**MLOPS Group 51**

1. RAVISHANKAR R < 2022ac05117@wilp.bits-pilani.ac.in>
2. VISHAL PERIYASAMY R < 2022ac05033@wilp.bits-pilani.ac.in>
3. SAKTHI R < 2022ac05659@wilp.bits-pilani.ac.in>
4. YOGEESH BABU B R < 2022ac05544@wilp.bits-pilani.ac.in>
5. YASHODHA R < 2022ac05366@wilp.bits-pilani.ac.in>

**Summary:**

**1.** **Data Collection and Preprocessing**

We chose the Water Potability Dataset, which contains features related to water quality. The goal is to predict whether water is potable (safe to drink) based on these features:

* ph: pH level of the water
* Hardness: Measures of water hardness (CaCO3 in mg/L)
* Solids: Total dissolved solids in the water (ppm)
* Chloramines: Residual chlorine in the water (ppm)
* Sulfate: Concentration of sulfate in mg/L
* Conductivity: Electrical conductivity of water in μS/cm
* Organic\_carbon: Organic carbon concentration in mg/L
* Trihalomethanes: Trihalomethanes concentration in μg/L
* Turbidity: Water cloudiness in NTU

**1. Data Cleaning**

**Handling Missing Values:** Missing values were identified in multiple features. We used mean imputation for these features because it maintains the central tendency of the data without introducing bias.

**Outliers Removal**: Outliers were addressed using IQR-based filtering, as extreme values can skew the model and lead to poor generalization.

**Impact:** Cleaning the data ensures the model isn't biased by missing or extreme values, allowing it to generalize better to unseen data.

2**. Feature Engineering**

**Impact:** Feature engineering helps the model capture complex relationships and patterns in the data that might not be obvious from raw features.

3**. Scaling and Normalization**

**Scaling**: We applied Min-Max scaling to normalize the features to a range between 0 and 1. This was particularly important for features like Solids and Conductivity, which are on a different scale than other features.

**Normalization**: For models that assume a Gaussian distribution (e.g., Logistic Regression), we also tried Z-score normalization to standardize the features based on the mean and standard deviation.

**Impact:** Scaling helps models like K-Nearest Neighbours and Neural Networks perform better by ensuring all features contribute equally to the predictions, while normalization improves convergence in algorithms like Gradient Descent.

**4. AutoEDA with KizenML**

We utilized KizenML for Automated Exploratory Data Analysis (AutoEDA), which helped us quickly generate insights into feature distributions, correlations, and missing values. KizenML also provided an overview of the importance of features, aiding in the feature selection process.

**Impact**: AutoEDA with KizenML accelerates the data's understanding, ensuring that the most critical features are considered in the final model, and reducing the risk of introducing unnecessary complexity.

**2. Model Selection, Training, and Hyperparameter Tuning**

For this task, we focused on **Logistic Regression** as our primary model, as it is well-suited for binary classification problems like water potability prediction. Instead of exploring multiple model types, we concentrated on tuning the hyperparameters of Logistic Regression to achieve optimal performance.

**1. Why Logistic Regression?**

Logistic Regression is a simple yet powerful algorithm for classification tasks, particularly when interpretability and efficiency are important. Since the dataset features continuous variables and the target variable is binary (potable or not potable), Logistic Regression was a natural choice for this problem.

**2. Hyperparameter Tuning Process**

We conducted extensive hyperparameter tuning using techniques like **Grid Search** and **Random Search** to find the best parameter combinations for the Logistic Regression model. Below are the key hyperparameters tuned:

* **Regularization Strength (C)**: We tested different values of C (inverse of regularization strength) to balance between overfitting and underfitting. A lower C enforces stronger regularization, while a higher C allows the model more flexibility.
* **Penalty**: We experimented with both **L1** and **L2** regularization to see which form of regularization produced the best results for our dataset. L1 encourages sparsity in the model, while L2 penalizes large coefficients.

