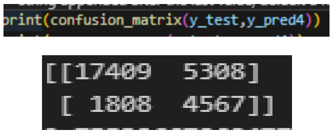
```
y_pred1 = rf.predict(X_test)
print(confusion_matrix(y_test,y_pred1))
print(accuracy_score(y_test,y_pred1))
print(classification_report(y_test,y_pred1))
```

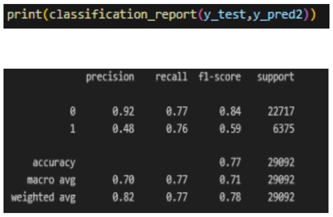
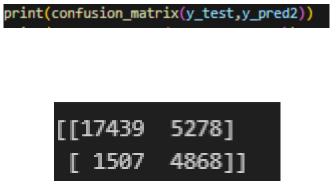
Model Validation and Evaluation Report:

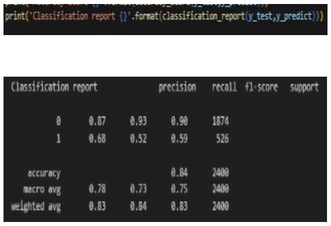
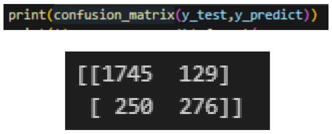
Model	Classification Report	Accuracy	Confusion Matrix

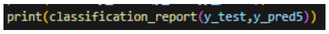
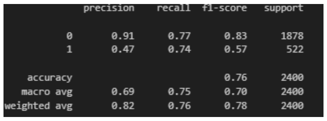
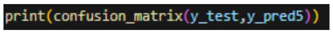
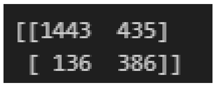
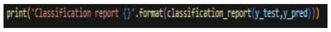
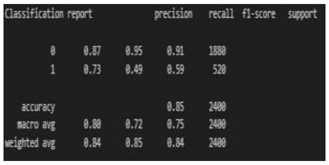
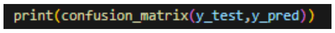
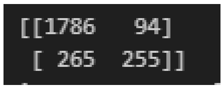
Random Forest	
	`print(classification_report(y_test,y_pred1))`
	`precision recall f1-score support`
	`0 0.88 0.89 0.89 1859`
	`1 0.61 0.57 0.59 541`
	`accuracy 0.82 2400`
	`macro avg 0.74 0.73 0.74 2400`
	`weighted avg 0.82 0.82 0.82 2400`
		82%	
			`print(confusion_matrix(y_test,y_pred1))`
			`[[1663 196]`
			`[ 235 306]]`

Decision Tree	
	`print('Classification report {}'.format(classification_report(y_test,y_pred_tree)))`
	`Classification report precision recall f1-score support`
	`0 0.80 0.99 0.89 1892`
	`1 0.71 0.09 0.17 508`
	`accuracy 0.80 2400`
	`macro avg 0.75 0.54 0.53 2400`
	`weighted avg 0.78 0.80 0.73 2400`
		80%	
			`print(confusion_matrix(y_test,y_pred_tree))`
			`[[1872 20]`
			`[ 460 48]]`

K Nearest Neighbour	
	`print(classification_report(y_test,y_pred4))`
	`precision recall f1-score support`
	`0 0.91 0.77 0.83 22717`
	`1 0.46 0.72 0.56 6375`
	`accuracy 0.76 29092`
	`macro avg 0.68 0.74 0.70 29092`
	`weighted avg 0.81 0.76 0.77 29092`
		75%	
			`print(confusion_matrix(y_test,y_pred4))`
			`[[17409 5308]`
			`[ 1808 4567]]`

Logistic Regression	
	`print(classification_report(y_test,y_pred2))`
	`precision recall f1-score support`
	`0 0.92 0.77 0.84 22717`
	`1 0.48 0.76 0.59 6375`
	`accuracy 0.77 29092`
	`macro avg 0.70 0.77 0.71 29092`
	`weighted avg 0.82 0.77 0.78 29092`
		76%	
			`print(confusion_matrix(y_test,y_pred2))`
			`[[17439 5278]`
			`[ 1507 4868]]`

XGBoost	
	`print('Classification report {}'.format(classification_report(y_test,y_predict)))`
	`Classification report precision recall f1-score support`
	`0 0.87 0.93 0.90 1874`
	`1 0.68 0.52 0.59 526`
	`accuracy 0.84 2400`
	`macro avg 0.78 0.73 0.75 2400`
	`weighted avg 0.83 0.84 0.83 2400`
		84%	
			`print(confusion_matrix(y_test,y_predict))`
			`[[1745 129]`
			`[ 250 276]]`

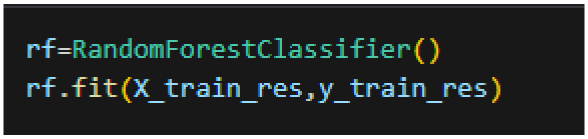
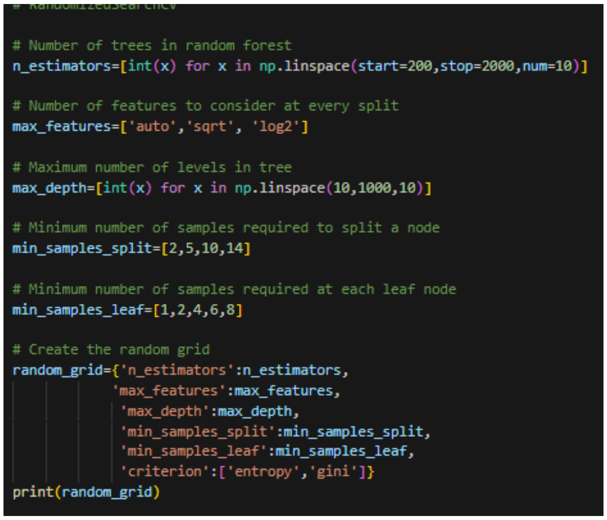
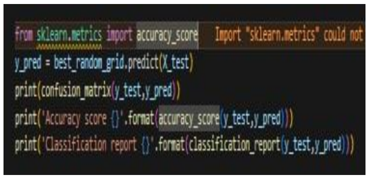
SVC	 print(classification_report(y_test,y_pred5))  precision recall f1-score support 0 0.91 0.77 0.83 1878 1 0.47 0.74 0.57 522 accuracy 0.76 2400 macro avg 0.69 0.75 0.70 2400 weighted avg 0.82 0.76 0.78 2400	76%	 print(confusion_matrix(y_test,y_pred5))  [[1443 435] [136 386]]
CatBoost	 print('Classification report {}'.format(classification_report(y_test,y_pred)))  Classification report precision recall f1-score support 0 0.87 0.95 0.91 1880 1 0.73 0.49 0.59 520 accuracy 0.85 2400 macro avg 0.80 0.72 0.75 2400 weighted avg 0.84 0.85 0.84 2400	85%	 print(confusion_matrix(y_test,y_pred))  [[1786 94] [265 255]]

