Ministry of Science and Higher Education of the Republic of Kazakhstan
L.N. Gumilyov Eurasian National University

Faculty of Information Technology
Department of Information Systems

COURSEWORK

ON THE SUBJECT
"Mathematical Foundations of Intelligent Systems"
For third-year students of the specialty 6B06103 - Information Systems

Topic: **Using machine learning for rainfall prediction and flood risk forecasting.**

Completed by:

Student of group IS-31

Kalybek Aruzhan

Coursework Supervisor:

Prof., Zhukabayeva T. K.

Full name, signature

Members of the Commission:

Assoc. Prof. Muhanova A.A.

Full name, signature

PhD, Serikbayeva S.K.

Full name, signature

Full name, signature

Grade
«_____»____________2024

Astana 2024

Table of Contents

INTRODUCTION

Rainfall prediction is helpful to avoid flood which save lives and properties of humans. Moreover, it helps in managing resources of water. Information of rainfall in prior helps farmers to manage their crops better which result in growth of country's economy. Fluctuation in rainfall timing and its quantity makes rainfall prediction a challenging task for meteorological scientists. This project aims to address these challenges by utilizing machine learning techniques to predict rainfall patterns and assess flood risks, offering a data-driven solution to enhance decision-making processes.

Machine learning provides powerful tools for analyzing large volumes of meteorological data and identifying complex patterns that traditional statistical methods often fail to capture. By leveraging these capabilities, this project seeks to develop a robust predictive model that integrates diverse data sources, including rainfall measurements, geographical features, and historical flood records.

The project is built on mathematical foundations, employing concepts such as regression analysis, gradient descent optimization, and performance evaluation metrics like mean squared error and accuracy. These mathematical models ensure that the machine learning algorithms are both reliable and interpretable, forming the core of the predictive system.

The relevance of this work extends to both academic research and industry applications. As climate-related challenges become more pressing, the demand for innovative solutions in environmental management, urban planning, and disaster preparedness continues to grow. By addressing these needs, this project aligns with global efforts to apply artificial intelligence in climate resilience and sustainable development. Combining predictive analytics with user-friendly tools, the project contributes to advancing machine learning in high-impact, real-world applications.

The goal of this work is to develop a machine learning-based system that predicts rainfall and evaluates flood risks, facilitating informed decision-making and improving disaster preparedness.

The objectives of the course work are as follows:

1. To analyze and preprocess meteorological datasets, including rainfall measurements, geographical data, and historical flood records.

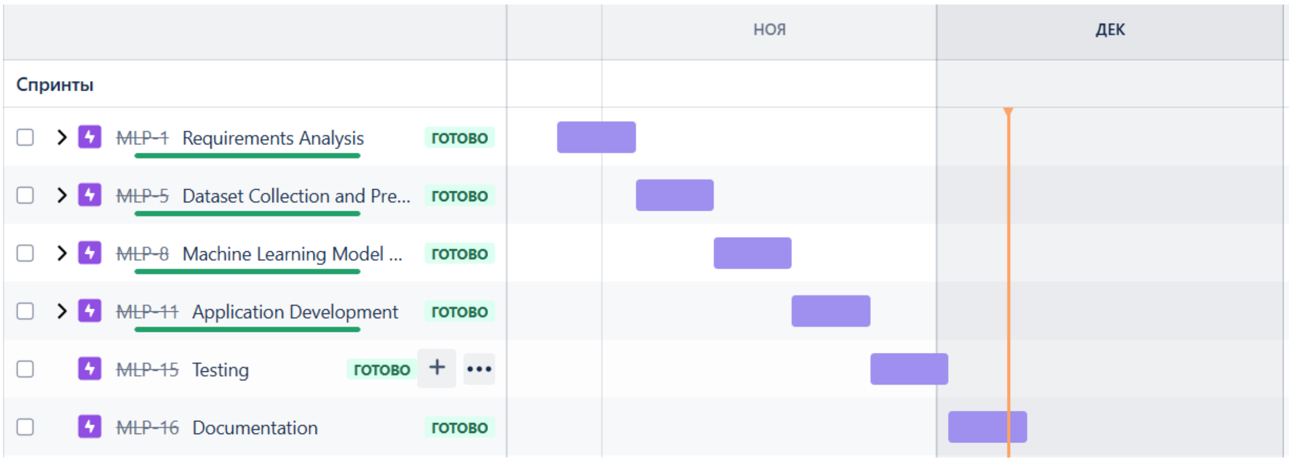
2. To implement machine learning algorithms for rainfall prediction and flood risk assessment.

3. To evaluate model performance using appropriate metrics such as accuracy and mean squared error.

4. To design and develop an SQL database for storing user credentials and relevant project data.

5. To create a Python-based user interface for seamless interaction with the predictive system.

The object of the course work is rainfall and flood prediction as an application of machine learning. The subject is the implementation of machine learning algorithms, database integration, and interface development to create a functional predictive system.



Спринты

☐ ~~MLP-1~~ Requirements Analysis ГОТОВО

☐ ~~MLP-5~~ Dataset Collection and Pre... ГОТОВО

☐ ~~MLP-8~~ Machine Learning Model ... ГОТОВО

☐ ~~MLP-11~~ Application Development ГОТОВО

☐ ~~MLP-15~~ Testing ГОТОВО

☐ ~~MLP-16~~ Documentation ГОТОВО

Figure 1. A plan of the work

A review of information sources used for this project includes academic journals, research papers, and textbooks focusing on machine learning algorithms, meteorological data analysis, and database development. Online resources and documentation for tools such as Python libraries, SQL integration, and interface design were also utilized. These sources provided foundational knowledge and technical guidance for implementing the system.

By leveraging machine learning, this project aims to analyze large volumes of meteorological data and identify complex patterns that traditional statistical methods often fail to capture. The system is built on mathematical foundations, employing concepts such as regression analysis, gradient descent optimization, and performance evaluation metrics like mean squared error and accuracy. These mathematical models ensure that the machine learning algorithms are both reliable and interpretable, forming the core of the predictive system.

The relevance of this work extends to both academic research and industry applications. As climate-related challenges become more pressing, the demand for innovative solutions in environmental management, urban planning, and disaster preparedness continues to grow. By addressing these needs, this project aligns with global efforts to apply artificial intelligence in climate resilience and sustainable development. Combining predictive analytics with user-friendly tools, the project contributes to advancing machine learning in high-impact, real-world applications.

LITERATURE REVIEW

Rainfall is one of the most influential meteorological factors, affecting various aspects of our lives, such as infrastructure damage during floods and disruptions in transport networks, as discussed by Le, Pham, Ly, Shirzadi, and Le in 2020. Floods, as a consequence of climate change, are expected to become more frequent and have devastating effects in the future, according to Yucel, Onen, Yilmaz, and Gochis in 2015. Recent studies also indicate that weather conditions can increase air pollution, which is linked to health problems like asthma, as shown by Mokrani, Zadtootaghaj, and others in 2019. Consequently, many studies have proposed rainfall forecasting methods to prepare for such events. For improving mobility and supporting agriculture and industrial growth, accurate predictions are essential, as noted by Salman, Xingjian, Aguasca-Colomo, and others in 2018, 2015, and 2019.

The process of developing rainfall prediction models using machine learning involves several key steps that ensure proper data preparation and analysis. (Fig. 1.1)

1. The process starts with the dataset containing historical rainfall data, including time-series information and related meteorological parameters.

2. At the pre-processing stage the data undergoes preparation, including normalization, cleaning, outlier removal, and other actions to improve data quality for analysis.

3. Missing values in the dataset are addressed using techniques such as mean imputation, interpolation, or removing rows with missing data.

4. At the feature reduction stage, principal component analysis is applied to minimize the number of features in the dataset while retaining the most critical information

5. The dataset is split into training (70%) and testing (30%) subsets. The training data is used to develop the model, while the testing data evaluates its accuracy and reliability.

6. The developed model employs machine learning algorithms to predict rainfall volumes based on input data.

7. In the final step, the model's performance is visualized through graphs, and statistical metrics are presented to assess prediction accuracy and algorithm efficiency.

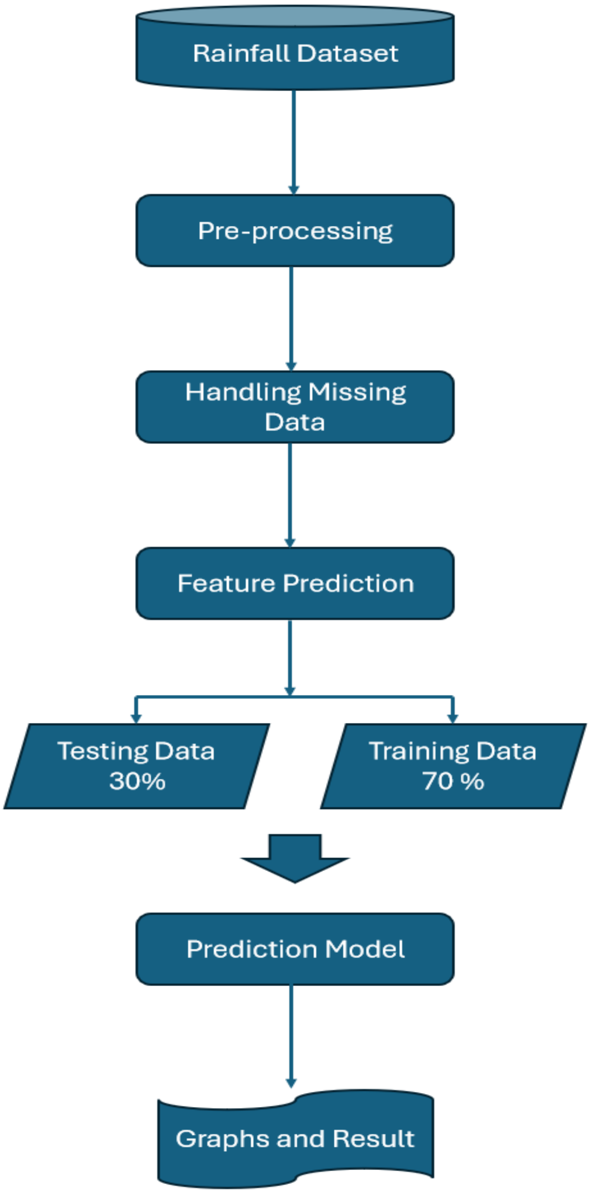


Figure 1.1 Stages of data processing

Traditional rainfall forecasting methods have used statistical techniques to assess correlations between rainfall, geographic coordinates, and atmospheric factors like pressure, temperature, wind speed, and humidity. However, due to the non-linearity of rainfall, accurate predictions are challenging, as pointed out by Wu and Chau in 2013. To address this, techniques such as Singular Spectrum Analysis and Wavelet Analysis have been explored by researchers like Gan, Xiang, and others in 2018. However, these mathematical models require significant computational resources and often yield limited results, as discussed by Singh and Borah in 2013.