**3. Performance Evaluation**

During the tuning process, we used **cross-validation** to evaluate the model's performance across different sets of hyperparameters. This ensured that the model generalizes well to unseen data and avoids overfitting. The following metrics were considered for model evaluation:

* **Accuracy**: The proportion of correctly predicted labels.
* **Precision, Recall, and F1-score**: These metrics were particularly important to ensure a balanced performance between classes, especially in cases of class imbalance.
* **ROC-AUC**: To evaluate the model’s ability to distinguish between potable and non-potable water across different thresholds.

**4. Best Performing Model**

After hyperparameter tuning, the best-performing configuration for **Logistic Regression** was:

* **C = 0.1**
* **Penalty = L2**

This combination achieved the highest accuracy and balanced precision and recall for the potability prediction task.

**5. Experimentation Process Using AutoML (KizenML)**

We utilized **KizenML** to automate parts of the hyperparameter tuning process. KizenML’s AutoML capabilities allowed us to efficiently explore a range of hyperparameter values without manually defining each configuration. It also provided valuable insights into model performance across different parameter settings, making the tuning process faster and more systematic.

**3. Explainable AI (XAI) Implementation**

For this project, we applied **Explainable AI (XAI)** techniques to make our **Logistic Regression** model's predictions interpretable. Since Logistic Regression is inherently interpretable due to its linear nature, we further enhanced the transparency of the decision-making process using **SHAP (Shapley Additive explanations)**.

**1. Importance of Interpretability**

Interpretability is crucial for models that impact critical decision-making areas like water potability. Being able to understand how the model arrives at its predictions increases trust and enables stakeholders to make informed decisions based on the results. Furthermore, interpretability helps identify any potential biases or flaws in the model, ensuring that predictions are both accurate and reliable.

**2. SHAP (Shapley Additive explanations)**

We used **SHAP** to explain the contribution of each feature to the model's predictions. SHAP values provide insights into how much each feature influenced the output for a given instance. In our case, SHAP helped us understand which water quality features (such as pH, Hardness, or Solids) played the most significant role in predicting whether the water was potable or not.

**SHAP Workflow:**

* **Global Interpretability**: SHAP was used to generate a global view of feature importance. Features such as Turbidity, Chloramines, and Organic carbon had the most substantial impact on the model's predictions. This helped us understand how the model behaves across the entire dataset.
* **Local Interpretability**: For individual predictions, SHAP values illustrated how specific features contributed to the decision for that particular instance. For example, in one case, a high Solids level might have had a strong negative influence on water potability, which SHAP clearly highlighted.

**3. Key Insights from SHAP**

* **Positive and Negative Contributions**: SHAP values allowed us to see both the positive and negative contributions of each feature. For instance, higher levels of **Chloramines** and **Trihalomethanes** often contributed negatively to water potability, while features like **ph** and **Hardness** tended to have more positive influences.
* **Feature Ranking**: The SHAP summary plots clearly ranked features based on their overall importance in the model. This provided an interpretable ranking of the most influential factors affecting water quality predictions.

**4. How XAI Tools Improved Interpretability**

SHAP offered both **global** and **local** explanations, giving us a comprehensive understanding of how the model works:

* **Global Level**: It helped identify the most impactful features across all predictions, enabling us to focus on the most relevant variables for future data collection and analysis.
* **Local Level**: It allowed for a deep dive into specific predictions, explaining why the model made a certain decision for individual samples.

**4. Docker and Flask Application**

1. **Dockerization**: The application is containerized using Docker to ensure consistency across different environments.
2. **Flask API**: A Flask application is developed to serve the machine learning model, providing a RESTful API for inference.

**Reason for Using Docker and Flask:** Containerization with Docker ensures consistent environments across development and production. Flask was chosen for its simplicity and flexibility in creating web applications and APIs, making it a suitable choice for serving machine learning models.

**5. Deploying to AWS**

1. **Build and Push Docker Image**: The model is containerized, built, and pushed to Amazon ECR.
2. **Prepare CloudFormation Template**: A CloudFormation template is created to deploy the model on Amazon ECS.
3. **Deploy ECS Service**: The ECS service is configured and deployed using the prepared CloudFormation template.

**Reason for Choosing ECR & ECS:** ECS was selected for its ability to manage Docker containers in a scalable manner. Amazon ECR provides a scalable and secure repository for Docker images.