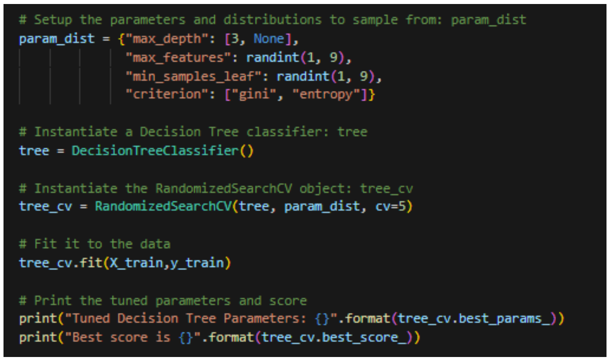
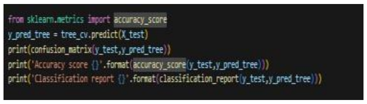
5. Model Optimization and Tuning Phase

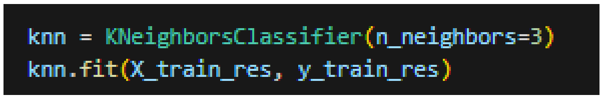
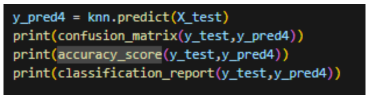
The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

5.1 Hyperparameter Tuning Documentation

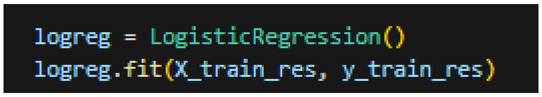
Model	Tuned Hyperparameters	Optimal Values

Random Forest	
	`rf=RandomForestClassifier()`
	`rf.fit(X_train_res,y_train_res)`
	
	`# RandomizedSearchCV`
	`# Number of trees in random forest`
	`n_estimators=[int(x) for x in np.linspace(start=200,stop=2000,num=10)]`
	`# Number of features to consider at every split`
	`max_features=['auto','sqrt', 'log2']`
	`# Maximum number of levels in tree`
	`max_depth=[int(x) for x in np.linspace(10,1000,10)]`
	`# Minimum number of samples required to split a node`
	`min_samples_split=[2,5,10,14]`
	`# Minimum number of samples required at each leaf node`
	`min_samples_leaf=[1,2,4,6,8]`
	`# Create the random grid`
	`random_grid={'n_estimators':n_estimators,`
	`'max_features':max_features,`
	`'max_depth':max_depth,`
	`'min_samples_split':min_samples_split,`
	`'min_samples_leaf':min_samples_leaf,`
	`'criterion':['entropy','gini']}`
	`print(random_grid)`
		
		`from sklearn.metrics import accuracy_score   Import "sklearn.metrics" could not`
		`y_pred = best_random_grid.predict(X_test)`
		`print(confusion_matrix(y_test,y_pred))`
		`print('Accuracy score {}'.format(accuracy_score(y_test,y_pred)))`
		`print('Classification report {}'.format(classification_report(y_test,y_pred)))`

Decision Tree	
	`# Setup the parameters and distributions to sample from: param_dist`
	`param_dist = {"max_depth": [3, None],`
	`"max_features": randint(1, 9),`
	`"min_samples_leaf": randint(1, 9),`
	`"criterion": ["gini", "entropy"]}`
	`# Instantiate a Decision Tree classifier: tree`
	`tree = DecisionTreeClassifier()`
	`# Instantiate the RandomizedSearchCV object: tree_cv`
	`tree_cv = RandomizedSearchCV(tree, param_dist, cv=5)`
	`# Fit it to the data`
	`tree_cv.fit(X_train,y_train)`
	`# Print the tuned parameters and score`
	`print("Tuned Decision Tree Parameters: {}".format(tree_cv.best_params_))`
	`print("Best score is {}".format(tree_cv.best_score_))`
		
		`from sklearn.metrics import accuracy_score`
		`y_pred_tree = tree_cv.predict(X_test)`
		`print(confusion_matrix(y_test,y_pred_tree))`
		`print('Accuracy score {}'.format(accuracy_score(y_test,y_pred_tree)))`
		`print('Classification report {}'.format(classification_report(y_test,y_pred_tree)))`

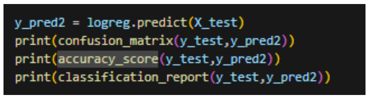
K-Neighbors Classifier	
	`knn = KNeighborsClassifier(n_neighbors=3)`
	`knn.fit(X_train_res, y_train_res)`
		
		`y_pred4 = knn.predict(X_test)`
		`print(confusion_matrix(y_test,y_pred4))`
		`print(accuracy_score(y_test,y_pred4))`
		`print(classification_report(y_test,y_pred4))`

<table>
<tr>
<td>Logestic
Regression</td>
<td>



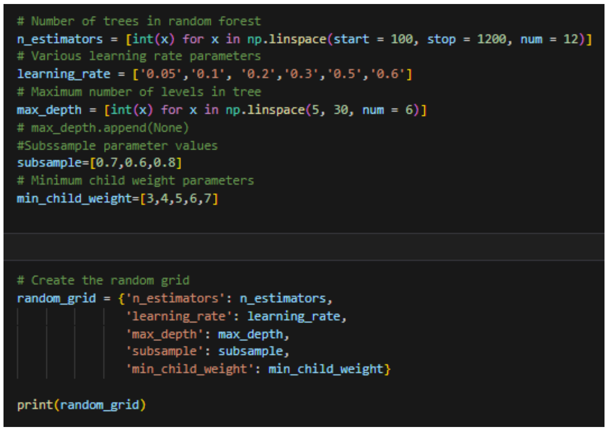
```
logreg = LogisticRegression()
logreg.fit(X_train_res, y_train_res)
```

</td>
<td>



```
y_pred2 = logreg.predict(X_test)
print(confusion_matrix(y_test,y_pred2))
print(accuracy_score(y_test,y_pred2))
print(classification_report(y_test,y_pred2))
```

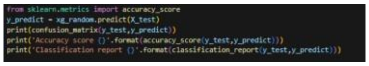
</td>
</tr>
<tr>
<td>XGBoost</td>
<td>



```
# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start = 100, stop = 1200, num = 12)]
# Various learning rate parameters
learning_rate = ['0.05','0.1', '0.2','0.3','0.5','0.6']
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(5, 30, num = 6)]
# max_depth.append(None)
#Subssample parameter values
subsample=[0.7,0.6,0.8]
# Minimum child weight parameters
min_child_weight=[3,4,5,6,7]
```

