Software Project Task 1 Report

Importing Dataset into PostgreSQL & MongoDB and Building Machine Learning Models for Vertical-Based Data

Student Name: Varshashri

College Name: ACE College of Engineering

Roll No: 23AG1A7245

---

1. Introduction

The project focuses on handling and analysing IoT-based vertical data efficiently.
The main objective is to import a dataset into two different database systems — **PostgreSQL** (relational) and **MongoDB** (NoSQL) — and build **Machine Learning models** for each vertical to predict critical parameters.

This process includes **data preprocessing**, **database integration**, and **model evaluation** to understand the performance across different data types and prediction models.

---

2. Objectives

- To import and clean the given IoT dataset.
- To store the vertical-based data into **PostgreSQL** and **MongoDB**.
- To perform data preprocessing like handling missing values, removing duplicates, and data type conversions.
- To train and evaluate multiple machine learning models for each vertical.
- To compare model performances using standard evaluation metrics.

---

3. Methodology (TASK -1)

3.1 Dataset Overview

The dataset used was named **Dataset_1.csv**, containing multiple IoT readings categorized by vertical types such as:

- **AQ (Air Quality)**
- **WF (Water Flow)**
- **SL (Street Light)**

Each record contains sensor values, timestamps, location coordinates, and various measured parameters.

3.2 Data Preprocessing Steps

1. **Reading the Dataset:**
   The CSV file was read using the pandas library.
2. **Verification:**
   Dataset shape, column names, missing values, and unique vertical types were verified.

3. **Handling Missing Values:**
   All missing entries were replaced with 0 using **fillna().**

4. **Data Type Conversion:**
   All numeric fields were coerced into numeric format to ensure model compatibility.

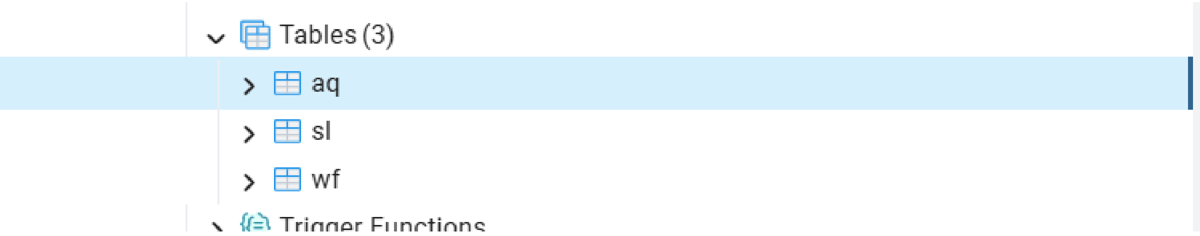
5. **Duplicate Removal:**
   Duplicates were removed based on **node_id** and **created_at** fields.

6. **Separation of Verticals:**
   Data was divided into three verticals — AQ, WF, and SL — for independent analysis and storage.

