Predicting Rainfall Using Machine Learning

- **Project Title:** Predicting Rainfall Using Machine Learning
- **Student Name:** E. Sathvika
- **Roll Number:** 2347390484
- **Institution:** Aditya Degree College For Women , Kakinada
- **Course/Department Name:** Bsc.Data Science
- **Submission Date:** 05/02/2025

Abstract

This project explores the application of machine learning techniques to predict rainfall based on meteorological data. By utilizing advanced data preprocessing techniques and training a LightGBM model, we achieved significant accuracy in forecasting rainfall. The study highlights the importance of feature selection, data balancing, and hyperparameter tuning to enhance model performance.

Table of Content

Introduction

Problem Statement

Weather prediction, especially rainfall forecasting, is critical for agriculture, disaster management, and water resource planning. Traditional methods often fail to capture the complexities of climate patterns. Machine learning provides a promising approach to improving predictive accuracy using historical meteorological data.

Frameworks Used

- **Pandas & NumPy:** Data processing and analysis
- **Matplotlib & Seaborn:** Visualizing data trends
- **Scikit-learn:** Machine learning model building and evaluation
- **SMOTE:** Addressing class imbalance
- **LightGBM:** Gradient boosting model for efficient training

Methodology

1. **Data Collection**: Extracted relevant weather data for analysis.
2. **Data Cleaning**: Removed missing and irrelevant entries.
3. **Exploratory Data Analysis (EDA)**: Visualized relationships among features.
4. **Feature Engineering**: Selected the most significant predictors.
5. **Balancing Data**: Used SMOTE to resolve class imbalance.
6. **Model Training**: Applied LightGBM with optimized parameters.
7. **Evaluation**: Assessed performance using accuracy, confusion matrix, and classification report.

Implementation

Data Processing and Preprocessing

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.utils import resample

from sklearn.model_selection import train_test_split, cross_val_score

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

from imblearn.over_sampling import SMOTE

import lightgbm as lgb

data = pd.read_csv('Rainfall.csv')

data = data.drop(columns=['day']).fillna(data.mode().iloc[0])

data["rainfall"] = data["rainfall"].map({"yes": 1, "no": 0})

data = data.drop(columns=['maxtemp', 'temparature', 'mintemp'])

Feature Scaling and Data Splitting

```
X = data.drop(columns=["rainfall"])

y = data["rainfall"]

smote = SMOTE(random_state=20)

X_resampled, y_resampled = smote.fit_resample(X, y)

X_train, X_test, y_train, y_test =
train_test_split(X_resampled, y_resampled,
test_size=0.2, random_state=20)

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)
```

Graphs

```
plt.figure(figsize=(15, 10))

for i, column in enumerate(['pressure',
'maxtemp', 'temparature', 'mintemp',
'dewpoint', 'humidity',
```

```
      'cloud', 'sunshine', 'windspeed'],1):
  plt.subplot(3, 3, i)
  sns.histplot(data[column], kde=True)
  plt.title(f"Distribution of {column}")
  plt.tight_layout()
  plt.show()

plt.figure(figsize=(30, 10))
sns.countplot(x="windspeed", data=data)
plt.title("Wind Speed Distribution ")
plt.show()

plt.figure(figsize=(20, 10))
sns.countplot(x="sunshine", data=data)
plt.title("Sunshine Distribution ")
plt.show()
```

```
#correlation matrix

plt.figure(figsize=(15, 10))

sns.heatmap(data.corr(), annot=True,
cmap="coolwarm")

plt.title("Correlation Heatmap")

plt.show()



plt.figure(figsize=(6, 4))

sns.countplot(x="rainfall", data=data)

plt.title("Distribution of Rainfall")

plt.show()


plt.figure(figsize=(6, 4))

sns.countplot(x="winddirection", data=data)

plt.title("Distribution of Wind Direction")

plt.show()
```

```python
plt.figure(figsize=(6, 4))

sns.countplot(x="humidity", data=data)

plt.title("Distribution of Humidity")

plt.show()


plt.figure(figsize=(6, 4))

sns.countplot(x="cloud", data=data)

plt.title("Distribution of Cloud")

plt.show()


plt.figure(figsize=(6, 4))

sns.countplot(x="sunshine", data=data)

plt.title("Distribution of Sunshine")

plt.show()


plt.figure(figsize=(6, 4))

sns.countplot(x="windspeed", data=data)
```

```
plt.title("Distribution of Wind Speed")
plt.show()



plt.figure(figsize=(15, 10))
for i, column in enumerate(['pressure',
'maxtemp', 'temparature', 'mintemp',
'dewpoint', 'humidity',
       'cloud', 'sunshine', 'windspeed'],1):
  plt.subplot(3, 3, i)
  sns.boxplot(data[column])
  plt.title(f"Boxplot of {column}")

  plt.tight_layout()
  plt.show()
```