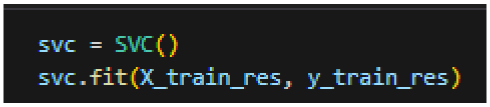
```
# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'learning_rate': learning_rate,
               'max_depth': max_depth,
               'subsample': subsample,
               'min_child_weight': min_child_weight}

print(random_grid)
```

</td>
<td>



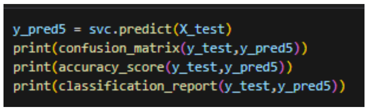
```
from sklearn.metrics import accuracy_score
y_predict = xg_random.predict(X_test)
print(confusion_matrix(y_test,y_predict))
print('Accuracy score {}'.format(accuracy_score(y_test,y_predict)))
print('Classification report {}'.format(classification_report(y_test,y_predict)))
```

</td>
</tr>
<tr>
<td>SVC</td>
<td>



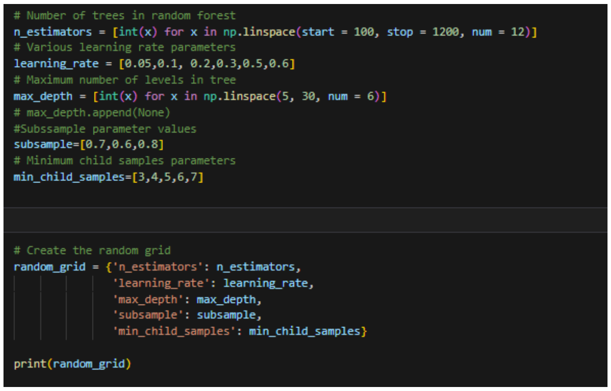
```
svc = SVC()
svc.fit(X_train_res, y_train_res)
```

</td>
<td>



```
y_pred5 = svc.predict(X_test)
print(confusion_matrix(y_test,y_pred5))
print(accuracy_score(y_test,y_pred5))
print(classification_report(y_test,y_pred5))
```

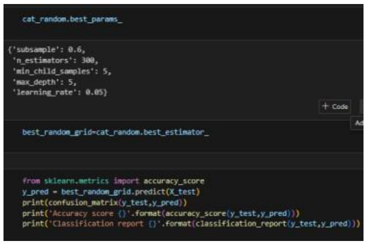
</td>
</tr>
<tr>
<td>CatBoost</td>
<td>



```
# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start = 100, stop = 1200, num = 12)]
# Various learning rate parameters
learning_rate = [0.05,0.1, 0.2,0.3,0.5,0.6]
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(5, 30, num = 6)]
# max_depth.append(None)
#Subssample parameter values
subsample=[0.7,0.6,0.8]
# Minimum child samples parameters
min_child_samples=[3,4,5,6,7]
```

```
# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'learning_rate': learning_rate,
               'max_depth': max_depth,
               'subsample': subsample,
               'min_child_samples': min_child_samples}

print(random_grid)
```

</td>
<td>



```
cat_random.best_params_
```

```
{'subsample': 0.6,
 'n_estimators': 300,
 'min_child_samples': 5,
 'max_depth': 5,
 'learning_rate': 0.05}
```

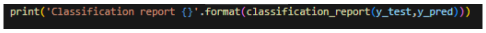
```
best_random_grid=cat_random.best_estimator_
```

```
from sklearn.metrics import accuracy_score
y_pred = best_random_grid.predict(X_test)
print(confusion_matrix(y_test,y_pred))
print('Accuracy score {}'.format(accuracy_score(y_test,y_pred)))
print('Classification report {}'.format(classification_report(y_test,y_pred)))
```

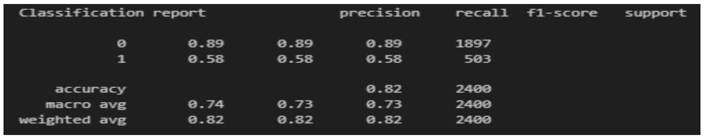
</td>
</tr>
</table>

5.2 Performance Metrics Comparison Report

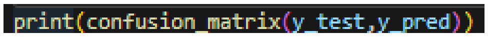
<table>
<tr><th>Model</th><th>Optimized Metric</th></tr>
<tr><td>Random Forest</td><td>



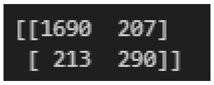
```
print('Classification report {}'.format(classification_report(y_test,y_pred)))
```



```
Classification report              precision    recall  f1-score   support

           0       0.89      0.89      0.89      1897
           1       0.58      0.58      0.58       503

    accuracy                           0.82      2400
   macro avg       0.74      0.73      0.73      2400
weighted avg       0.82      0.82      0.82      2400
```

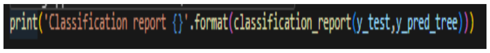


```
print(confusion_matrix(y_test,y_pred))
```

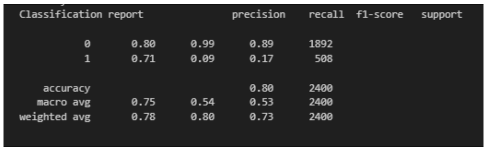


```
[[1690  207]
 [ 213  290]]
```

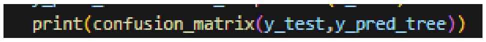
</td></tr>
<tr><td>Decision Tree</td><td>



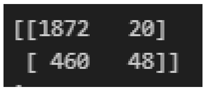
```
print('Classification report {}'.format(classification_report(y_test,y_pred_tree)))
```



```
Classification report              precision    recall  f1-score   support

           0       0.80      0.99      0.89      1892
           1       0.71      0.09      0.17       508

    accuracy                           0.80      2400
   macro avg       0.75      0.54      0.53      2400
weighted avg       0.78      0.80      0.73      2400
```

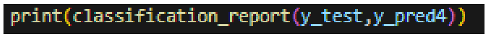


```
print(confusion_matrix(y_test,y_pred_tree))
```

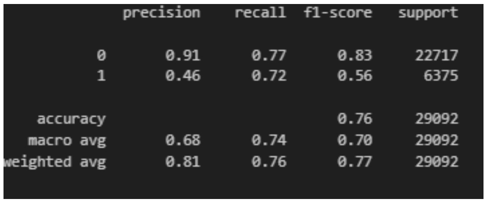


```
[[1872   20]
 [ 460   48]]
```

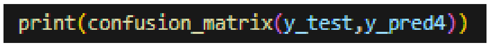
</td></tr>
<tr><td>K-Neighbors
Classifier</td><td>



```
print(classification_report(y_test,y_pred4))
```

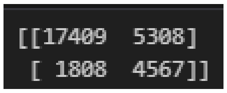


```
              precision    recall  f1-score   support

           0       0.91      0.77      0.83     22717
           1       0.46      0.72      0.56      6375

    accuracy                           0.76     29092
   macro avg       0.68      0.74      0.70     29092
weighted avg       0.81      0.76      0.77     29092
```



