

**National Institute of Electronics & Information Technology**

**राष्ट्रीय इलेक्ट्रॉनिकी एवं सूचना प्रौद्योगिकी संस्थान**

**MACHINE LEARNING INTERNSHIP REPORT**

# Diabetics Prediction System

**Submitted By:**

**Name: Sai Ranjan Panda**

**Program: MCA**

**Institution: Utkal University**

**Reg. Number: 15830V24028**

# Submitted To Organization: NIELIT,BHUBANESWAR

**Acknowledgment**

I take this opportunity to express my profound gratitude to Utkal University and the Department of Computer Science and Applications for facilitating the successful completion of my internship titled "Foundation Course on Machine Learning using Python."

I am especially thankful to Khageswar Behera and Bijaylaxmi Behera, whose expert guidance, constructive feedback, and consistent encouragement were invaluable throughout the course of this internship. Their mentorship played a critical role in enhancing both my theoretical understanding and practical application of machine learning techniques using Python.

I would also like to extend my sincere appreciation to the coordinators and faculty members associated with the internship programme for designing a structured, industry-relevant learning experience that effectively bridged the gap between academic knowledge and practical implementation.

**NAME: Sai Ranjan Panda REG. NO: 15830V24028**

**Certificate of Completion**

This is to certify that Mr. Juel Dash, a student of the Department of Computer Science and Applications, Utkal University, has successfully completed the internship titled "Foundation Course on Machine Learning using Python", conducted during the period 15-05-2025 to 12-07-2025.This internship was carried out under the guidance and supervision of Khageswar Behera and Bijaylaxmi Behera, Instructors, at NIELIT.We acknowledge the intern's active participation, consistent effort, and sincere contribution throughout the internship duration. The student has demonstrated a good understanding of machine learning concepts and hands-on proficiency in Python.

**Signature of Guide**

**A Project Report On Diabetes Prediction System**

1. **Introduction**

Diabetes is a chronic medical condition that affects the way the human body processes blood glucose (sugar). Early detection and management are crucial to prevent complications such as cardiovascular diseases, kidney failure, and vision problems. With the growth of data science and machine learning, it has become possible to predict the likelihood of diabetes in individuals using historical data and predictive modeling. This project aims to develop a diabetes prediction system using machine learning techniques on a publicly available dataset.

1. **Objective**

The primary objective of this project is to build an intelligent system capable of classifying air quality into standard AQI categories (e.g., Good, Moderate, Poor) using supervised learning models and deploying it through an interactive web interface.

1. **Technologies Used**

The following key technologies and libraries were utilized in the development of this Diabetes Prediction System:

* **Python**: The core programming language for the project.
* **NumPy**: Essential for numerical operations, especially array manipulations and mathematical functions.
* **Pandas**: Used extensively for data manipulation, cleaning, and analysis, particularly for handling DataFrames.
* **Matplotlib**: Used for creating static, animated, and interactive visualizations in Python, specifically for generating the confusion matrix and ROC curve plots.
* **Seaborn**: A data visualization library based on Matplotlib, providing a high-level interface for drawing attractive and informative statistical graphics, used here for enhancing plot aesthetics.
* **Scikit-learn**: The primary machine learning library, providing tools for data preprocessing (e.g., StandardScaler, train\_test\_split), model selection (e.g., LogisticRegression, SVC, RandomForestClassifier, GradientBoostingClassifier), and model evaluation (e.g., accuracy\_score, precision\_score, recall\_score, f1\_score, roc\_auc\_score, confusion\_matrix).
* **Streamlit**: Employed to rapidly build and deploy the interactive web application, facilitating user input and visualization of results.

1. **Dataset Description**

The dataset used for this project is the AQI of Different Cities Dataset, available from the Github Repository. It contains 29531 instances and 16 features, including:

* PM2.5
* PM10
* NO
* NO2
* NOx
* CO
* SO2
* O3
* Benzene
* Toluene

**5. Methodology**

The development process involved several key machine learning and software engineering stages:

**5.1. Data Collection and Loading**

The dataset is loaded directly from a public URL (UCI Machine Learning Repository via Kaggle). This ensures easy access and reproducibility. Streamlit's @st.cache\_data decorator is used to cache the dataset, preventing redundant loading during application reruns and enhancing performance.

**5.2. Data Preprocessing**

Data preprocessing is a critical step to prepare the raw data for model training. The following steps were performed:

* **Handling Missing Values:** In the Pima Indians dataset, a value of 0 for features like Glucose, BloodPressure, SkinThickness, Insulin, and BMI often indicates a missing or unrecorded measurement, as these values cannot realistically be zero in a living person. These 0 values were replaced with NaN (Not a Number) and then imputed using the median of their respective columns. The median is preferred over the mean as it is less sensitive to outliers, which can be present in medical data.
* **Feature Scaling:** Numerical features were scaled using StandardScaler. This transformation adjusts the data to have a mean of 0 and a standard deviation of 1. Scaling is essential for algorithms sensitive to feature magnitudes, such as Logistic Regression and Support Vector Machines, ensuring that no single feature dominates the learning process due to its larger numerical range.
* **Feature and Target Separation:** The dataset was divided into X (features) and y (target variable, 'Outcome').
* **Train-Test Split:** The data was split into training (80%) and testing (20%) sets using train\_test\_split. The stratify=y parameter was used to ensure that the proportion of

diabetic to non-diabetic cases is maintained across both the training and testing sets,which is crucial for balanced evaluation, especially in classification tasks.