In recent years, there has been growing interest in using Artificial Neural Networks for rainfall forecasting, as they can handle the non-linearity of the data and require minimal knowledge of variable relationships, according to Liu and colleagues in 2019. Among the most suitable types of neural networks for such tasks are Recurrent Neural Networks, which help address the spatial and temporal variability of rainfall data, as noted by Hossain, Rasel, Imteaz, and Mekanik in 2020. Specifically, LSTM networks have proven effective in multi-step rainfall forecasting, as demonstrated by Greff, Srivastava, Koutník, Steunebrink, and Schmidhuber in 2016 and Kratzert, Yunpeng, and others in 2017.

Some studies have also applied AutoML tools to identify the best algorithms and hyperparameters for rainfall forecasting, yielding positive results in experimental studies, as discussed by Hutter, Kotthoff, and Vanschoren in 2019 and Le, Fu, and Moore in 2020. The authors suggest that AutoML can effectively select the most accurate model for specific datasets, which is crucial for developing reliable rainfall prediction models.

Therefore, while the task of rainfall forecasting remains complex, the use of neural networks like LSTM and Bidirectional LSTM, along with AutoML tools for model selection, has shown promise in improving prediction accuracy, as indicated by Balluff, Bendfeld, and Krauter in 2020.

Different studies have used various methods for rainfall prediction. For example, Thirumalai et al. applied linear regression to predict rainfall based on agricultural crop seasons. This helps farmers plan which crops to harvest.

Geetha and Nasira used data mining techniques to predict weather phenomena like rainfall and fog. They applied decision trees and the RapidMiner tool, achieving 80.67% accuracy. They suggested that fuzzy logic could improve the results.

Parmar and colleagues reviewed different machine learning models for rainfall prediction, including neural networks and ARIMA models, discussing their advantages and limitations.

Dash et al. applied artificial neural networks and other algorithms for rainfall prediction using historical data. They found that the ELM algorithm provided the best results.

Singh and Kumar proposed a hybrid approach combining Random Forest and Gradient Boosting with other machine learning techniques like AdaBoost. The best results were achieved with Gradient Boosting and AdaBoost.

MACHINE LEARNING AND MODEL IMPLEMENTATION

2.1 Dataset description

The chosen dataset covers historical meteorological and hydrological data from the United States, providing a diverse variety of indicators useful for examining weather trends and monitoring environmental dangers. This dataset was obtained from Kaggle, a recognized site that offers high-quality, well-structured datasets. The dataset uses Kaggle to assure dependability, correctness, and simplicity of use, making it excellent for research and practical applications.

The dataset is a comprehensive collection of meteorological data obtained from 20 major cities across the United States between 2024 and 2025. It has a diverse collection of features, making it ideal for predictive modeling, weather trend research, and the creation of weather-related applications. With two years of daily weather observations, the dataset provides a solid basis for a variety of analytical and machine learning projects. The dataset provides numerous potential applications, including training and evaluating machine learning algorithms to predict the likelihood of rain, analyzing weather patterns to identify trends and correlations across different cities, exploring relationships between variables such as humidity, temperature, and precipitation, and applying advanced machine learning techniques to enhance forecast accuracy and improve prediction reliability.

Table 2.1

Dataset features

Feature	**Description**
Temperature	The average daily temperature, which is crucial for analyzing seasonal and regional climatic conditions.
Humidity	The measurement of moisture content in the air, vital for evaluating precipitation probabilities and atmospheric comfort levels.
Wind Speed	A parameter that influences weather formation and can indicate extreme conditions.

Feature	Description
Precipitation	The amount of rainfall recorded for a given day, a fundamental feature for analyzing rainfall patterns and intensities.
Cloud Cover	The extent of cloudiness, an important factor in forecasting sunlight exposure and predicting weather phenomena.
Pressure	A key indicator often used to detect weather changes, such as the likelihood of rain or storms.
Water Level In Rivers	A hydrological metric critical for assessing flood risks and monitoring water resource conditions.

The dataset includes two binary columns that provide valuable insights for predictive analysis – "Rain Tomorrow" and "Flood Risk " (1 for yes, 0 for no)

Table 2.2

Target variables

Variable	Description
Rain Tomorrow	A binary feature indicating whether rain is expected the following day
Flood Risk	A binary feature denoting the risk of flooding based on current conditions

This dataset is suitable for machine learning applications since it contains both numerical and categorical data, allowing for the implementation of a variety of analysis and prediction techniques. Classification models, such as logistic regression, decision trees, or neural networks, can be used efficiently to predict rainfall and assess flood hazards. Furthermore, the dataset's range of properties enables the development of comprehensive models that reflect the interactions between meteorological and hydrological parameters.

Table 2.3

Dataset

Date	**Location**	**Temperature**	**Humidity**	**Wind speed**	**Precipitation**	**Cloud Cover**	**Pressure**	**Water Level In Rivers**	**Rain tomorrow**	**Flood Risk**
01.01.2024	New York	87.5	75.6	28.3	0.0	69.6	1026.03	4,3	0	0
02.01.2024	New York	83.2	28.7	12.4	0.5	41.6	995.96	5,5	0	0
03.01.2024	New York	80.9	64.7	14.1	0.9	77.3	980.7	7,5	1	1
...	...	...	...	...	...	...	...	...	...	...
20.06.2025	Chicago	49.9	23.1	1.5	0.5	94.2	987.3	2,6	0	0

A correlation matrix is a table that shows the pairwise correlation coefficients of numerical variables in a dataset, allowing you to measure the strength and direction of linear interactions. The correlation coefficients range from -1 to 1, with 1 indicating a perfect positive correlation (as one variable increases, so does the other), 0 indicating no linear relationship, and -1 indicating a perfect negative correlation. A correlation matrix is a useful feature selection technique in machine learning applications because it identifies variables that are significantly linked with the target variable, hence increasing model performance. It also helps to discover multicollinearity by emphasizing variables that are highly linked with one another, which may decrease redundancy and improve model dependability, especially in algorithms like linear regression. It also gives insights into the dataset by displaying the correlations between the variables.

The colors indicate the strength and direction of correlations, ranging from -1 (perfect negative correlation) to 1 (perfect positive correlation), with lighter shades showing weaker correlations and deeper shades reflecting stronger correlations (Fig. 2.1).

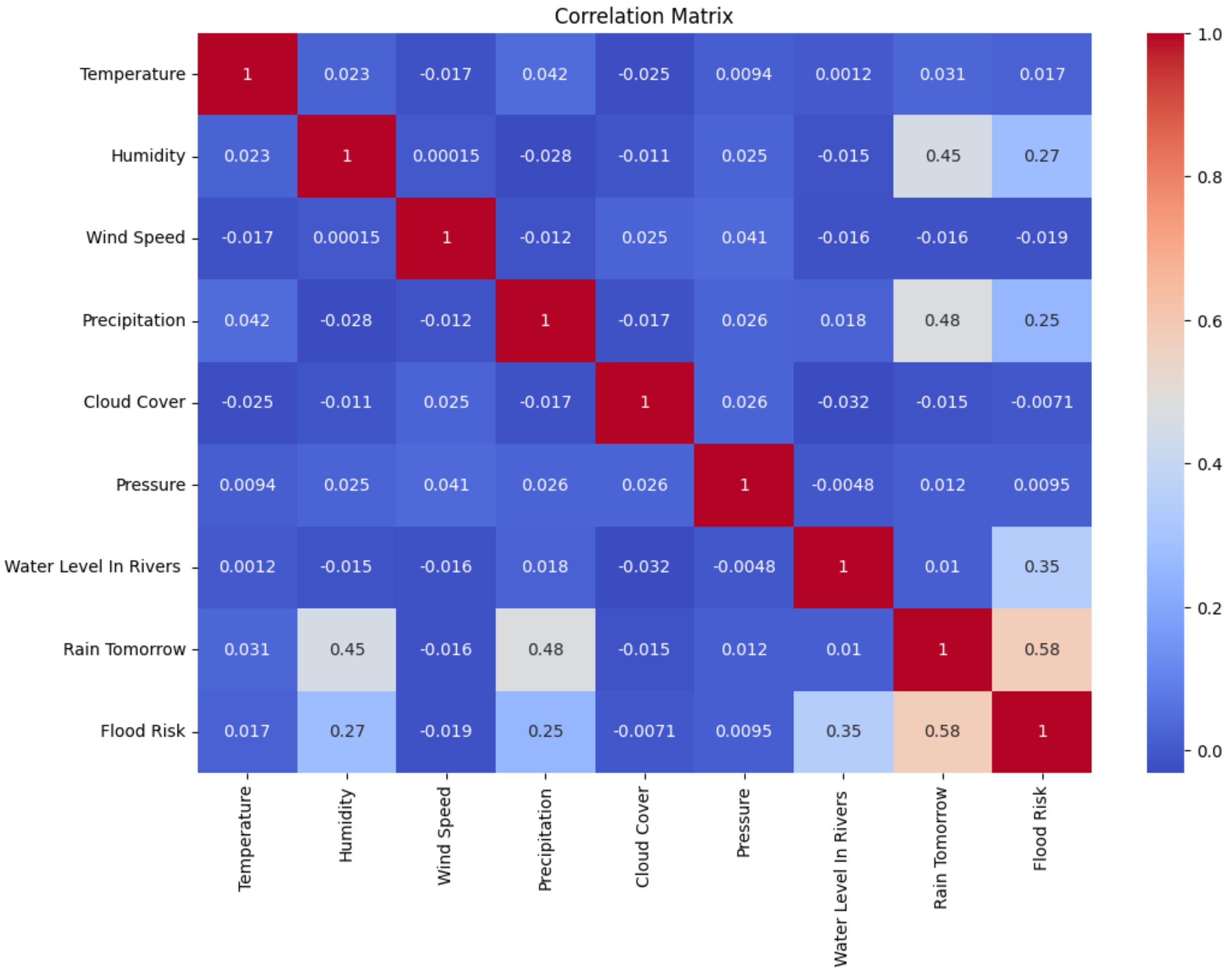


Figure 2.1 Correlation matrix

“Rain Tomorrow” is positively correlated with “Humidity” (0.45) and “Precipitation” (0.48), making these features significant predictors for rainfall. Flood Risk shows a strong positive correlation with “Rain Tomorrow” (0.58) and a moderate correlation with “Water Level in Rivers” (0.35), confirming the relevance of these variables in flood prediction. Most other features, such as “Temperature”, “Wind Speed”, and “Cloud Cover”, exhibit weak correlations with the target variables.