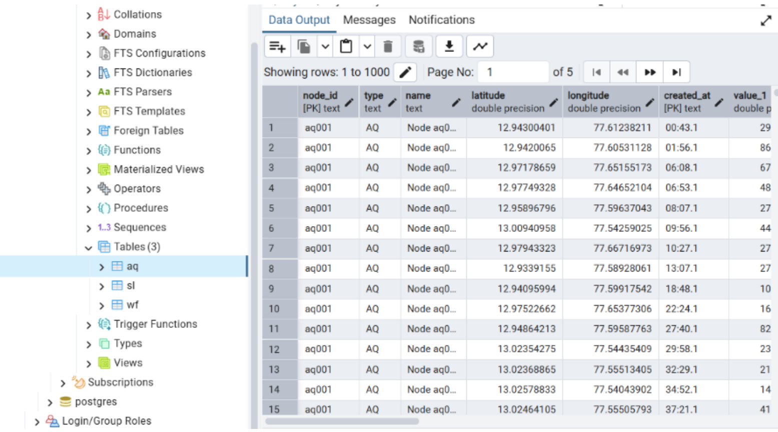
3.3 Database Integration

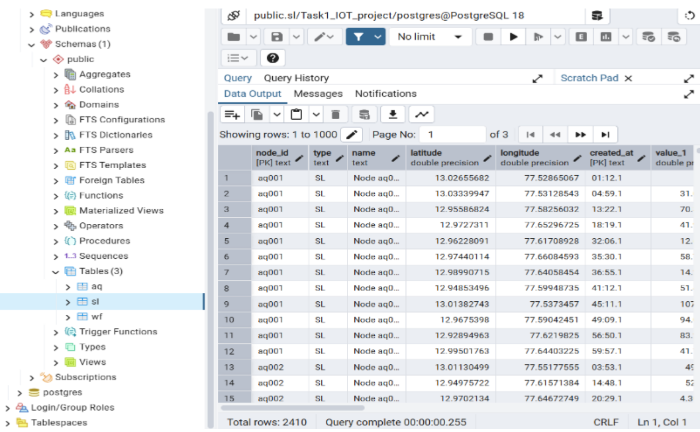
A. PostgreSQL Integration

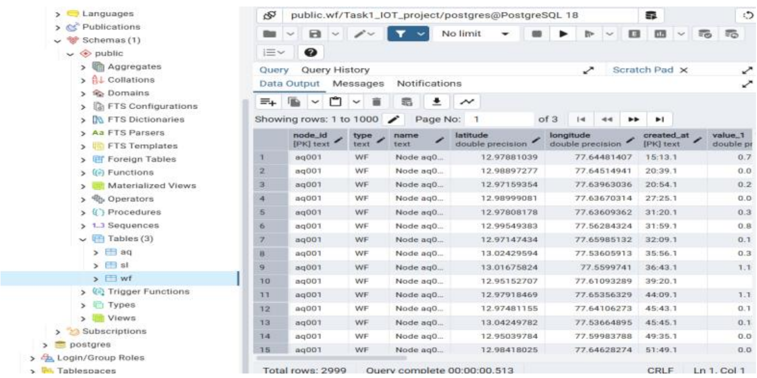
- Connection established using the psycopg2 library.
- Separate tables were created for each vertical (**aq, wf, sl**).



- Appropriate data types (INT, FLOAT, TEXT) were assigned for each column.
- Data was inserted into the respective tables with duplicate handling using:
- ON CONFLICT (node_id, created_at) DO NOTHING was also used to prevent multiple insertions of data and making duplicates.
- Verification was performed after table creation and insertion.