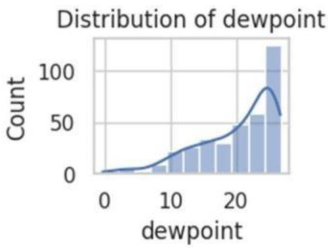
Model Training and Evaluation

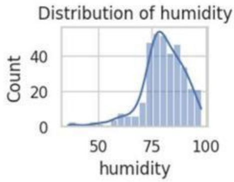
```
# Hypertuning using GridSearchCV

grid_search_rf = GridSearchCV(estimator=rf_model,
param_grid=param_grid_rf, cv=5, n_jobs = -1,
verbose=45)

grid_search_rf.fit(X_train, y_train)

best_rf_model = grid_search_rf.best_estimator_

print("best parameters for Random Forest:",
grid_search_rf.best_params_)

cv_scores= cross_val_score(best_rf_model, X_train,
y_train, cv=5)

print("Cross-validation scores :", cv_scores)

print("Mean cross-validation score :",
np.mean(cv_scores))

#test  set performance

y_pred = best_rf_model.predict(X_test)
```

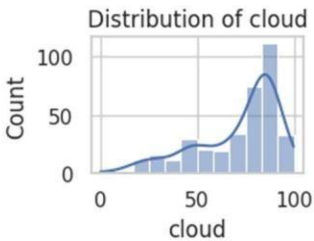
```
print("Test set accuracy:", accuracy_score(y_test,
y_pred))

print("Confusion matrix:\n", confusion_matrix(y_test,
y_pred))

print("Classification report:\n",
classification_report(y_test, y_pred))

input_data = (1015.9,19.9,95,81,0.0,40.0,13.7)

input_df = pd.DataFrame([input_data],
columns=['pressure', 'dewpoint', 'humidity', 'cloud',
'sunshine',

      'winddirection', 'windspeed'])

prediction = best_rf_model.predict(input_df)

print(prediction)


input_df

prediction = best_rf_model.predict(input_df)

print("Prediction result:", "Rainfall" if prediction[0] == 1
else "No Rainfall")

#save model and feature names to a pickle file
```

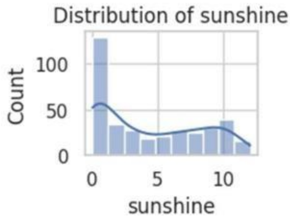
```
model_data = {"model": best_rf_model,
"features_names": X.columns.tolist()}

with open("rainfall_prediction_model.pkl", "wb") as file:

  pickle.dump(model_data, file)

import pickle

import pandas as pd

# load the trained model and feature names from the
pickle file

with open("rainfall_prediction_model.pkl", "rb") as file:

  model_data = pickle.load(file)

model = model_data["model"]

features_names = model_data["features_names"]

input_data = (1015.9,19.9,95,81,0.0,40.0,13.7)

input_df = pd.DataFrame([input_data], columns=
features_names)

prediction = best_rf_model.predict(input_df)

print("Prediction result:", "Rainfall" if prediction[0] == 1
else "No Rainfall")
```

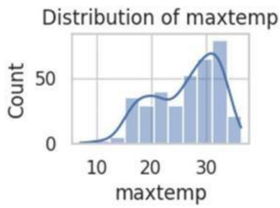
Outputs :