A pairplot is an effective visualization instrument that facilitates the examination of correlations among numerous variables within a dataset (Fig. 2.2). The pairplot compares each variable against every other variable, resulting in a grid of scatter plots for relationships and histograms for distributions along the diagonal. This style of map aids in identifying trends, relationships, and potential outliers in the data.

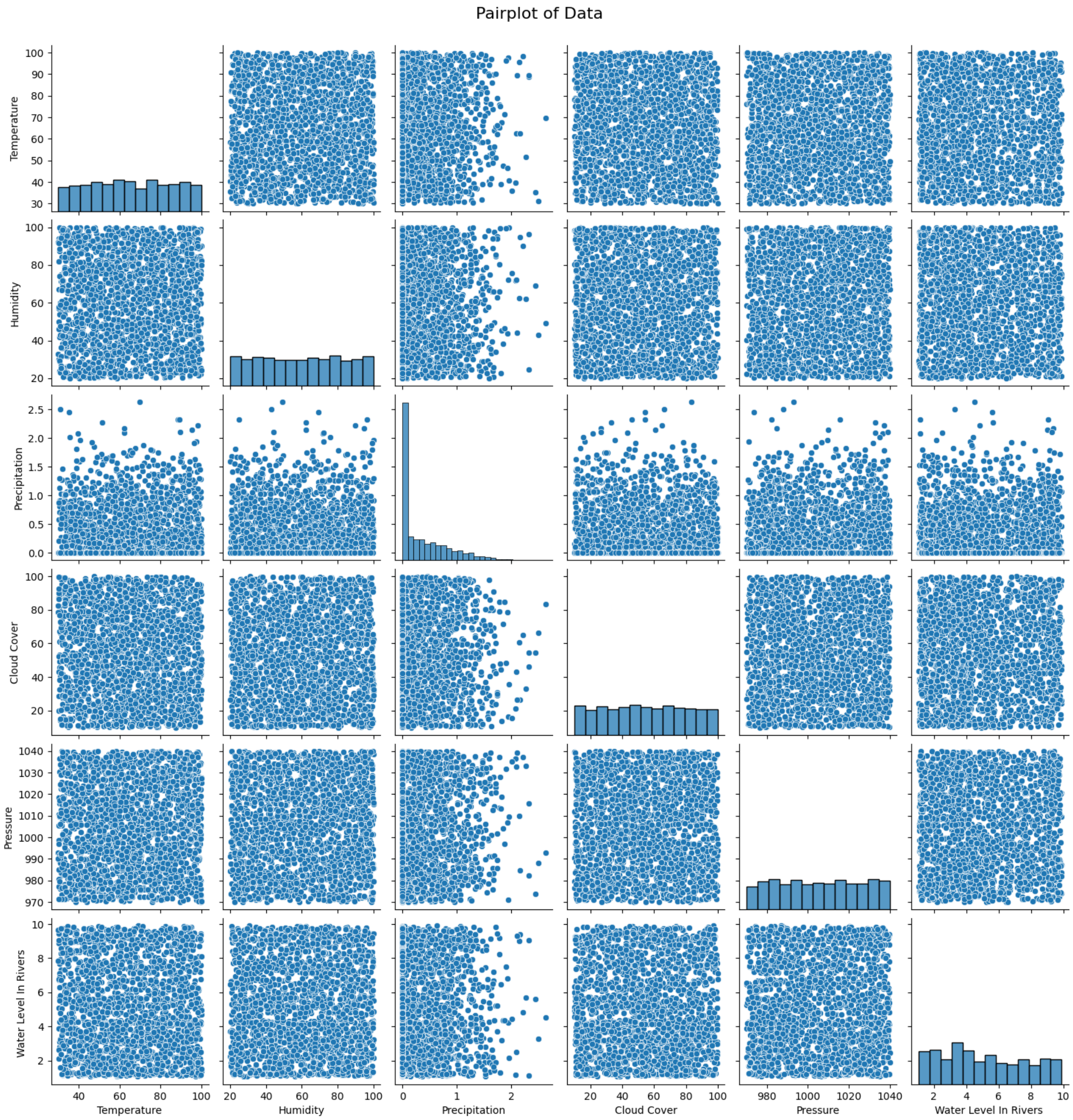


Figure 2.2 Pairplot of data

The distribution of variables represented on the diagonal histograms corresponds to the distribution of individual parameters. For example, “Precipitation” has an asymmetric distribution with a majority of low values, but “Temperature” and “Humidity” have a more uniform distribution, indicating no notable deviations. Because the dots are randomly distributed, the correlations between the variables displayed in the off-diagonal dot plots demonstrate only a modest association between “Temperature” and “Humidity”. Similarly, there are no discernible patterns in “Precipitation” and “Cloud Cover”, however their statistical link can be investigated further. The association between “Water level in rivers” and “Precipitation” appears to be modest, indicating a potential positive relationship that needs to be investigated further. Furthermore, certain variables, such as precipitation, have exceptionally high values, which might indicate outliers or essential situations for further investigation. Overall, the data is scattered randomly, showing the lack of recognizable groupings.

Table 2.4

Statistical metrics

Features	Mean	Median	Std Dev	Min	Max	IQR
Temperature	65.36	64.95	19.94	30.01	99.96	34.36
Humidity	59.98	60.23	23.33	20.06	99.97	39.90
Precipitation	0.38	0.19	0.47	0	2.63	0.65
Cloud Cover	54.4	53.48	25.74	10.03	99.86	44.07
Pressure	1005.53	1005.41	20.23	970	1039.98	35.12
Wind Speed	14.58	14.22	8.66	0.01	29.99	15.34
Water Level In Rivers	5.09	4.81	2.57	1.06	9.9	4.35

During the data analysis, basic statistical metrics for each variable were obtained, including mean, median, standard deviation, minimum and maximum values, and interquartile range. These measurements help you understand the distribution of data and detect potential outliers. Temperature, humidity, precipitation, clouds, pressure, wind speed, and river water level are all statistically significant factors. Temperature and humidity have a minimal standard deviation, indicating their stability. Simultaneously, precipitation and river water levels exhibit significant fluctuation, as seen by a large standard deviation and interquartile range. This might be owing to their inherent unpredictability.

2.2 Mathematical foundations and algorithms

Machine learning, a branch of artificial intelligence, uses mathematical concepts and formulae to build models that can learn and predict from data. Understanding the underlying mathematical ideas is critical for practitioners to properly use machine learning algorithms.

Linear Regression

Linear regression is a basic yet powerful technique for predicting a continuous variable based on one or more input features. The formula for simple linear regression can be expressed as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon$$

where y is the target variable, x_1, x_2, …, x_n are the input features, β_0, β_1, β_2, …, β_n are the coefficients to be learned, and ε represents the error term.

Linear regression, while a strong tool for predicting numerical values, is not appropriate for classification problems like rain and flood prediction, where the outcome is binary. The purpose of classification tasks such as rain or flood prediction is to assess the likelihood of an event occurring, and linear regression cannot accurately characterize such binary outcomes.

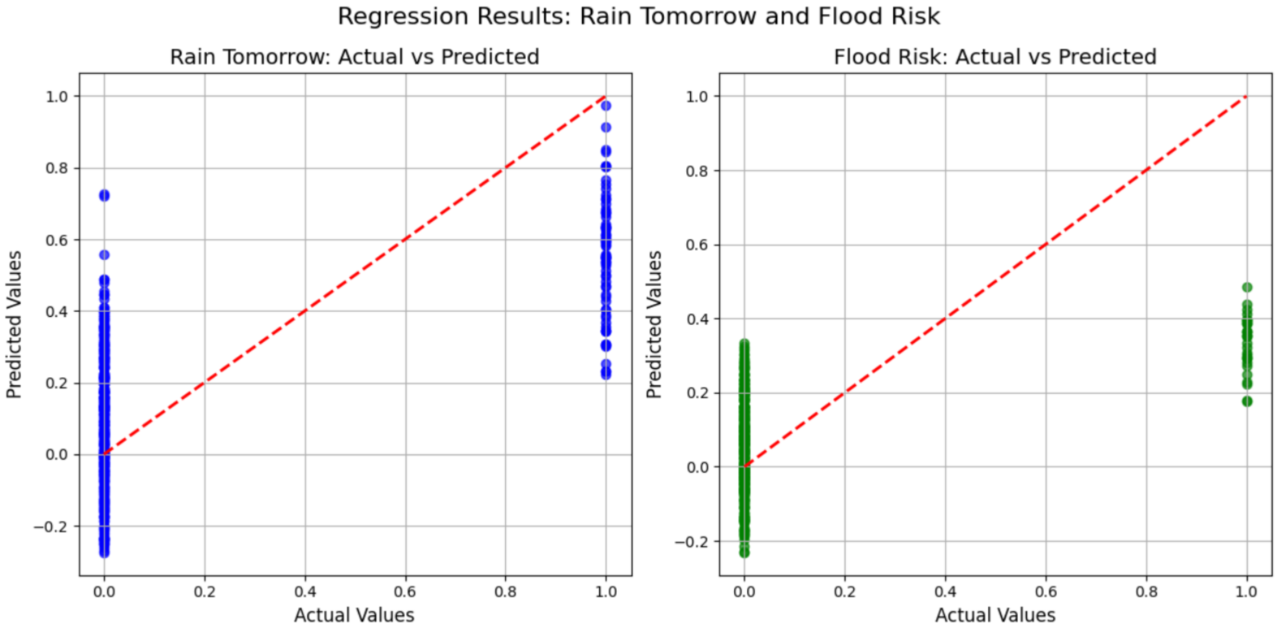


Figure 2.3 Linear regression results

Rain Tomorrow - Mean Squared Error: 0.07, R2 Score: 0.46

Flood Risk - Mean Squared Error: 0.05, R2 Score: 0.27

Linear regression predicts continuous values, making it unsuitable for classification jobs that need a probability. Linear regression can give values outside of this range, yielding illogical results. Linear regression predicts rain and flood risk with low R^2 scores. This suggests that the model does not explain much of the variation in the data and has low accuracy.