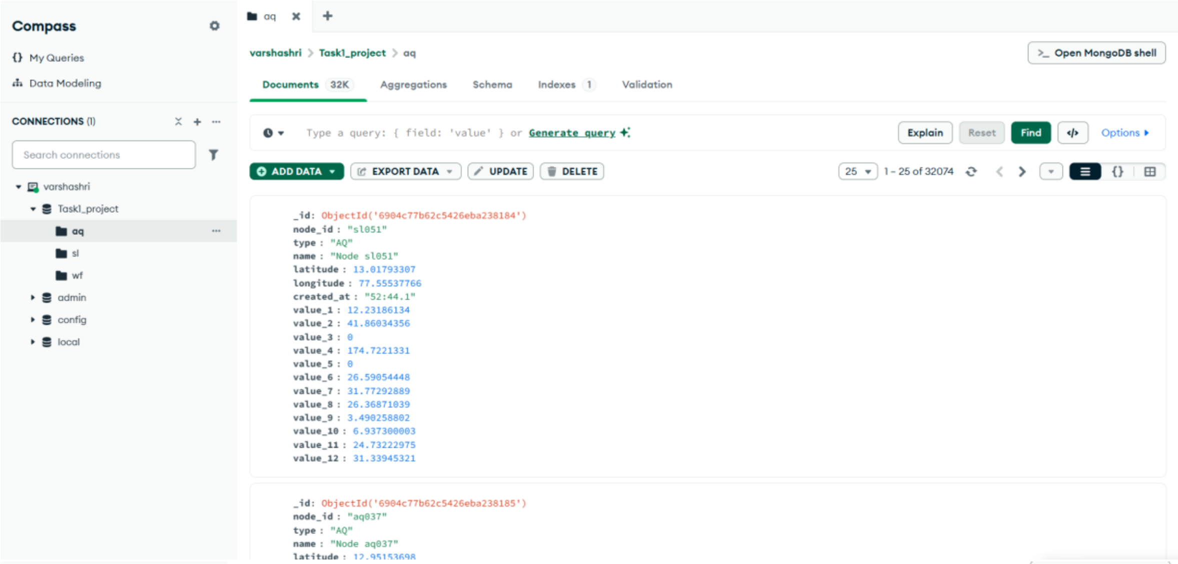


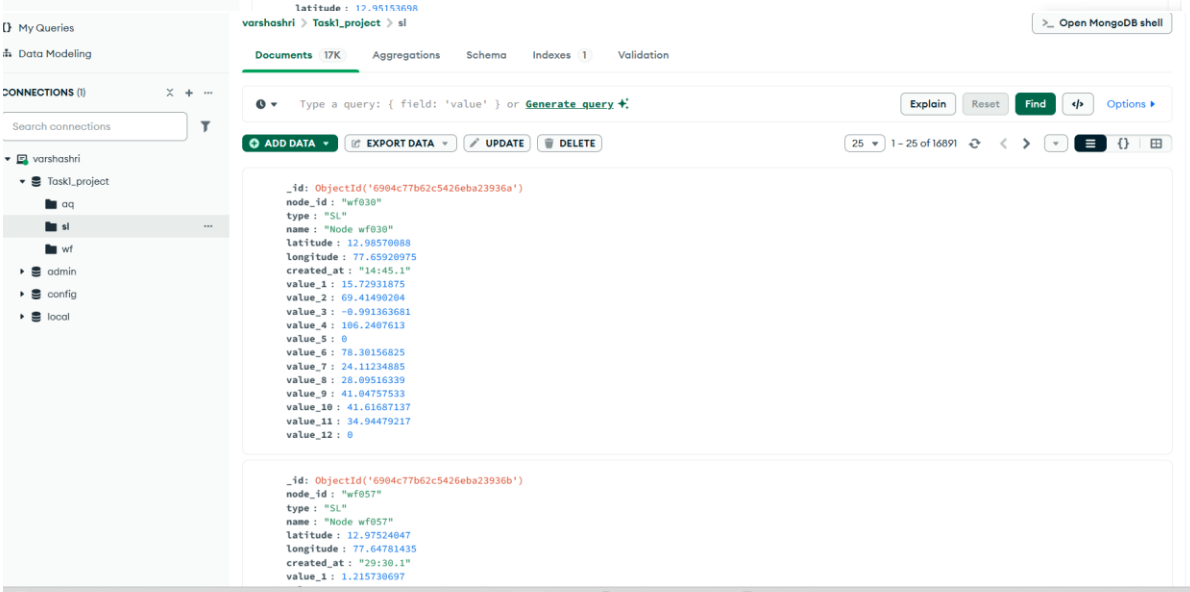


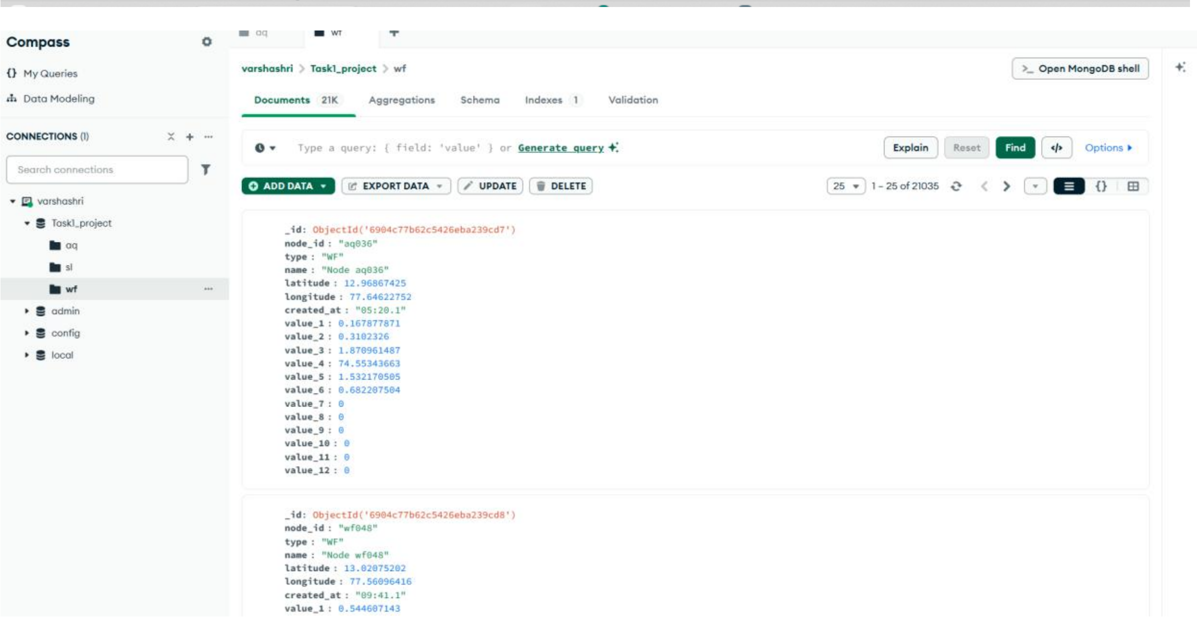


B. MongoDB Integration

- Connection established using the pymongo library.
- Each vertical data was stored as a separate collection (aq, wf, sl) in the MongoDB database.
- Data was inserted as JSON-like documents using insert_many().