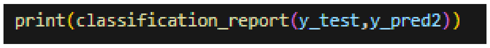
```
print(confusion_matrix(y_test,y_pred4))
```

</td></tr>
</table>

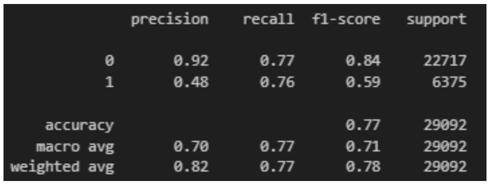


```
[[17409  5308]
 [ 1808  4567]]
```

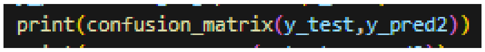
Logistic Regression



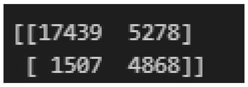
```
print(classification_report(y_test,y_pred2))
```



```
              precision    recall  f1-score   support

           0       0.92      0.77      0.84     22717
           1       0.48      0.76      0.59      6375

    accuracy                           0.77     29092
   macro avg       0.70      0.77      0.71     29092
weighted avg       0.82      0.77      0.78     29092
```

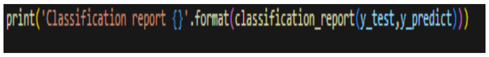


```
print(confusion_matrix(y_test,y_pred2))
```

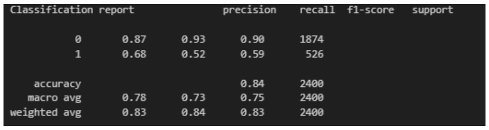


```
[[17439  5278]
 [ 1507  4868]]
```

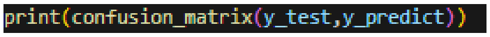
XGBoost



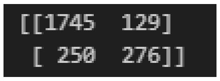
```
print('Classification report {}'.format(classification_report(y_test,y_predict)))
```



```
Classification report               precision    recall  f1-score   support

           0       0.87      0.93      0.90      1874
           1       0.68      0.52      0.59       526

    accuracy                           0.84      2400
   macro avg       0.78      0.73      0.75      2400
weighted avg       0.83      0.84      0.83      2400
```

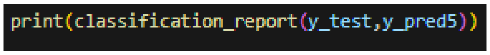


```
print(confusion_matrix(y_test,y_predict))
```

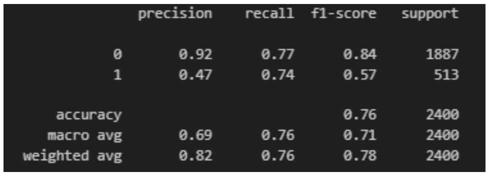


```
[[1745  129]
 [ 250  276]]
```

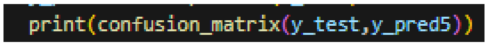
SVC



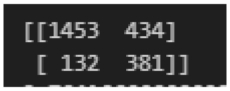
```
print(classification_report(y_test,y_pred5))
```



```
              precision    recall  f1-score   support

           0       0.92      0.77      0.84      1887
           1       0.47      0.74      0.57       513

    accuracy                           0.76      2400
   macro avg       0.69      0.76      0.71      2400
weighted avg       0.82      0.76      0.78      2400
```



```
print(confusion_matrix(y_test,y_pred5))
```



```
[[1453  434]
 [ 132  381]]
```

5.3 Final Model Selection Justification :

Final Model	Reasoning
Random Forest	The Random Forest model was selected for its superior performance, exhibiting high accuracy during hyperparameter tuning. Its ability to handle complex relationships, minimize overfitting, and optimize predictive accuracy aligns with project objectives, justifying its selection as the final model

6.Results

Output Screenshots

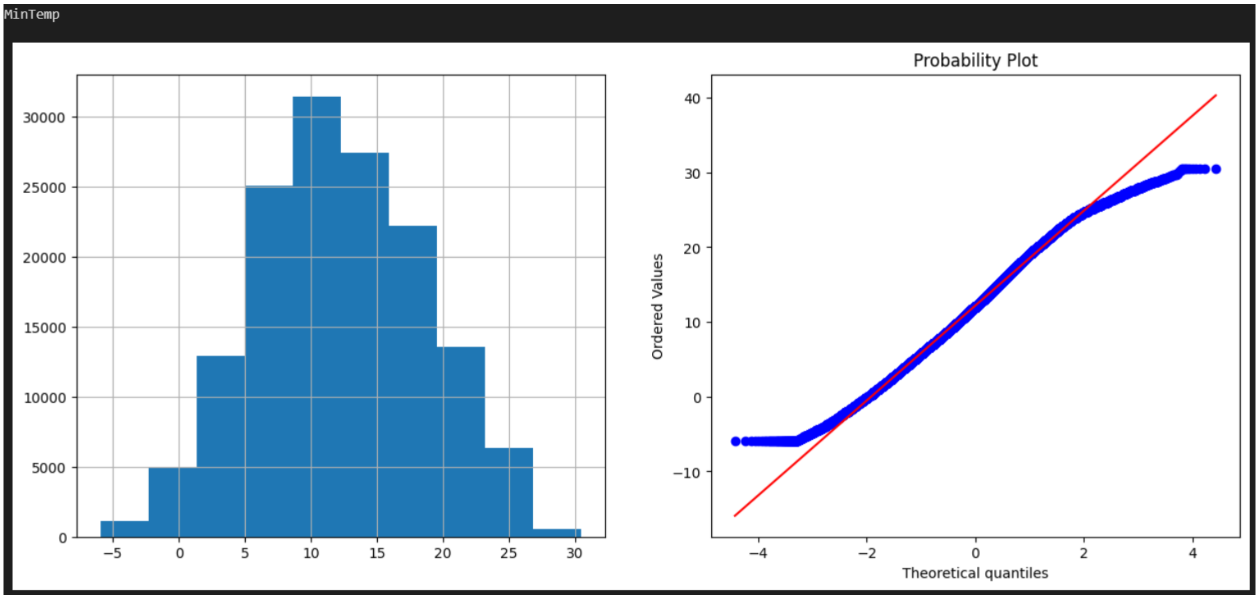
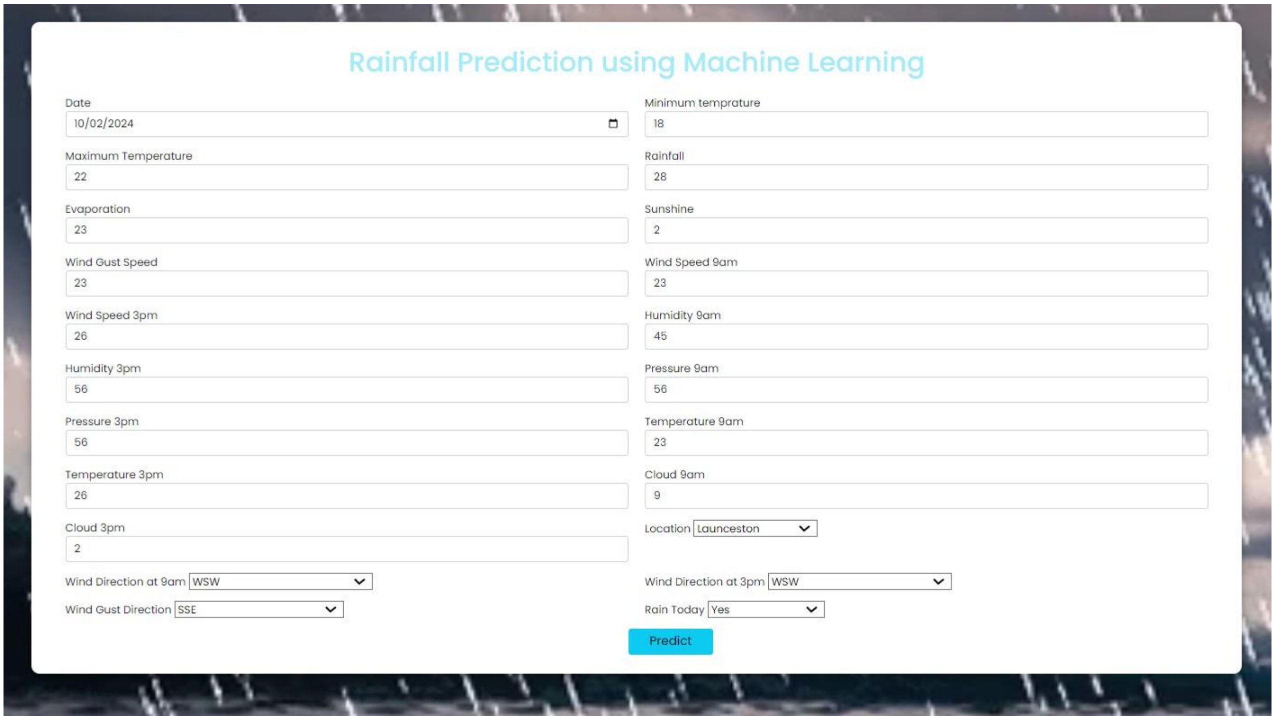