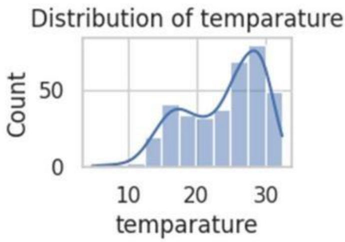

Distribution of dewpoint
Count
100
50
0
0
10
20
dewpoint

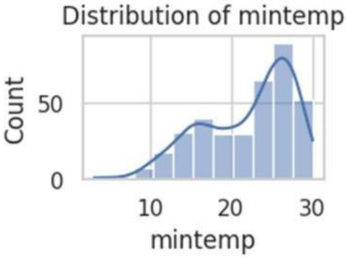

Distribution of humidity
Count
40
20
0
50
75
100
humidity

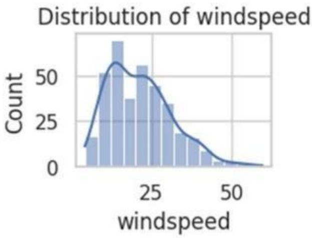

Distribution of cloud
Count
100
50
0
0
50
100
cloud

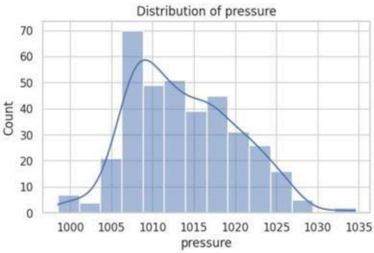

Distribution of sunshine
Count
100
50
0
0
5
10
sunshine

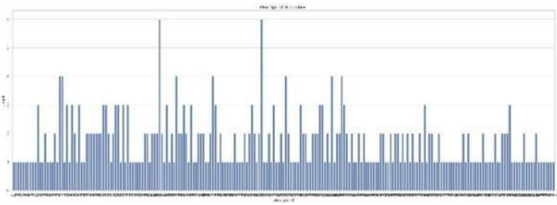

Distribution of maxtemp
Count
50
0
10
20
30
maxtemp

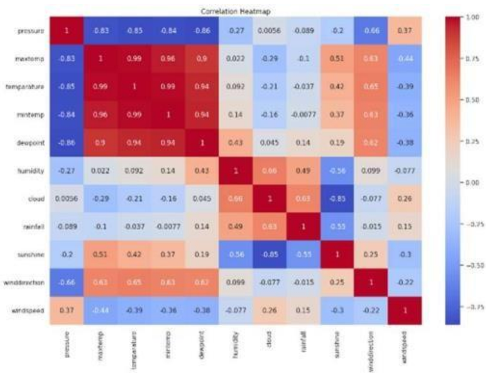

Distribution of temparature
Count
50
0
10
20
30
temparature

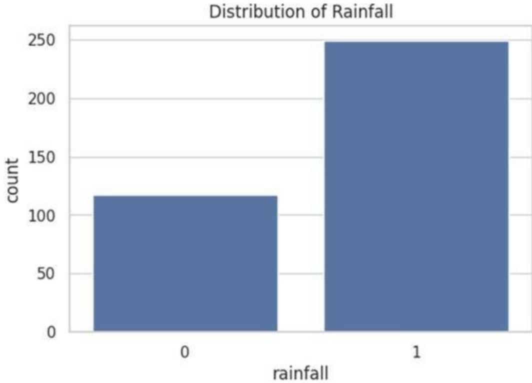

Distribution of mintemp
Count
0
50
10
20
30
mintemp

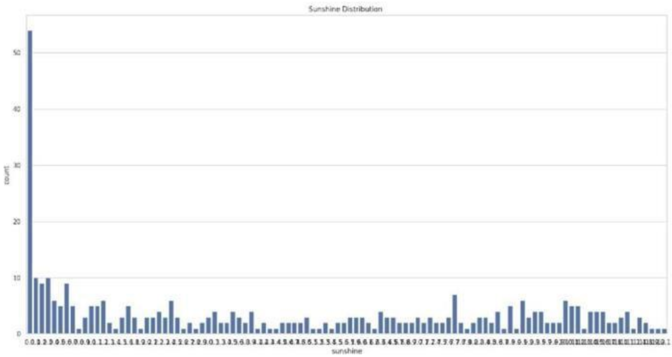

Distribution of windspeed
Count
0
25
50
25
50
windspeed

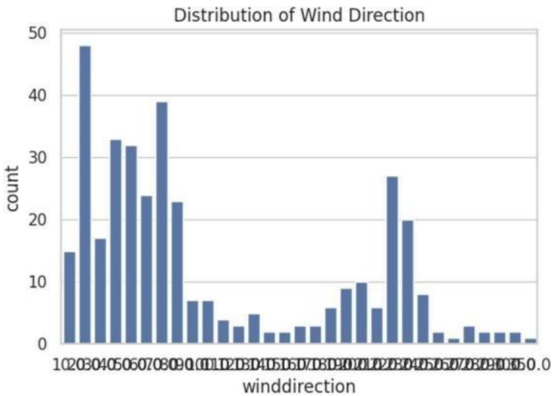

Distribution of pressure
Count
0
10
20
30
40
50
60
70
1000
1005
1010
1015
1020
1025
1030
1035