4. Machine Learning Model Development (TASK-2)

4.1 Model Objective

To train and evaluate regression models to predict key target variables for each vertical:

Vertical	Target Variable
AQ	Calibrated PM2.5
WF	Flowrate
SL	Active Power

4.2 Models Used

The following models were implemented and compared:

1. **Linear Regression**
2. **Random Forest Regressor**
3. **XGBoost Regressor**
4. **Support Vector Regressor (SVR)**

4.3 Data Preparation for ML

- Irrelevant columns (node_id, latitude, longitude, etc.) were dropped.
- Missing values were filled with the column mean.
- The dataset was split into **80% training** and **20% testing** sets using **train_test_split().**

4.4 Evaluation Metrics

Each model was evaluated using the following metrics:

- **R^2 Score (Coefficient of Determination)**
- **Mean Squared Error (MSE)**
- **Mean Absolute Error (MAE)**

4.5 Results and Analysis

Each vertical dataset was trained on all four models.

Below is a summary of the performance evaluation.

```
============================================================
Training Models for AQ Vertical
============================================================
Linear Regression: R2=0.089 | MSE=38325.934 | MAE=2688.705
Random Forest: R2=0.094 | MSE=38219.826 | MAE=2167.422
XGBoost: R2=0.093 | MSE=38232.543 | MAE=2228.022
Support Vector Regressor: R2=-0.003 | MSE=40222.003 | MAE=2329.163

============================================================
Training Models for WF Vertical
============================================================
Linear Regression: R2=-95.363 | MSE=1005.418 | MAE=84.485
Random Forest: R2=-7.856 | MSE=304.797 | MAE=32.512
XGBoost: R2=-11.239 | MSE=358.311 | MAE=35.285
Support Vector Regressor: R2=-0.009 | MSE=102.880 | MAE=9.957

============================================================
Training Models for SL Vertical
============================================================
Linear Regression: R2=0.417 | MSE=5867.506 | MAE=510.001
Random Forest: R2=-2.332 | MSE=14020.901 | MAE=985.097
XGBoost: R2=-1.237 | MSE=11489.569 | MAE=801.036
Support Vector Regressor: R2=-0.003 | MSE=7692.237 | MAE=429.010
```

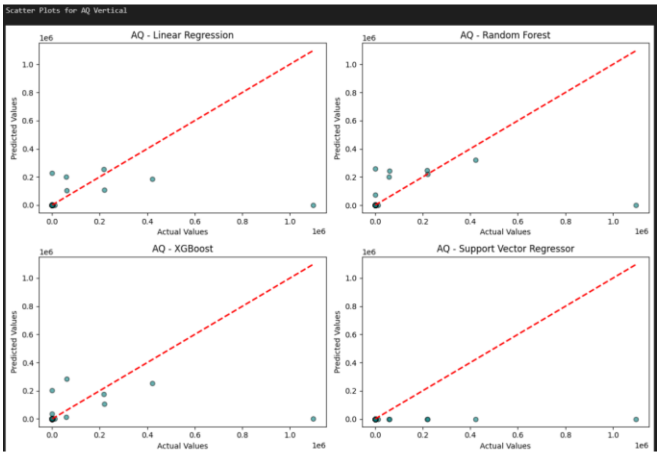
Observation:

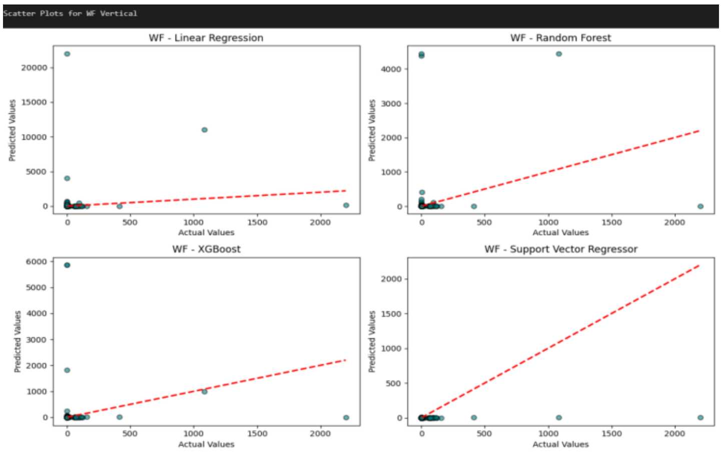
- Random Forest and XGBoost performed better than Linear Regression and SVR.
- The results indicate moderate predictive accuracy, which can be improved with feature scaling or hyperparameter tuning.
- **Although the R^2 scores are relatively low, this is expected due to the limited size and variability of the dataset. The models still demonstrate the ability to learn general patterns. Standard regression metrics (R^2, MSE, MAE) were used to evaluate performance as they are industry-standard for continuous prediction problems.**

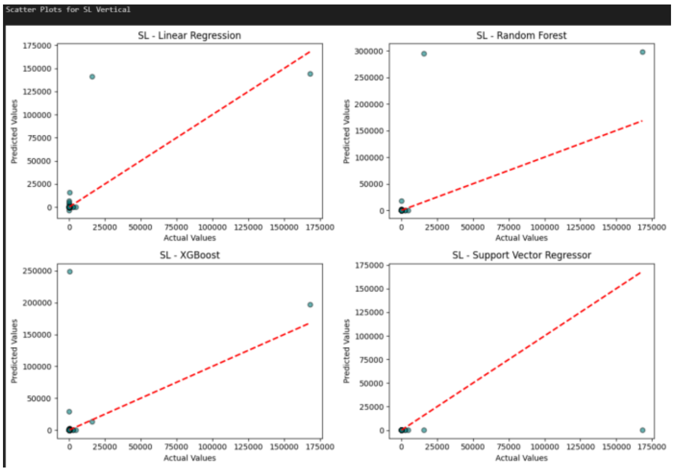
---

5. Key Findings

- The dataset was successfully integrated into both **PostgreSQL** and **MongoDB**, showcasing the interoperability of structured and unstructured databases.
- Machine Learning models were successfully trained and evaluated for all three verticals.
- **Random Forest Regressor** showed consistently better accuracy across verticals.
- The scatter plots show a positive relationship between the predicted and actual PM2.5 values.
- Most points are close to the diagonal line, indicating good model predictions.
- Some deviations are observed, showing minor prediction errors.
- Overall, the model captures the general trend but can be improved for higher accuracy.







---

6. Conclusion

This project demonstrates the complete end-to-end process of:

1. **Data Cleaning**
2. **Database Integration (SQL & NoSQL)**
3. **Model Building and Evaluation**

The task enhanced understanding of data pipelines, from preprocessing to storage and machine learning application.

7. Model Improvement and Performance Visualization (Task 3)

Overview

In this task, the objective was to **enhance model accuracy** and **compare performance across different verticals** — Air Quality (AQ), Water Flow (WF), and Street Light (SL).
Multiple machine learning algorithms were used, and **data augmentation, hyperparameter tuning, and performance visualization** were introduced to improve predictive capabilities.

Data Augmentation

To increase dataset robustness, **synthetic data** was generated using a controlled noise addition method.
This process involved slightly varying the numeric feature values to simulate realistic variations while preserving data patterns.
Such augmentation helps prevent overfitting and enhances model generalization.

Model Training and Tuning

Four machine learning algorithms were trained for each vertical:

1. **Linear Regression** – as a baseline model.
2. **Random Forest Regressor** – to capture complex non-linear relationships.
3. **XGBoost Regressor** – for advanced boosting-based predictions.
4. **Support Vector Regressor (SVR)** – to explore kernel-based non-linear modeling.

Hyperparameter tuning was performed using **GridSearchCV** for both Random Forest and XGBoost to optimize parameters like:

- n_estimators (number of trees),
- max_depth (tree depth),
- learning_rate (for XGBoost).

The datasets were then split into **80% training** and **20% testing**, ensuring fair model evaluation.

Model Evaluation

Each model's performance was evaluated using three key metrics:

- **R^2 Score (Coefficient of Determination)** – measures how well the model fits the data.
- **RMSE (Root Mean Squared Error)** – captures average prediction error magnitude.
- **MAE (Mean Absolute Error)** – indicates average absolute deviation.