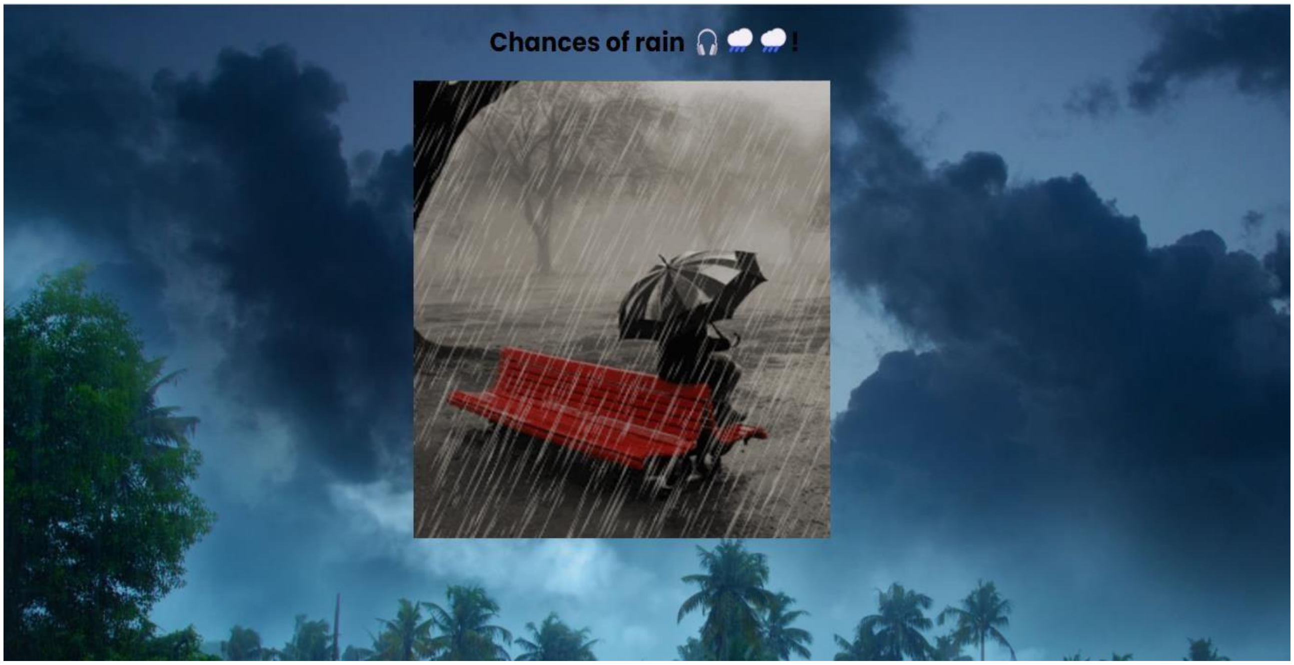
Fig: Exploratory Data Analysis



Rainfall Prediction using Machine Learning

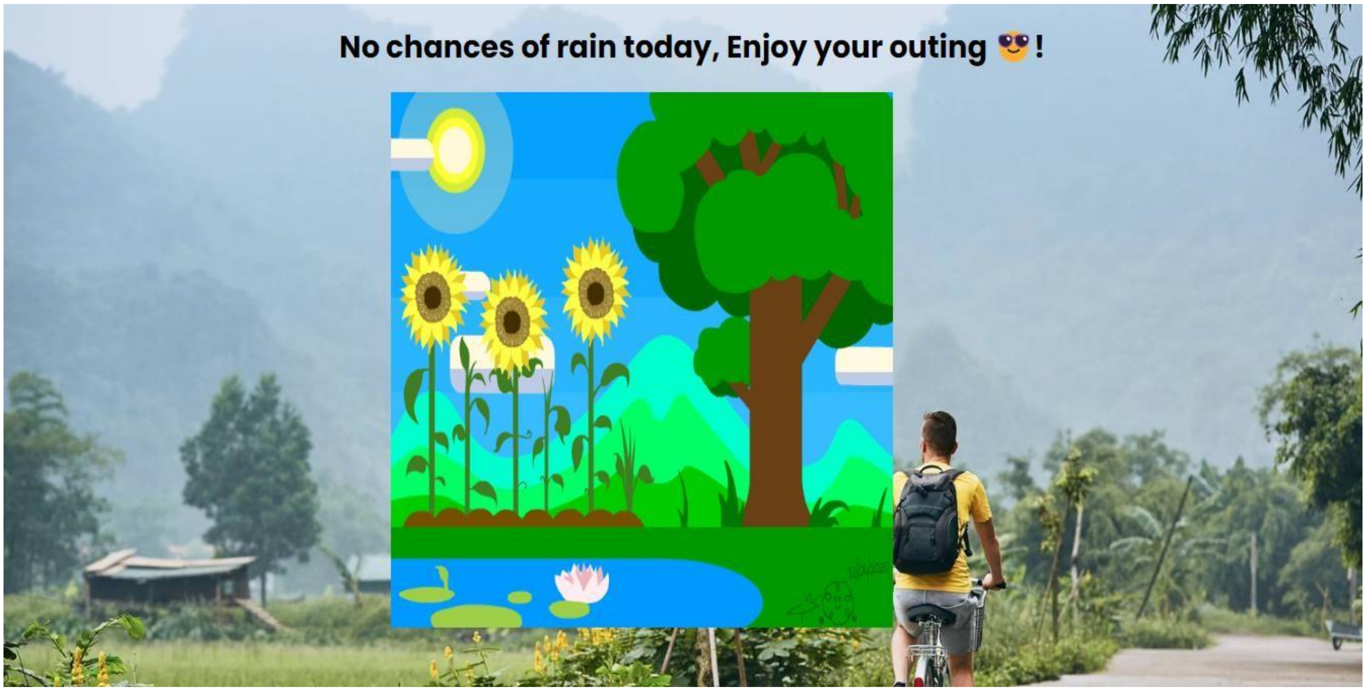
Date 10/02/2024	Minimum temprature 18
Maximum Temperature 22	Rainfall 28
Evaporation 23	Sunshine 2
Wind Gust Speed 23	Wind Speed 9am 23
Wind Speed 3pm 26	Humidity 9am 45
Humidity 3pm 56	Pressure 9am 56
Pressure 3pm 56	Temperature 9am 23
Temperature 3pm 26	Cloud 9am 9
Cloud 3pm 2	Location Launceston
Wind Direction at 9am WSW	Wind Direction at 3pm WSW
Wind Gust Direction SSE	Rain Today Yes

Fig: Home Page



Chances of rain 🎧 🌧️ 🌧️ !

Fig: User Interface



No chances of rain today, Enjoy your outing 😎!

Fig: User Interface

7. Advantages & Disadvantages

Advantages

- **Improved Accuracy**: Machine learning models can capture complex patterns in weather data that traditional statistical models may miss, leading to more accurate rainfall predictions.
- **Automation**: The predictive model can automate the process of rainfall forecasting, reducing human intervention and the chances of manual errors.
- **Adaptability**: The model can be retrained with new data, allowing it to adapt to changing weather patterns and improve over time.
- **Efficiency**: It processes large datasets quickly, providing timely predictions for sectors like agriculture, transportation, and disaster management.
- **Data-Driven Insights**: Provides deeper insights into factors influencing rainfall, helping researchers and meteorologists understand weather patterns better.

Disadvantages

- **Data Dependency**: The accuracy of the model heavily depends on the quality and quantity of historical weather data available.
- **Overfitting Risk**: Machine learning models might overfit the training data, making predictions less reliable when applied to new or unseen data.
- **Complexity**: Setting up and maintaining machine learning models requires expertise in both data science and meteorology, which can be a barrier for smaller organizations.
- **High Computational Resources**: Some advanced models, like deep learning, may require significant computational power, which could be expensive.
- **Limited by Unpredictable Events**: Sudden weather changes, such as localized storms or extreme conditions, can still be hard to predict even with machine learning.

8. Conclusion

The **Rainfall Prediction Using Machine Learning** project demonstrates how machine learning techniques can be applied to enhance the accuracy of weather forecasting, specifically rainfall prediction. With the growing availability of large-scale meteorological datasets, the use of machine learning can significantly reduce uncertainty in weather predictions. Although there are challenges, such as data quality and the complexity of the models, the benefits—like increased accuracy, efficiency, and the potential for automation—make it a valuable tool for industries reliant on weather forecasts. By incorporating machine learning, stakeholders such as farmers, city planners, and emergency services can make more informed decisions and mitigate risks related to rainfall.

9. Future Scope

The future scope for **Rainfall Prediction Using Machine Learning** is vast, with several potential areas of advancement:

- **Incorporation of Real-Time Data**: Future models can include real-time data from satellite imagery, sensors, and IoT devices, allowing for more precise and up-to-date rainfall predictions.
- **Use of Advanced Models**: More sophisticated machine learning techniques, such as deep learning and neural networks, can be explored to further improve predictive accuracy.