Logistic regression

Logistic regression is widely used for classification tasks. The logistic function, or sigmoid function, plays a pivotal role in this algorithm. It is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

where z represents a linear combination of input features and their corresponding coefficients. The logistic function maps the linear output to a value between 0 and 1, allowing us to interpret it as a probability.

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Logistic regression is well-suited for this project because the goal is to predict binary outcomes – whether it will rain tomorrow or whether flooding is likely. By analyzing key features such as Humidity, Precipitation, Cloud Cover, and Water Level in Rivers, the model calculates the probability of each event. These probabilities help make data-driven decisions about future weather and flood risk.

Logistic regression addresses the tasks of predicting Rain Tomorrow and assessing Flood Risk by modeling the probability of these binary outcomes. The target variable Rain Tomorrow is binary, representing whether it will rain (1) or not (0). Logistic regression calculates the probability of rain based on features such as temperature, humidity, cloud cover, precipitation, and pressure, learning the relationship between these variables and the occurrence of rain. For a new data point, the model predicts the probability of rain tomorrow, and if this probability exceeds a certain threshold (e.g., 0.5), the prediction is "Yes"; otherwise, it is "No."

Similarly, logistic regression assesses Flood Risk, another binary target variable, by analyzing features such as water level in rivers, rainfall, cloud cover, and wind speed. It determines the likelihood of flooding by using a sigmoid function to model the probability of flood risk. If the predicted probability exceeds the defined threshold, the model classifies it as high flood risk (1); otherwise, it predicts low flood risk (0).

Cost functions quantify the error or discrepancy between the predicted values and the actual values. Mean Squared Error (MSE) is a commonly used cost function for regression problems, defined as:

$$MSE = \left(\frac{1}{N}\right) * \sum(y_i - \bar{y})$$

where N is the number of samples, y_i is the actual value, and ȳ is the predicted value. For classification problems, Cross-Entropy Loss is often employed, given by:

$$CE = -\sum(y_i * log(p_i) + (1 — y_i) * log(1 — p_i))$$

where y_i is the actual label (0 or 1) and p_i is the predicted probability.

Gradient descent is an optimization algorithm used to minimize the cost function and find the optimal values of the coefficients. The update rule for gradient descent can be expressed as:

$$\theta = \theta — \alpha * \nabla J(\theta)$$

where θ represents the coefficients, α is the learning rate, J(θ) is the cost function, and ∇J(θ) is the gradient of the cost function with respect to θ.

Matrix operations are fundamental in many machine learning algorithms. Some common operations include matrix multiplication, transpose, and inverse. Matrix multiplication is defined as:

$$C = A * B$$

where A and B are matrices, and C is the resulting matrix. The transpose of a matrix A is denoted as A^T, and the inverse of a matrix A is denoted as $A^{(-1)}$.

2.3 Model training and testing process

In this chapter, the focus is on the systematic process of training and testing logistic regression models to predict two critical outcomes: Rain Tomorrow and Flood Risk. The training phase involves preparing the dataset, applying preprocessing techniques, and splitting the data into training and testing subsets to ensure reliable model evaluation. By identifying patterns and relationships within the data, the models aim to deliver accurate and meaningful predictions. Additionally, the testing phase evaluates the performance and robustness of the models using various metrics, providing insights into their effectiveness and potential limitations.

```
from sklearn.model_selection import train_test_split
X = data[['Temperature', 'Humidity', 'Precipitation', 'Cloud Cover', 'Pressure',
'Wind Speed', 'Water Level In Rivers ']]
y_rain = data['Rain Tomorrow']
y_flood = data['Flood Risk']
X_train, X_test, y_train_rain, y_test_rain = train_test_split(X, y_rain,
test_size=0.2, random_state=42)
X_train, X_test, y_train_flood, y_test_flood = train_test_split(X, y_flood,
test_size=0.2, random_state=42)
```

The code snippet provided splits the dataset into training and testing sets for both rain prediction and flood prediction. First, it defines the features which include variables like temperature, humidity, precipitation, cloud cover, pressure, wind speed, and water levels in rivers. These features are used to predict two target variables: Rain Tomorrow (y_rain) and Flood Risk (y_flood).

The dataset is then divided using the train_test_split function from sklearn.model_selection. The function splits the data into training and testing sets, with 80% of the data used for training and the remaining 20% for testing. The random_state = 42 argument ensures that the split is reproducible, meaning the same data is used for training and testing each time the code is run.

The result of this process is the creation of training and testing sets for both features (X_train, X_test) and target variables (y_train_rain, y_test_rain for rain prediction and y_train_flood, y_test_flood for flood prediction). This approach allows the model to be trained on one subset of the data and evaluated on another, helping to prevent overfitting and providing a more accurate assessment of the model's performance.

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
import seaborn as sns
import matplotlib.pyplot as plt
```

- *accuracy_score* is used to evaluate how well the model performs by calculating the proportion of correctly classified samples.
- *confusion_matrix* provides a matrix showing the number of correct and incorrect predictions categorized by their actual and predicted labels. It helps evaluate the performance of the classification model in more detail.
- *classification_report* generates a detailed report that includes precision, recall, f1-score, and support for each class, giving a more comprehensive view of the model's performance.
- *seaborn* is a data visualization library based on Matplotlib that is used to make statistical graphics in Python. It provides high-level functions for drawing attractive and informative statistical plots, such as heatmaps for visualizing confusion matrices.
- *matplotlib.pyplot* is a plotting library used for creating static, animated, and interactive visualizations. The pyplot submodule is commonly used to generate basic plots such as line plots, bar charts, and histograms.

```
model_rain = LogisticRegression()          #Creating a logistic regression model
model_rain.fit(X_train, y_train_rain)      # Training the model
y_pred_rain=model_rain.predict(X_test)     #Making predictions
```

The code creates and trains a logistic regression model to predict whether it will rain tomorrow. First, a logistic regression model is initialized with LogisticRegression(). Then, the model is trained using the fit() method on the training data (X_train, y_train_rain). Finally, predictions are made on the test data using the predict() method, and the predicted values are stored in y_pred_rain.

```
accuracy_rain = accuracy_score(y_test_rain, y_pred_rain)
conf_matrix_rain = confusion_matrix(y_test_rain, y_pred_rain)
class_report_rain = classification_report(y_test_rain, y_pred_rain)
```

In machine learning, assessing a model's performance is essential to making sure it performs as planned and generates accurate predictions. A variety of measures, including accuracy, confusion matrix, and thorough classification reports, are employed to obtain a thorough grasp of the model's efficacy. These measures not only aid in evaluating the model's overall performance but also offer more in-depth understanding of particular facets of it, such its capacity to manage imbalances and accurately anticipate various classes.

2.4 Model result

Logistic regression produces probabilities that are limited between 0 and 1, which makes it ideal for binary outcomes like forecasting the likelihood of rain or flooding, in contrast to linear regression, which produces continuous values.

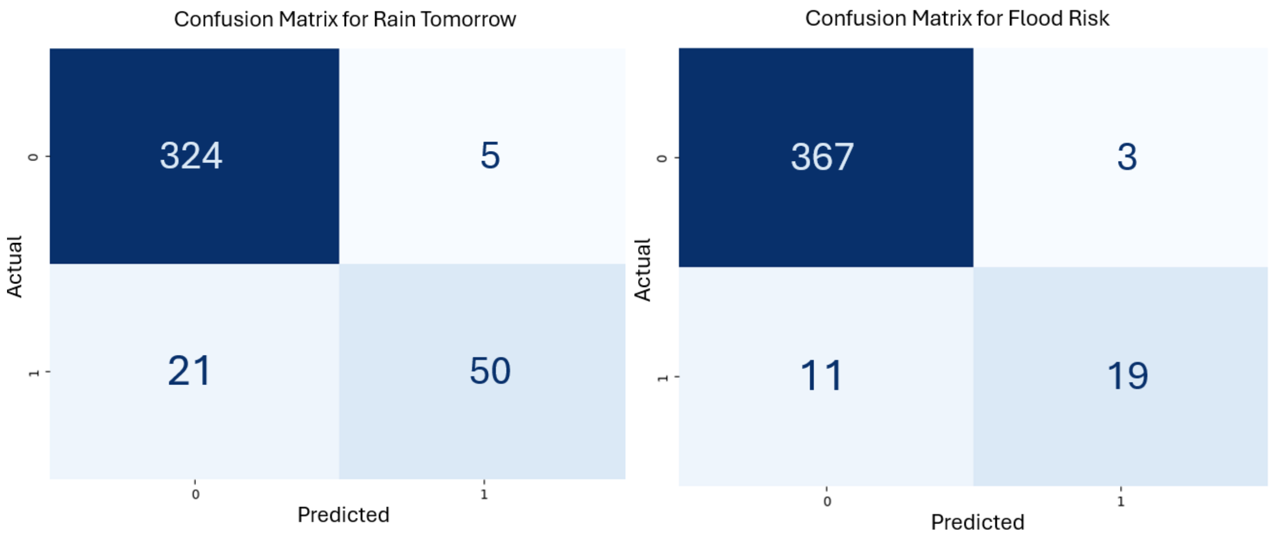


Figure 2.4 Confusion matrix