**5.3. Model Training**

Multiple classification algorithms were explored to identify the most suitable model for diabetes prediction. A Scikit-learn Pipeline was not explicitly used in the Streamlit code provided, but the preprocessing (scaling) is applied separately before model training and then applied consistently to new input data. The @st.cache\_resource decorator caches the trained models to avoid retraining on every interaction.

The following algorithms were implemented:

* **Logistic Regression:** A linear model widely used for binary classification. It's simple, efficient, and provides probability estimates**.**
* **Support Vector Machine (SVC):** A powerful algorithm that constructs a hyperplane or set of hyperplanes in a high-dimensional space to separate data points into different classes. The probability=True parameter was set to enable the calculation of prediction probabilities for the ROC AUC score.
* **Random Forest Classifier:** An ensemble learning method that builds multiple decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. It's robust to overfitting and handles non-linear relationships well.
* **Gradient Boosting Classifier:** Another ensemble technique that builds trees sequentially, where each new tree corrects the errors made by previous ones. It generally provides high predictive accuracy.

**5.4. Model Evaluation**

The performance of each trained model was rigorously evaluated using standard classification metrics:

* **Accuracy:** The proportion of correctly predicted instances out of the total instances.
* **Precision**: The ratio of correctly predicted positive observations to the total predicted positives. It minimizes false positives.
* **Recall (Sensitivity):** The ratio of correctly predicted positive observations to all actual positives. It minimizes false negatives, which is crucial in medical diagnosis (avoiding missed diabetes cases).
* **F1 Score:** The harmonic mean of Precision and Recall, providing a balance between the two.
* **ROC AUC Score:** The Area Under the Receiver Operating Characteristic Curve. It measures the classifier's ability to distinguish between classes. A higher AUC indicates better discriminatory power.
* **Confusion Matrix:** A table that visualizes the performance of a classification model, showing true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

**6. Streamlit Application Development**

A user-friendly web application was developed using Streamlit, providing an interactive interface for the system:

* **Data Overview:** Displays the head of the loaded dataset and its dimensions.
* Preprocessing Summary: Shows a glimpse of the scaled data and details about the train-test split.
* **Model Selection:** Allows users to choose their preferred classification algorithm from a dropdown menu.
* **Model Training Button:** Initiates the training process for the selected algorithm.
* **Performance Metrics:** Presents the calculated accuracy, precision, recall, F1 score, and ROC AUC score of the trained model on the test set.
* **Visualizations:** Includes a Confusion Matrix heatmap and a Receiver Operating Characteristic (ROC) Curve plot to visually represent model performance.
* **Interactive Prediction Form:** Enables users to input new medical parameters (Pregnancies, Glucose, Blood Pressure, etc.) via intuitive sliders. The system then uses the trained model and the *fitted scaler* to preprocess the new input and generate a real-time prediction (Likely/Unlikely to have Diabetes) along with a confidence score.
* **Session State Management:** st.session\_state is used to persist the trained model and scaler across Streamlit reruns, ensuring they are available for new predictions without re-training

**7. Results and Discussion**

The AQI Prediction System successfully loads, preprocesses, trains, and evaluates multiple machine learning models to classify air quality into standard AQI categories based on pollutant data. The Streamlit interface provides an intuitive way for users to interact with the models, train classifiers, and view performance metrics in real time. Initial testing with algorithms such as Logistic Regression, SVC, Random Forest, and Gradient Boosting reveals varying levels of predictive accuracy, with ensemble methods like Random Forest and Gradient Boosting often achieving superior results due to their ability to model complex, non-linear relationships in the data. The system evaluates models using key classification metrics such as accuracy, F1-score, and confusion matrices, giving a comprehensive view of performance. The interactive prediction feature further enhances the system’s utility by enabling users to input live pollutant levels and instantly receive the predicted AQI category, thereby demonstrating the practical applicability of the system for quick assessments and environmental monitoring.

**8. Conclusion**

This AQI Prediction System project successfully integrates Scikit-learn, NumPy, Pandas, and Streamlit to deliver a functional and interactive machine learning application. It showcases the end-to-end process of building a predictive model, from data acquisition and preprocessing to model training, evaluation, and deployment, through a user-friendly interface. The project demonstrates the practical application of machine learning in environmental monitoring and serves as a solid foundation for future enhancements, such as incorporating real-time data streams, implementing cross-validation and hyperparameter tuning, exploring deep learning models, and adding advanced visualization tools for better interpretability.

**9. References**

Aie Quality Index(AQI) Dataset:

* Available from the Central Pollution Control Board (India) via the AQI open dataset initiative:

https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india

(Used via Kaggle’s raw dataset link as referenced in the code.)