- **Region-Specific Models**: Developing localized models tailored to specific regions will enhance prediction accuracy by focusing on unique geographical and climatic factors.

- **Integration with Climate Change Models**: Machine learning models can be integrated with climate change simulations to predict long-term shifts in rainfall patterns, helping with sustainability and adaptation planning.
- **Cross-Disciplinary Collaboration**: Future work can focus on collaborations between meteorologists, data scientists, and other experts to improve model interpretability and practicality.
- **Scalability to Other Weather Phenomena**: The machine learning techniques used in rainfall prediction can be expanded to predict other weather phenomena like hurricanes, snow, and droughts, broadening the application of these models.

10. Appendix

10.1 Source Code

#App.py

```
# -*- coding: utf-8 -*-

from flask import Flask,render_template,url_for,request,jsonify

from flask_cors import cross_origin

import pandas as pd

import numpy as np

import datetime

import pickle

from xgboost import XGBClassifier



app = Flask(__name__, template_folder="template")

model = pickle.load(open("xg_random.pkl", "rb"))

print("Model Loaded")
```

```
@app.route("/")
@cross_origin()
def home():
return render_template("home.html")


import pandas as pd  # Ensure pandas is imported


@app.route("/predict",methods=['GET', 'POST'])
@cross_origin()
def predict():
if request.method == "POST":
# DATE
date = request.form['date']
day = float(pd.to_datetime(date, format="%Y-%m-%d").day)
month = float(pd.to_datetime(date, format="%Y-%m-%d").month)
# MinTemp
minTemp = float(request.form['mintemp'])
# MaxTemp
maxTemp = float(request.form['maxtemp'])
# Rainfall
rainfall = float(request.form['rainfall'])
# Evaporation
evaporation = float(request.form['evaporation'])
```

```
# Sunshine

sunshine = float(request.form['sunshine'])

# Wind Gust Speed

windGustSpeed = float(request.form['windgustspeed'])

# Wind Speed 9am

windSpeed9am = float(request.form['windspeed9am'])

# Wind Speed 3pm

windSpeed3pm = float(request.form['windspeed3pm'])

# Humidity 9am

humidity9am = float(request.form['humidity9am'])

# Humidity 3pm

humidity3pm = float(request.form['humidity3pm'])

# Pressure 9am

pressure9am =   float(request.form['pressure9am'])

# Pressure 3pm

pressure3pm = float(request.form['pressure3pm'])

# Temperature 9am

temp9am = float(request.form['temp9am'])

# Temperature 3pm

temp3pm = float(request.form['temp3pm'])

# Cloud 9am

cloud9am = float(request.form['cloud9am'])

# Cloud 3pm

cloud3pm = float(request.form['cloud3pm'])
```

```
# Cloud 3pm
location = float(request.form['location'])
# Wind Dir 9am
winddDir9am = float(request.form['winddir9am'])
# Wind Dir 3pm
winddDir3pm = float(request.form['winddir3pm'])
# Wind Gust Dir
windGustDir = float(request.form['windgustdir'])
# Rain Today
rainToday = float(request.form['raintoday'])


input_lst = [location , minTemp , maxTemp , rainfall , evaporation , sunshine ,
windGustDir , windGustSpeed , winddDir9am , winddDir3pm , windSpeed9am ,
windSpeed3pm ,
humidity9am , humidity3pm , pressure9am , pressure3pm , cloud9am , cloud3pm , temp9am ,
temp3pm ,
rainToday , month , day]
pred = model.predict([input_lst]) # Ensure input_lst is wrapped in a list
output = pred[0]  # Get the prediction value (assuming pred is a list/array)


if output == 0:
return render_template("sunny.html")
else:
return render_template("rainy.html")
return render_template("home.html")


if __name__=='__main__':

app.run(debug=True)
```