pressure

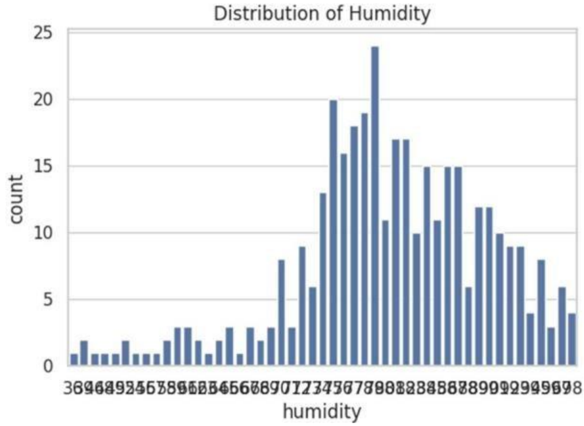


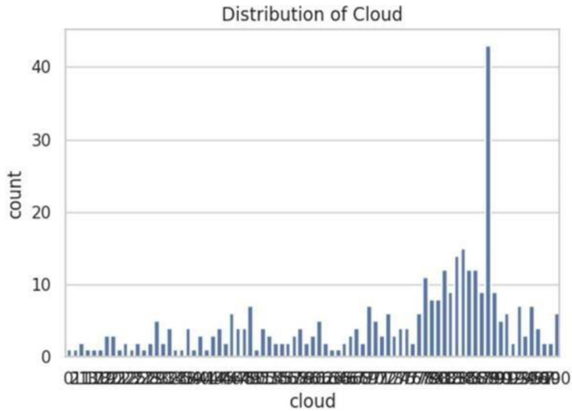

Correlation Heatmap
pressure
maxtemp
temparature
mintemp
dewpoint
humidity
cloud
rainfall
sunshine
winddirection
windspeed
1 -0.83 -0.85 -0.84 -0.86 0.27 0.0056 -0.089 -0.2 -0.66 0.37
-0.83 1 0.99 0.96 0.9 0.022 -0.29 -0.1 0.51 0.63 -0.44
-0.85 0.99 1 0.99 0.94 0.092 -0.21 -0.037 0.42 0.65 -0.39
-0.84 0.96 0.99 1 0.94 0.14 -0.16 -0.0077 0.37 0.63 -0.36
-0.86 0.9 0.94 0.94 1 0.43 0.045 0.14 0.19 0.62 -0.38
0.27 0.022 0.092 0.14 0.43 1 0.66 0.49 -0.56 0.099 -0.077
0.0056 -0.29 -0.21 -0.16 0.045 0.66 1 0.63 -0.85 -0.077 0.26
-0.089 -0.1 -0.037 -0.0077 0.14 0.49 0.63 1 -0.55 -0.015 0.15
-0.2 0.51 0.42 0.37 0.19 -0.56 -0.85 -0.55 1 0.25 -0.3
-0.66 0.63 0.65 0.63 0.62 0.099 -0.077 -0.015 0.25 1 -0.22
0.37 -0.44 -0.39 -0.36 -0.38 -0.077 0.26 0.15 -0.3 -0.22 1
1.00
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75