```
Accuracy for Rain Tomorrow: 0.935
Accuracy for Flood Risk: 0.965
```

Rain Tomorrow results:

- 50 cases where the model correctly predicted rain (True Positives).
- 324 cases where the model correctly predicted no rain (True Negatives).
- 5 cases where the model incorrectly predicted rain when there was no rain (False Positives).
- 21 cases where the model failed to predict rain when it occurred (False Negatives).

Flood Risk:

- 19 cases where the model correctly predicted flood risk (True Positives).
- 367 cases where the model correctly predicted no flood risk (True Negatives).
- 3 cases where the model incorrectly predicted flood risk when there was none (False Positives).
- 11 cases where the model failed to predict flood risk when it occurred (False Negatives).

Higher accuracy and fewer misclassifications show that the logistic regression model outperforms rain tomorrow in forecasting flood danger. The model's dependability is demonstrated by the low rates of false positives and false negatives seen in both confusion matrices. Depending on the application, changes to the decision threshold might increase sensitivity or specificity, particularly for crucial forecasts like flood risk, where false negatives could have dire repercussions.

The ROC (Receiver Operating Characteristic Curve) curve is a graph that shows how well a classification model distinguishes between two classes at various thresholds. It lets you visualize the model’s performance at all possible thresholds, not just a single one (Fig. 2.5).

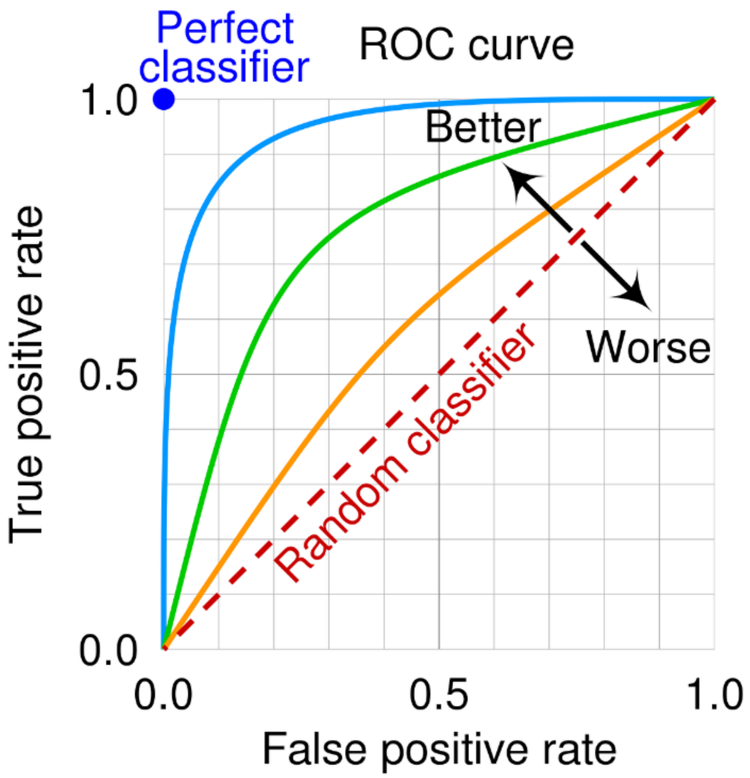


Figure 2.5 Receiver Operating Characteristic Curve

Top-Left Corner: If the curve is closer to the top-left corner, that's great. (Maximum True Positives, minimum false positives)

Above the Diagonal Line: If the curve is above the diagonal line (from bottom-left to top-right), our model is better than just random guessing because the diagonal line means where TPR = FPR.

Area Under Curve: The bigger the area under the curve (closer to 1), the better our model is.

The orange line represents the performance of the model at various probability thresholds (Fig. 2.6, 2.7). The curve demonstrates how effectively the model distinguishes between two classes: days when it will rain and days when it is not expected to rain. The graph shows that the model has high accuracy, since the curve is close to the upper left corner. This indicates the good ability of the model to simultaneously provide high sensitivity and specificity. The area under the curve is 0.96, which is close to 1, which indicates that the model separates positive and negative classes very well. At the bottom of the graph there is a diagonal dotted line corresponding to a random model. A significant upward deviation of the curve from this line confirms that the constructed model is significantly superior to random guessing.

Thus, the ROC curve and the AUC value confirm that the model effectively copes with the task of forecasting rain, minimizing the probability of errors of both the first and second kind.

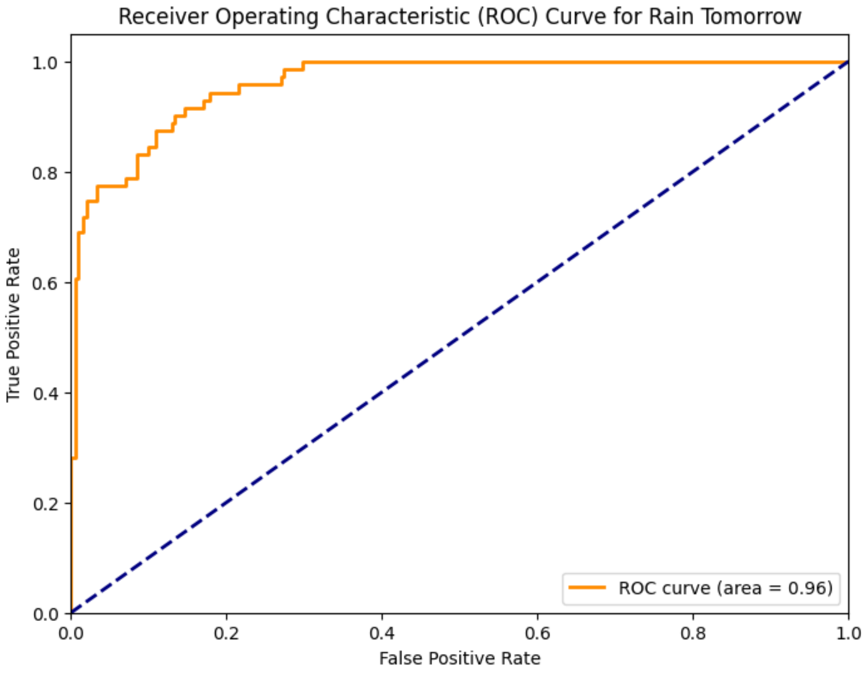


Figure 2.6 ROC for Rain Tomorrow

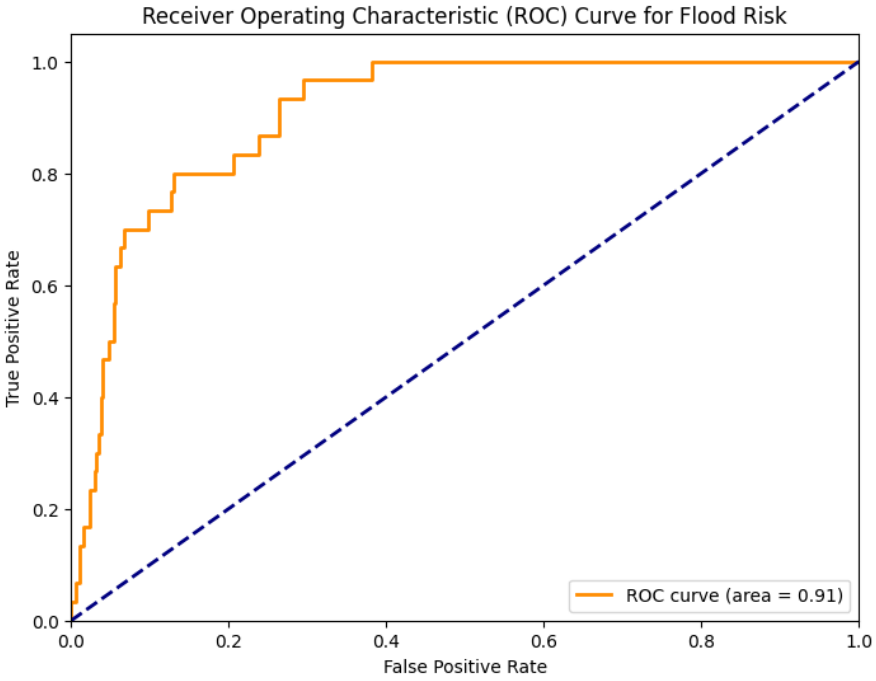
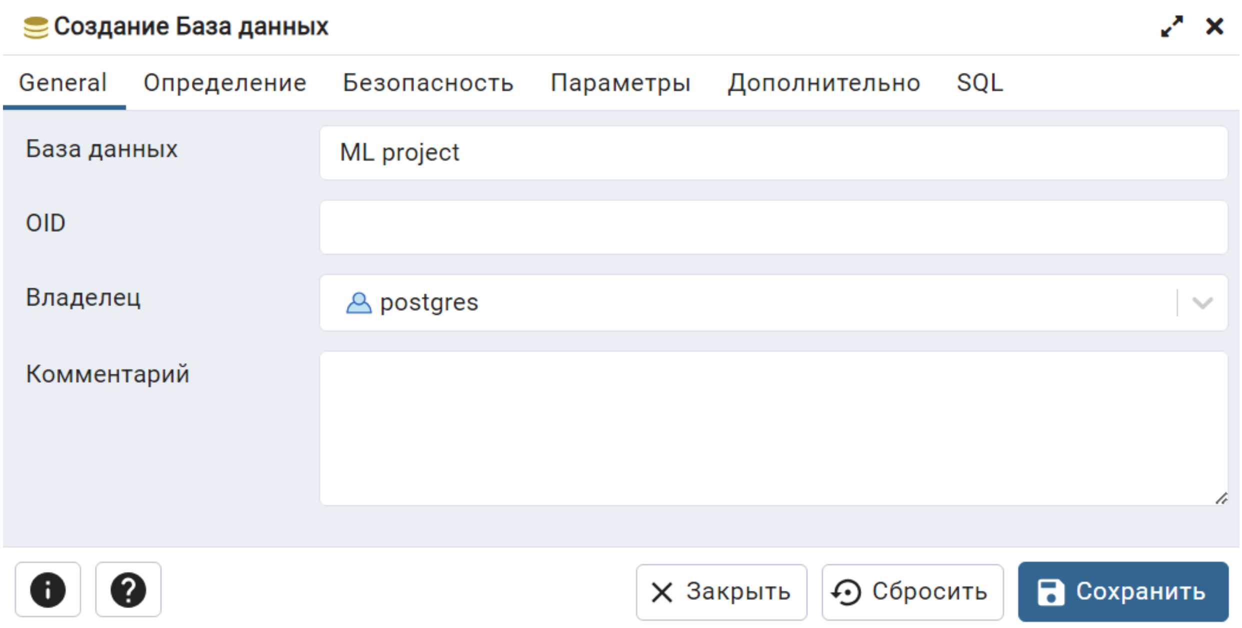


Figure 2.7 ROC for Flood Risk

APPLICATION DEVELOPMENT

3.1 Database

Creating a database in PostgreSQL begins with using the pgAdmin GUI, which allows you to interact with the database server. To do this, the user connects to the server using the administrator credentials. In the process of creating a database, a suitable server is selected, and then the procedure for creating it is initiated through the appropriate interface functionality. At the configuration stage, the database name is set, which is entered in a special field. In addition, if necessary, the owner of the database is indicated, which allows you to determine who will manage this database. Parameters such as encoding and access rights can also be configured to ensure the correct operation and security of the system.



База данных: ML project

OID:

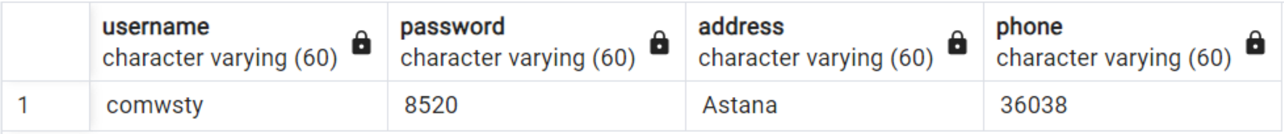
Владелец: postgres

Комментарий:

Figure 3.1. "Database Creation" Window

An SQL query was employed to establish the structure and key parameters of the table in the database.