#Home.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="preconnect" href="https://fonts.gstatic.com">

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;400;500;600;700;800;900&display=swap" rel="stylesheet">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-BmbxuPwQa2lc/FVzBcNJ7UAyJxM6wuqIj61tLrc4wSX0szH/Ev+nYRRuWlolflfl" crossorigin="anonymous">

<link rel="stylesheet" href={{url_for('static',filename='predictor.css')}}>
```

```
<title>Rain Prediction</title>

</head>
<body>
<section id="prediction-form">
<form class="form" action="/predict", method="POST">
<h1 class="my-3 text-center">Rainfall Prediction using Machine Learning</h1>
<div class="row">
<div class="col-md-6 my-2">
<div class="md-form">
<label for="date" class="date">Date</label>
<input type="date" class="form-control" id="date" name="date">
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="mintemp" class="mintemp"> Minimum temprature</label>
<input type="text" class="form-control" id="mintemp" name="mintemp">
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
```

```
<label for="maxtemp" class="maxtemp">Maximum Temperature</label>
<input type="text" class="form-control" id="maxtemp" name="maxtemp">
</div>
</div>

<div class="col-md-6 my-2">

<div class="md-form">
<label for="rainfall" class="rainfall">Rainfall</label>
<input type="text" class="form-control" id="rainfall" name="rainfall">
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="evaporation" class="evaporation">Evaporation</label>
<input type="text" class="form-control" id="evaporation" name="evaporation">
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="sunshine" class="sunshine">Sunshine</label>
<input type="text" class="form-control" id="sunshine" name="sunshine">
```

```
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="windgustspeed" class="windgustspeed">Wind Gust Speed</label>

<input type="text" class="form-control" id="windgustspeed" name="windgustspeed">
</div>
</div>
<div class="col-md-6 my-2">

<div class="md-form">
<label for="windspeed9am" class="windspeed9am">Wind Speed 9am</label>
<input type="text" class="form-control" id="windspeed9am" name="windspeed9am">
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="windspeed3pm" class="windspeed3pm">Wind Speed 3pm</label>
<input type="text" class="form-control" id="windspeed3pm" name="windspeed3pm">
</div>
</div>
```

```
<div class="col-md-6 my-2">
<div class="md-form">
<label for="humidity9am" class="humidity9am">Humidity 9am</label>
<input type="text" class="form-control" id="humidity9am" name="humidity9am">
</div>
</div>


<div class="col-md-6 my-2">
<div class="md-form">
<label for="humidity3pm" class="humidity3pm">Humidity 3pm</label>
<input type="text" class="form-control" id="humidity3pm" name="humidity3pm">
</div>
</div>
<div class="col-md-6 my-2">

<div class="md-form">
<label for="pressure9am" class="pressure9am">Pressure 9am</label>
<input type="text" class="form-control" id="pressure9am" name="pressure9am">
</div>
</div>
<div class="col-md-6 my-2">
```

```
<div class="md-form">
<label for="pressure3pm" class="pressure3pm">Pressure 3pm</label>
<input type="text" class="form-control" id="pressure3pm" name="pressure3pm">
</div>
</div>

<div class="col-md-6 my-2">

<div class="md-form">

<label for="temp9am" class="temp9am">Temperature 9am</label>

<input type="text" class="form-control" id="temp9am" name="temp9am">

</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="temp3pm" class=temp3pm">Temperature 3pm</label>
<input type="text" class="form-control" id="temp3pm" name="temp3pm">
</div>
</div>
```

```html
<div class="col-md-6 my-2">
<div class="md-form">
<label for="cloud9am" class="cloud9am">Cloud 9am</label>
<input type="text" class="form-control" id="cloud9am" name="cloud9am">
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="cloud3pm" class="cloud3pm">Cloud 3pm</label>
<input type="text" class="form-control" id="cloud3pm" name="cloud3pm">
</div>
</div>

<div class="col-md-6 my-2">
<div class="md-form">
<label for="location" class="location" name="location">Location</label>
<select class="location" id="location" name="location" aria-label="Location">
<option selected>Select Location</option>
<option value= 24>Adelaide</option>
<option value= 7>Albany</option>
<option value= 30>Albury</option>
<option value= 46>AliceSprings</option>
```

```
<option value= 33>BadgerysCreek</option>
<option value= 14>Ballarat</option>
<option value= 36>Bendigo</option>
<option value= 21>Brisbane</option>
<option value= 2>Cairns</option>
<option value= 43>Cobar</option>
<option value= 9>CoffsHarbour</option>
<option value= 4>Dartmoor</option>
<option value= 11>Darwin</option>
<option value= 15>GoldCoast</option>
<option value= 17>Hobart</option>
<option value= 45>Katherine</option>
<option value= 23>Launceston</option>
<option value= 28>Melbourne</option>

<option value= 25>Melbourne Airport</option>
<option value= 44>Mildura</option>
<option value= 42>Moree</option>
<option value= 5>MountGambier</option>
<option value= 12>MountGinini</option>
<option value= 19>Newcastle </option>
<option value= 47>Nhil</option>
```

```
<option value= 13>NorahHead</option>
<option value= 6>NorfolkIsland</option>
<option value= 32>Nuriootpa</option>
<option value= 40>PearceRAAF</option>
<option value= 31>Penrith</option>
<option value= 26>Perth</option>
<option value= 35>Perth Airport</option>
<option value= 1>Portland</option>
<option value= 37>Richmond</option>
<option value= 27>Sale</option>
<option value= 41>Salmon Gums</option>
<option value= 10>Sydney</option>
<option value= 16>Sydney Airport</option>
<option value= 39>Townsville</option>
<option value= 34>Tuggeranong</option>