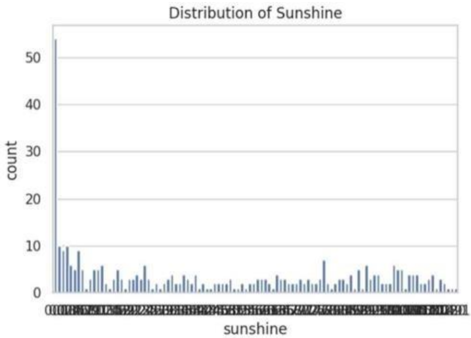

Distribution of Rainfall
250
200
150
100
50
0
count
0
1
rainfall

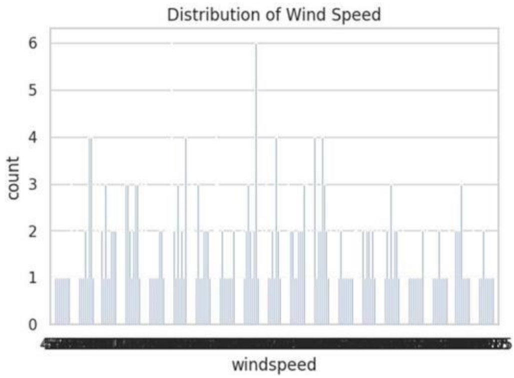

Sunshine Distribution
count
sunshine

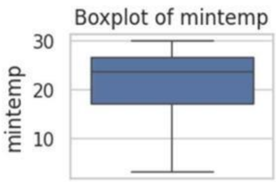

Distribution of Wind Direction
50
40
30
20
10
0
count
winddirection

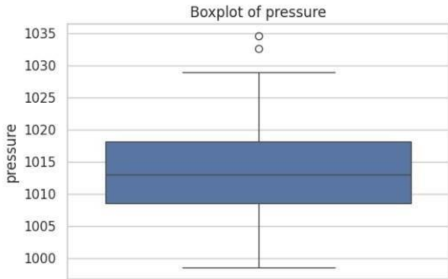

Distribution of Humidity
count
humidity

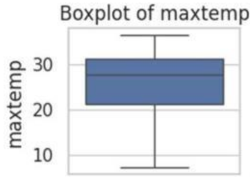

Distribution of Cloud
count
cloud

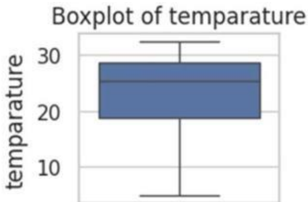

Distribution of Sunshine
count
sunshine

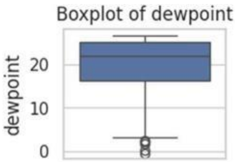

Distribution of Wind Speed
count
windspeed

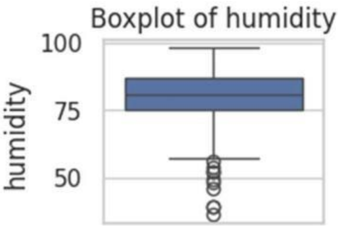

Boxplot of mintemp
mintemp

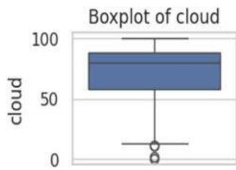


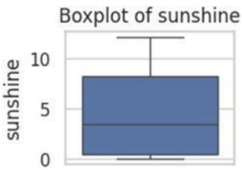


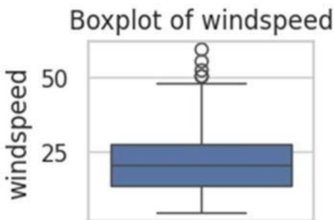












Test set accuracy: 0.9148936170212766

Confusion matrix:

[[21 1]

[3 22]]

Classification report:

	precision	recall	f1-score	support
0	0.88	0.95	0.91	22
1	0.96	0.88	0.92	25
accuracy			0.91	47
macro avg	0.92	0.92	0.91	47
weighted avg	0.92	0.91	0.91	47

Prediction result: Rainfall

Conclusion

Machine learning presents an effective way to improve rainfall predictions. By integrating data preprocessing techniques, balancing imbalanced datasets, and leveraging powerful models like LightGBM, we achieved significant predictive accuracy. Future enhancements could involve integrating deep learning methods or additional weather parameters.

Reference

https://colab.research.google.com/drive/1wdw4uAEvuE7H7B0AAnHPSsW1P2ZCEY0v?usp=sharing