```
create table users (
        username text not null,
        password text not null,
        address text not null
        phone text not null
        );
```



	username character varying (60) 🔒	password character varying (60) 🔒	address character varying (60) 🔒	phone character varying (60) 🔒
1	comwsty	8520	Astana	36038

Figure 3.2 Table “users”

The users table is used to store the data of users who will interact with the system. The users table is used for safe and efficient storage and management of user data. Each user has unique data that allows them to safely interact with the system, and the presence of these fields helps to manage accounts, ensure security, and establish communication with users, if necessary for the operation of the application.

```
db_name = 'postgres'
db_user = 'postgres'
db_password = '5658'
db_host = 'localhost'
db_port = '5432'
```

This data is used to establish a connection between the application and the PostgreSQL database. When the application is operating, it must interface with the database to save, retrieve, and process data.These arguments are supplied to a library, such as psycopg2, which then connects the program to the database. Following a successful connection, you may perform requests such as storing, retrieving, updating, and removing records.

3.2 User Interface development

The application is intended to anticipate precipitation and flood threats based on meteorological data. At launch, the user inputs the city and date, and the program gets the necessary meteorological data from the dataset and filters it based on the criteria entered. Next, using trained machine learning models, the program forecasts if it will rain and whether there is a risk of flooding in the selected city on the stated day. The forecast is shown as a series of meteorological characteristics, including temperature, humidity, wind speed, and others. Finally, the program shows information regarding probable precipitation and flood danger, telling the user about the present meteorological conditions and hazards.

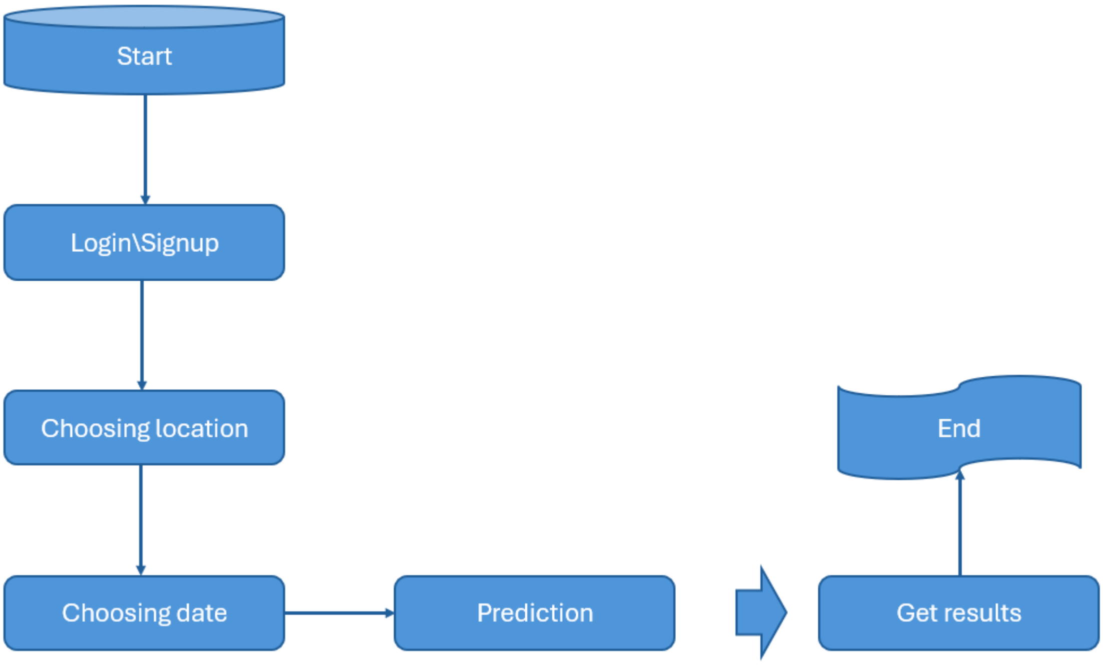


Figure 3.2 The flowchart of the application

The programm imports the necessary libraries for creating a graphical user interface using Tkinter, interacting with a PostgreSQL database, and working with data and machine learning.

Tkinter is a library for creating graphical user interfaces in Python. This library is used to create the main window, Label for text labels, and other components like messagebox to display messages.

psycopg2 is a library for interacting with PostgreSQL. It allows application to connect to a database, retrieve, and update data.

pandas is a library for data manipulation and analysis. It’s used to load and manipulate data, such as reading from Excel or CSV files and preparing the data for model training.

sklearn (scikit-learn) is a machine learning library. In the case, it’s used:

✓ *LogisticRegression* is an algorithm for classification tasks (e.g., predicting rainfall or flood risks).

✓ *train_test_split* is a function for splitting the data into training and testing sets.

✓ *StandardScaler* is a tool for data standardization (normalization), which is important for efficient model training.

```
root = Tk()
root.title('Login')
root.geometry('500x500+500+150')
root.configure(bg="#F8F8FF")
root.resizable(False, False)
```

This is an example of creating an interface with the Tkinter library. The application's primary GUI window is created using Python code and the Tkinter framework. The "Login" window is 500x500 pixels in size and has a fixed location on the screen (500 pixels horizontally and 150 pixels vertically offset). The window's backdrop is painted in a bright color using the code #F8F8FF. Resizable settings (False, False) prevent the user from resizing the window, hence preserving its design and functioning.