<option value= 49>Uluru</option>
<option value= 38>WaggaWagga</option>
<option value= 3>Walpole</option>
<option value= 18>Watsonia</option>
<option value= 22>William Town</option>
<option value= 8>Witchcliffe</option>
```

```html
<option value= 20>Wollongong</option>
<option value= 48>Woomera</option>
</select>
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="winddir9am" class="winddir9am" name = "winddir9am">Wind Direction at 9am</label>
<select class="winddir9am" id="winddir9am" name="winddir9am" aria-label="Wind Direction 9am">
<option selected>Select Wind Direction at 9am</option>
<option value= 1>N</option>
<option value= 5>W</option>
<option value= 10>S</option>
<option value= 15>E</option>

<option value= 2>NW</option>
<option value= 9>NE</option>
<option value= 7>SW</option>
<option value= 13>SE</option>
<option value= 0>NNW</option>
```

```
<option value= 3>NNE</option>
<option value= 8>SSW</option>
<option value= 11>SSE</option>
<option value= 4>WNW</option>
<option value= 6>WSW</option>
<option value= 12>ENE</option>
<option value= 14>ESE</option>
</select>
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="winddir3pm" class="winddir3pm" name = "winddir3pm">Wind Direction at 3pm</label>
<select class="winddir3pm" id="winddir3pm" name = "winddir3pm" aria-label="Wind Direction at 3pm">
<option selected>Select Wind Direction at 3pm</option>
<option value= 2>N</option>
<option value= 4>W</option>
<option value= 8>S</option>
<option value= 14>E</option>
<option value= 0>NW</option>
```

```
<option value= 11>NE</option>
<option value= 9>SW</option>
<option value= 10>SE</option>
<option value= 1>NNW</option>
<option value= 5>NNE</option>
<option value= 7>SSW</option>
<option value= 12>SSE</option>
<option value= 3>WNW</option>
<option value= 6>WSW</option>
<option value= 13>ENE</option>
<option value= 15>ESE</option>
</select>
</div>
</div>
<div class="col-md-6 my-2">
<div class="md-form">
<label for="windgustdir" class="windgustdir" name = "windgustdir">Wind Gust Direction</label>
<select class="windgustdir" id="windgustdir" name = "windgustdir" aria-label="Wind Gust Direction">
```

```
<option selected>Select Wind Gust Direction</option>

<option value= 3>N</option>

<option value= 4>W</option>

<option value= 7>S</option>

<option value= 15>E</option>

<option value= 1>NW</option>

<option value= 11>NE</option>

<option value= 9>SW</option>

<option value= 12>SE</option>

<option value= 0>NNW</option>

<option value= 6>NNE</option>

<option value= 8>SSW</option>

<option value= 10>SSE</option>

<option value= 2>WNW</option>

<option value= 5>WSW</option>

<option value= 14>ENE</option>

<option value= 13>ESE</option>

</select>

</div>

</div>

<div class="col-md-6 my-2">

<div class="md-form">
```

```
<label for="raintoday" class="raintoday" name="raintoday">Rain Today</label>
<select class="raintoday" id="raintoday" name="raintoday" aria-label="Rain Today">
<option selected>Did it Rain Today</option>
<option value= 1>Yes</option>
<option value= 0>No</option>
</select>
</div>
</div>
<div class="col-md-6 my-2 d-flex align-items-end justify-content-around">
<button type="submit" class="btn btn-info button" color= #ff0000 style="margin-left:
100%;">Predict</button>
</div>
</div>
</form>
</section>
<div>
<h1><center> {{ prediction }} </center></h1>
</div>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta2/dist/js/bootstrap.bundle.min.js" integrity="sha384-
b5kHyXgcpbZJO/tY9Ul7kGkf1S0CWuKcCD38l8YkeH8z8QjE0GmW1gYU5S9FOnJ0
" crossorigin="anonymous"></script>
```

```
</body>

</html>
```

#Rainy.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link

href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;400;500;600;

700;800;900&display=swap" rel="stylesheet">

    <link rel="stylesheet" href={{url_for('static',filename='style02.css')}}>

    <title>Rainy Day</title>

</head>

<body>


<h1 style="text-align: center; font-size: 3 rem; font-weight: bolder">Chances of

rain 🎧🌧🌧!</h1>

    <div class="rainyimg">
```

```
 <img src="../static/rainy.gif" style="height: 550px; width: 550px; margin-left:
32%">
    </div>


   </div>
</body>
</html>
```

#Sunny.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;400;500;600;
700;800;900&display=swap" rel="stylesheet">
    <link rel="stylesheet" href={{url_for('static',filename='style01.css')}}>


 <title>Sunny Day</title>
</head>
```

```
<body>

    <h1 style="text-align: center; font-size: 3 rem; font-weight: bolder"> No

chances of rain today, Enjoy your outing 😎!</h1>

    <div class="rainyimg">

        <img src="../static/sunny.gif" style="height: 550px; width: 550px; margin-

left: 28%">

    </div>

    <div>



   </div>

</body>

</html>
```

#Predictor.css

```
body {


    background-image: url('https://cdn.pixabay.com/animation/2023/03/26/01/15/01-15- 42-

612_512.gif');


    background-repeat: no-repeat;


 background-size: cover; /* Scale the image to cover the entire area */
 background-position: center;

font-family: 'Poppins', sans-serif;

}

.form {

    background-color: white; ;

    width: 70vw;

    margin: 50px auto;

    padding: 20px 50px;

    box-shadow: 0 5px 11px 0 rgba(0,0,0,0.18),0 4px 15px 0 rgba(0,0,0,0.15);

    border-radius: 12px;

}

.form h1 {

    color: #a3edfa;

}

.button {
  padding: 5px 30px;

    font-size: 18px;

  }
```

#Style01.css

```
  body {

    background-image:

  url('https://cdn.tourradar.com/s3/tour/1500x800/136950_64ad2b68cf6a0.jpg'

  );

    font-family: 'Poppins', sans-serif;

  }


  h2{

    font-size: 2 rem;

    font-weight: bold;

  }
```

#Style02.css

```
body {
    background-image:
url('https://media.istockphoto.com/id/1321878632/photo/cloudy-sky-over-
beautiful-flood-plain-
landscape.jpg?s=2048x2048&w=is&k=20&c=ayW8lRlZMaeQloY80kSiR
vCwsEjv0cyELwzDW5hqfaY=');
    font-family: 'Poppins', sans-serif;
}


h2{
    font-size: 2 rem;
    font-weight: bold;
}
```

10.2 GitHub & Project Demo Link

https://github.com/kanilkumar32/Rainfall-Prediction-using-Machine-Learning