```
username = user.get()
password = code.get()
confirm_password = conform_code.get()
if password == confirm_password:
    window.destroy()
    additional_info_window(username, password)
else:
    messagebox.showerror('Error', 'Passwords do not match')
```

This code is designed to check the information given by the user during registration. After receiving the username, password, and confirmation, the application compares the entered passwords. If the passwords match, the current window closes and a new one appears to input more information. If the passwords do not match, an error notice is provided to the user, allowing them to try again. This strategy helps you avoid mistakes while creating an account.

Functions and details of the main part of the code:

1. *prepare_data(dataset)* prepares the dataset by selecting specific weather-related columns, scaling the data, and splitting it into training and test sets for both rain and flood predictions.

2. *train_models(X_train, y_rain_train, y_flood_train)* trains two logistic regression models: one for predicting rain and another for flood risk, using the training data.

3. *load_new_data() loads a new dataset (without target labels)* from an Excel file to make predictions on unseen data.

4. *fahrenheit_to_celsius(fahrenheit)* converts a temperature value from Fahrenheit to Celsius.

5. *predict_weather_and_flood(city, date, rain_model, flood_model, scaler, new_dataset)* filters data for a specified city and date, scales the features, and uses the trained models to predict whether it will rain and the risk of flooding. Displays an error if no matching data is found.

6. *open_page1_window()* defines a function that opens the main window for rainfall and flood risk prediction. This function initializes the models (rain_model, flood_model, scaler) using a dataset and prepares them for making predictions.

7. *search()* is inner function handles the user's search request. It gets the city and date entered by the user, loads a new dataset, and uses the models to predict rainfall and flood risk for that city on the given date. It displays the relevant weather details such as temperature, humidity, wind speed, etc., and shows predictions for rain and flood risk.

8. *City and date inputs*. The user selects a city from a dropdown menu and enters a date. The cities are predefined in a list. The city_var stores the selected city, and the date_entry stores the entered date.

9. *The weather details* are displayed using Entry widgets, which are updated dynamically based on the predictions.

10. *GUI Layout.* The window has a title "Rainfall Prediction and Flood Risk Forecast". It uses Label, OptionMenu, Entry, and Button widgets to create the interface. The layout is organized with padding to ensure a clean and user-friendly interface.

11. The dataset is loaded using *pd.read_excel()*, and the models are trained using logistic regression. When the user searches, it loads the data for the specified city and date, scales it, and predicts the outcomes using the models.

RESULTS AND DISCUSSUION

This chapter presents the important findings from the data analysis, with an emphasis on the correlations between meteorological variables such as temperature, humidity, precipitation, and others. This part evaluates the findings, highlighting key patterns and identifying any outliers or unexpected outcomes. It also explores the consequences for weather and flood risk prediction, as well as limits and recommendations for further study. The objective is to convey a clear grasp of the data's insights and practical applications.

The graphic on the Figure 4.1 depicts the value of each factor in forecasting rain tomorrow using logistic regression. The most significant element in the model is “Precipitation”, which is understandable given that the presence of rainfall directly affects the chance of rain the next day. The “Temperature” element is also essential, however it has a lower influence than precipitation. “Humidity’ and “Cloud Cover” have some impact, although their function is less important. The coefficients for the parameters “Pressure” and “Wind Speed” are low, showing a limited link with rainfall prediction. “Water Level in Rivers” has the least coefficient, indicating that it has little influence on predicting tomorrow's rainfall.

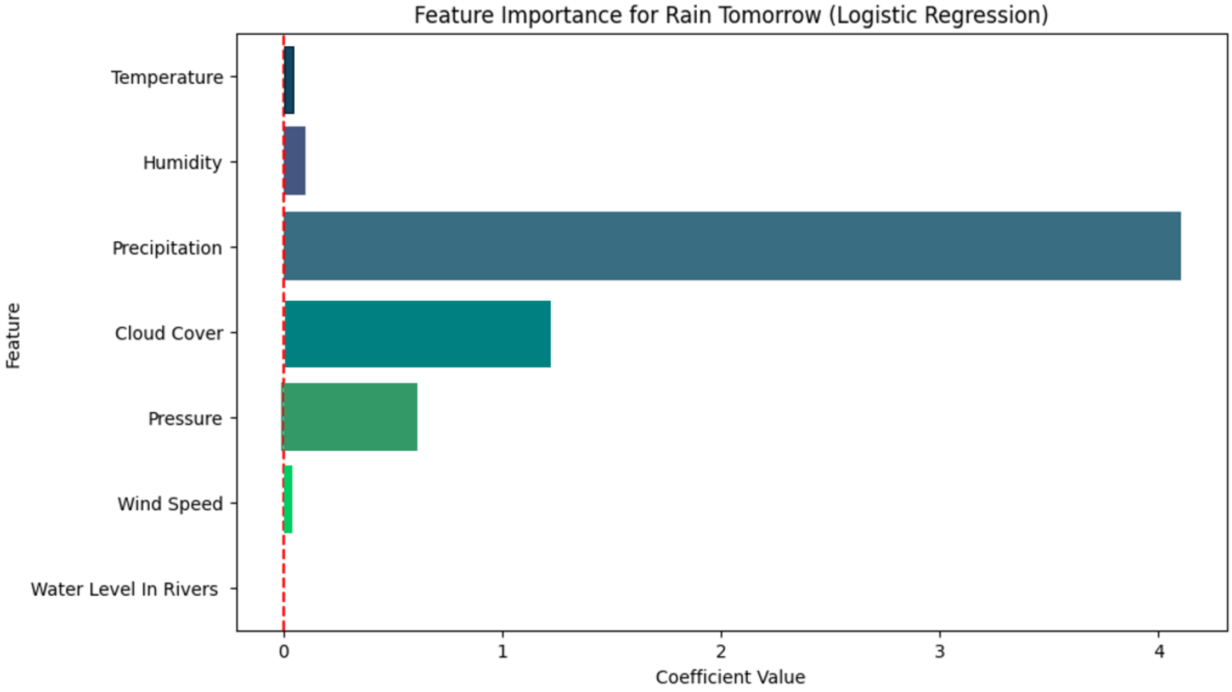


Figure 4.1 Feature importance for rain prediction

The graph in Figure 4.2 depicts the relevance of features in forecasting flood danger. Again, “precipitation” is the most relevant factor, which is consistent with the reality that excessive rainfall can raise river levels, increasing the danger of floods. Another important component is “Water Level in Rivers,” which directly influences the chance of flooding. “Temperature” and “Humidity” have a part in flood risk assessment, however they contribute less than precipitation and river water levels. “Cloud Cover” and "Pressure" have little significance in predicting flood danger. “Wind Speed” provides the least contribution to the model, indicating that it has a little influence in flood forecasting.

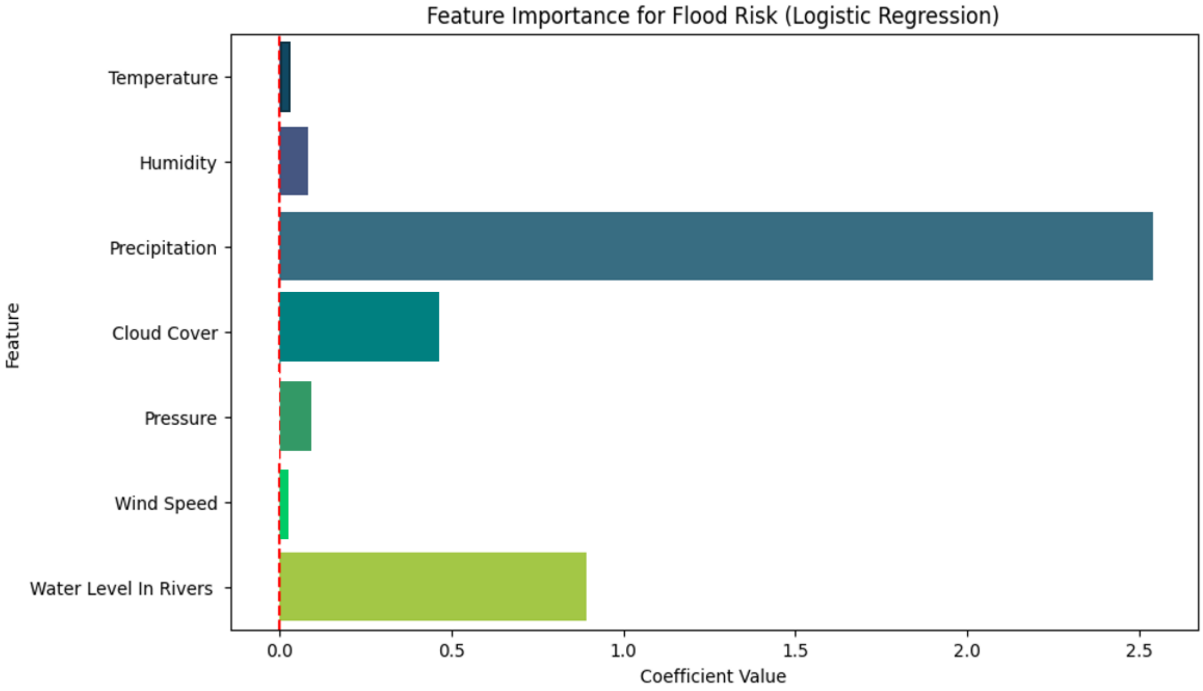


Figure 4.2 Feature importance for flood risk

CONCLUSION

This project demonstrates the effective application of machine learning techniques for rainfall prediction and flood risk forecasting, addressing a critical global challenge exacerbated by climate change. By leveraging meteorological data, geographical features, and historical flood records, the predictive model developed in this project provides a robust tool for improving disaster preparedness and resource allocation. The integration of machine learning algorithms, mathematical foundations, and performance evaluation metrics ensures that the system is both accurate and interpretable, offering valuable insights for decision-makers.

In addition to the predictive model, the project also includes the development of a structured SQL database for managing user credentials and data, as well as a user-friendly Python-based interface that facilitates seamless interaction with the system. This holistic approach ensures that the system is practical, accessible, and useful for end-users.

The project highlights the relevance of artificial intelligence in addressing climate-related challenges and contributing to sustainable development and climate resilience. As the demand for innovative solutions in environmental management and urban planning grows, this work aligns with ongoing efforts to apply machine learning in high-impact, real-world applications.

Future work could explore further improvements in model accuracy through the incorporation of additional data sources, more advanced machine learning algorithms, and real-time forecasting capabilities. The integration of these advancements could enhance the predictive system’s ability to respond to rapidly changing weather patterns, ultimately strengthening flood risk management efforts on a global scale.

REFERENCES

1. Parmar A. et al. Machine learning techniques for rainfall prediction: A review //International conference on innovations in information embedded and communication systems. – 2017. – T. 3. [1]

2. Mahesh B. Machine learning algorithms-a review //International Journal of Science and Research (IJSR).[Internet]. – 2020. – T. 9. – №. 1. – C. 381-386. [2]

3. Singh A., Thakur N., Sharma A. A review of supervised machine learning algorithms //2016 3rd international conference on computing for sustainable global development (INDIACom). – Ieee, 2016. – C. 1310-1315. [3]

4. Ayodele T. O. Types of machine learning algorithms //New advances in machine learning. – 2010. – T. 3. – №. 19-48. – C. 5-1. [4]

5. Basha C. Z. et al. Rainfall prediction using machine learning & deep learning techniques //2020 international conference on electronics and sustainable communication systems (ICESC). – IEEE, 2020. – C. 92-97. [5]

6. Hussein E. A. et al. Rainfall prediction using machine learning models: literature survey //Artificial Intelligence for Data Science in Theory and Practice. – 2022. – C. 75-108. [6]

7. Mohammed M. et al. Prediction of rainfall using machine learning techniques //International Journal of Scientific and Technology Research. – 2020. – T. 9. – №. 1. – C. 3236-3240. [7]

8. Barrera-Animas A. Y. et al. Rainfall prediction: A comparative analysis of modern machine learning algorithms for time-series forecasting //Machine Learning with Applications. – 2022. – T. 7. – C. 100204. [8]

Appendix

User interface

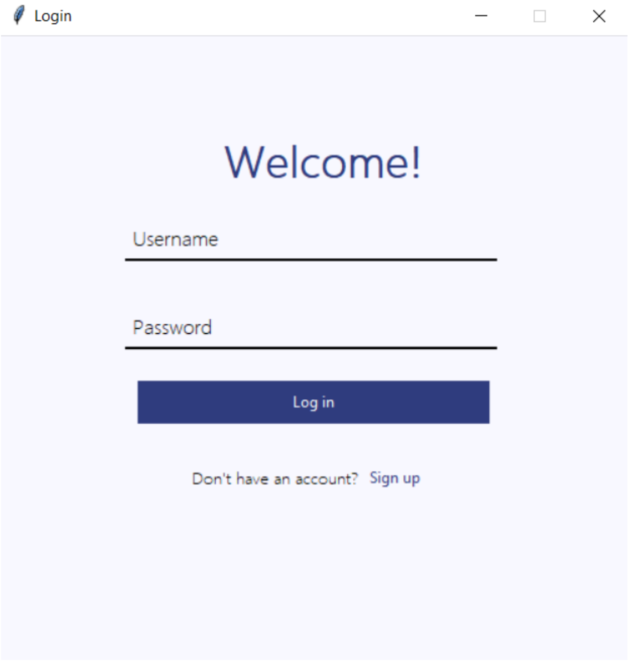


Figure 1. Login page

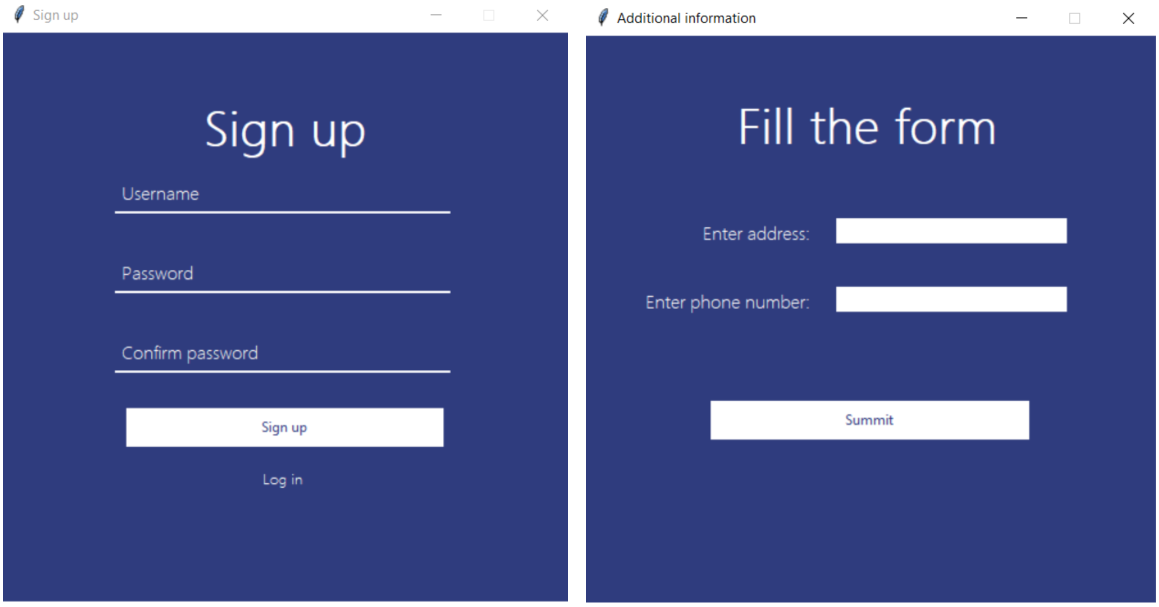
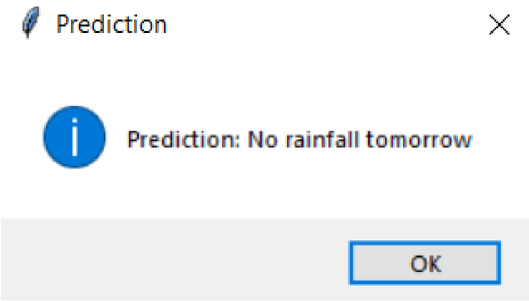
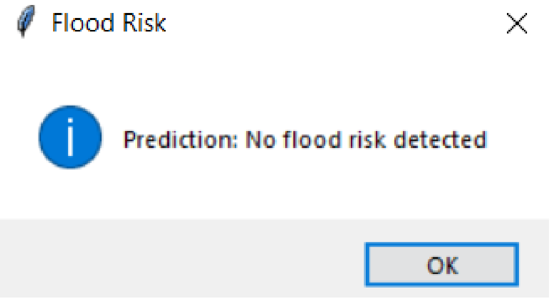


Figure 2. Sign up page

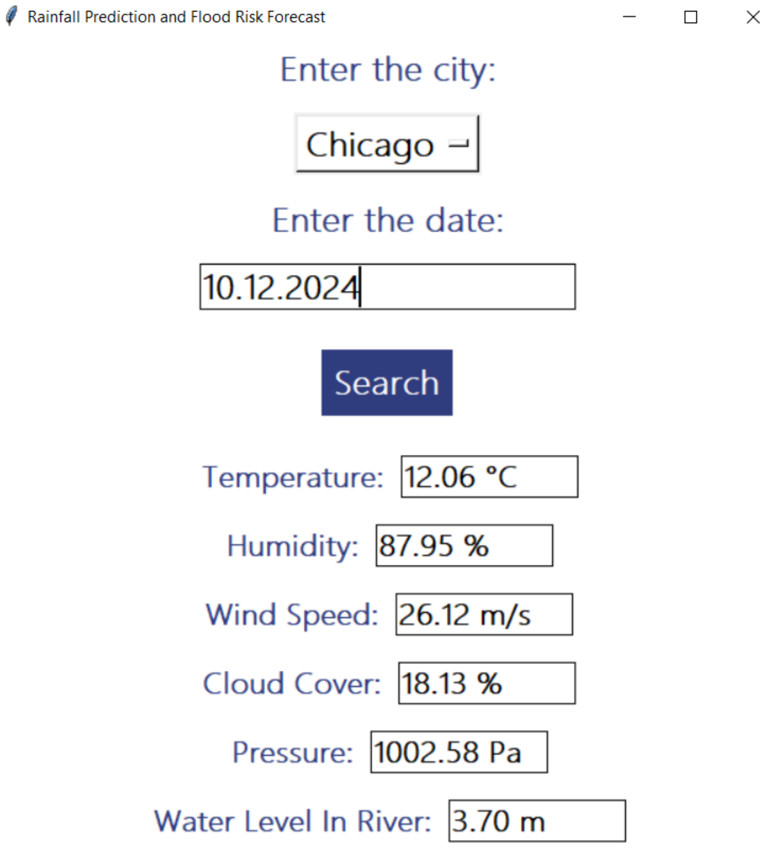


Prediction: No rainfall tomorrow



Prediction: No flood risk detected

Figure 3. Prediction window



Rainfall Prediction and Flood Risk Forecast

Enter the city:

Chicago

Enter the date:

10.12.2024

Temperature: 12.06 °C

Humidity: 87.95 %

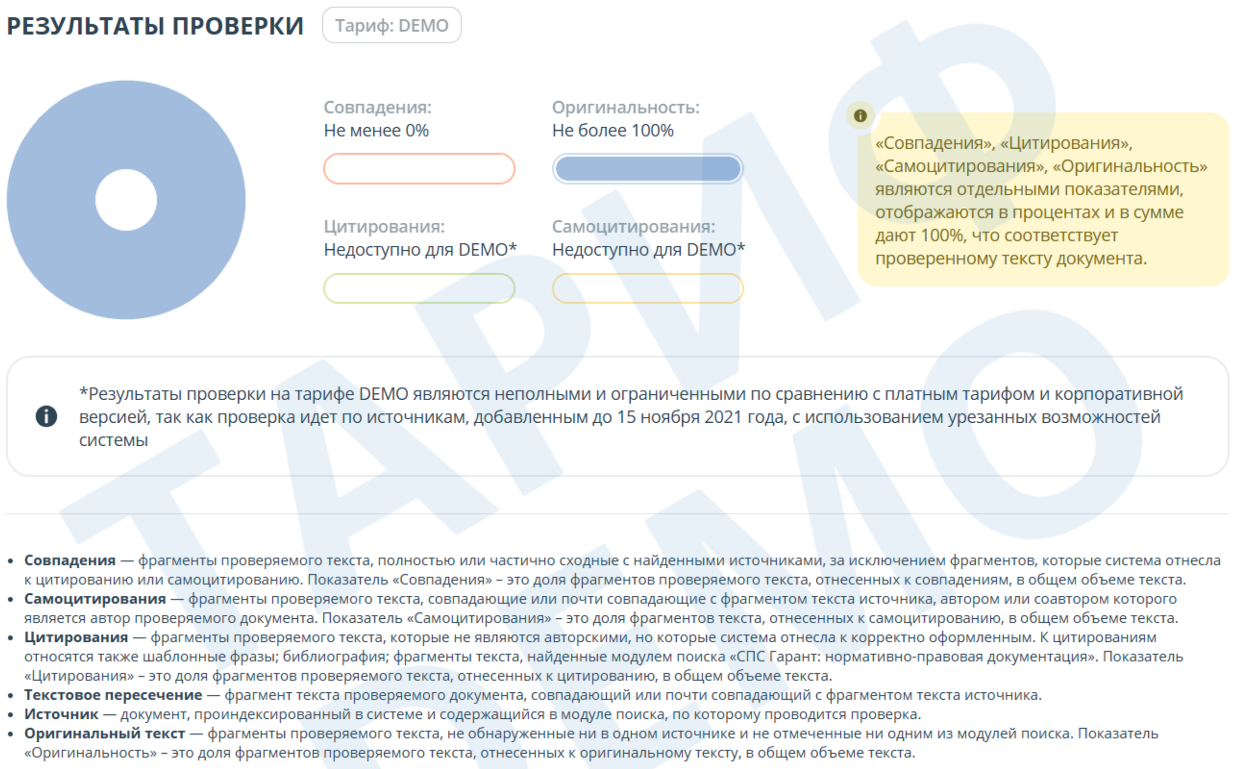
Wind Speed: 26.12 m/s

Cloud Cover: 18.13 %

Pressure: 1002.58 Pa

Water Level In River: 3.70 m

Figure 4. Main page



РЕЗУЛЬТАТЫ ПРОВЕРКИ

Тариф: DEMO

Совпадения:
Не менее 0%

Оригинальность:
Не более 100%

Цитирования:
Недоступно для DEMO*

Самоцитирования:
Недоступно для DEMO*

«Совпадения», «Цитирования», «Самоцитирования», «Оригинальность» являются отдельными показателями, отображаются в процентах и в сумме дают 100%, что соответствует проверенному тексту документа.

*Результаты проверки на тарифе DEMO являются неполными и ограниченными по сравнению с платным тарифом и корпоративной версией, так как проверка идет по источникам, добавленным до 15 ноября 2021 года, с использованием урезанных возможностей системы

- **Совпадения** — фрагменты проверяемого текста, полностью или частично сходные с найденными источниками, за исключением фрагментов, которые система отнесла к цитированию или самоцитированию. Показатель «Совпадения» – это доля фрагментов проверяемого текста, отнесенных к совпадениям, в общем объеме текста.
- **Самоцитирования** — фрагменты проверяемого текста, совпадающие или почти совпадающие с фрагментом текста источника, автором или соавтором которого является автор проверяемого документа. Показатель «Самоцитирования» – это доля фрагментов текста, отнесенных к самоцитированию, в общем объеме текста.
- **Цитирования** — фрагменты проверяемого текста, которые не являются авторскими, но которые система отнесла к корректно оформленным. К цитированиям относятся также шаблонные фразы; библиография; фрагменты текста, найденные модулем поиска «СПС Гарант: нормативно-правовая документация». Показатель «Цитирования» – это доля фрагментов проверяемого текста, отнесенных к цитированию, в общем объеме текста.
- **Текстовое пересечение** — фрагмент текста проверяемого документа, совпадающий или почти совпадающий с фрагментом текста источника.
- **Источник** — документ, проиндексированный в системе и содержащийся в модуле поиска, по которому проводится проверка.
- **Оригинальный текст** — фрагменты проверяемого текста, не обнаруженные ни в одном источнике и не отмеченные ни одним из модулей поиска. Показатель «Оригинальность» – это доля фрагментов проверяемого текста, отнесенных к оригинальному тексту, в общем объеме текста.