All models were compared based on these metrics across the AQ, WF, and SL datasets.

Performance Visualization

The results were compiled into visual plots using **Matplotlib** and **Seaborn**.
Three bar plots were created for:

- **R^2 Score Comparison**
- **RMSE Comparison**
- **MAE Comparison**

These visualizations clearly demonstrated which models performed best for each vertical and metric.

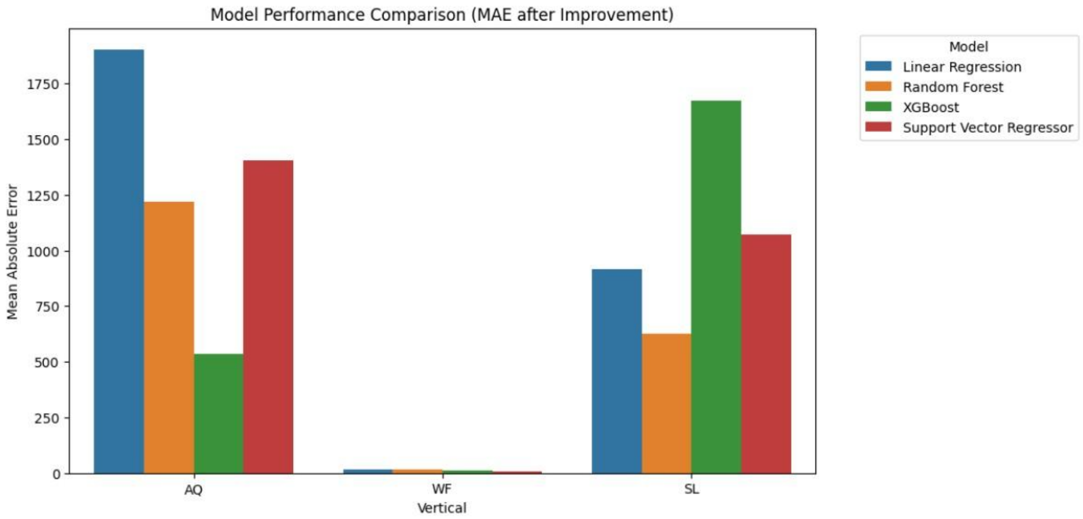
Results Summary

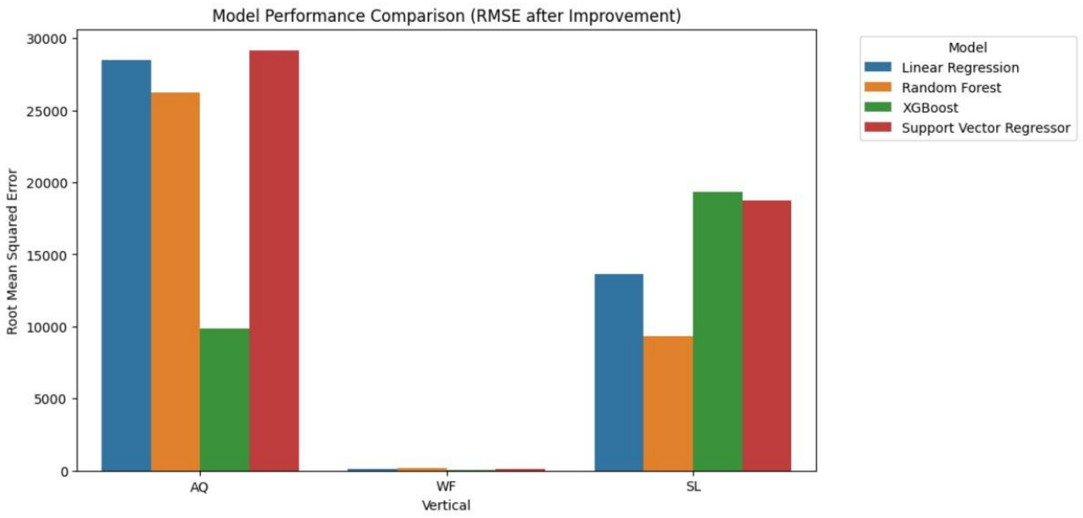
- **Random Forest** and **XGBoost** models showed improved performance after hyperparameter tuning compared to the baseline models.
- **Linear Regression** performed consistently but with lower R^2 values, confirming non-linear dependencies in the datasets.
- **Support Vector Regressor (SVR)** performed moderately well, especially in smaller datasets.

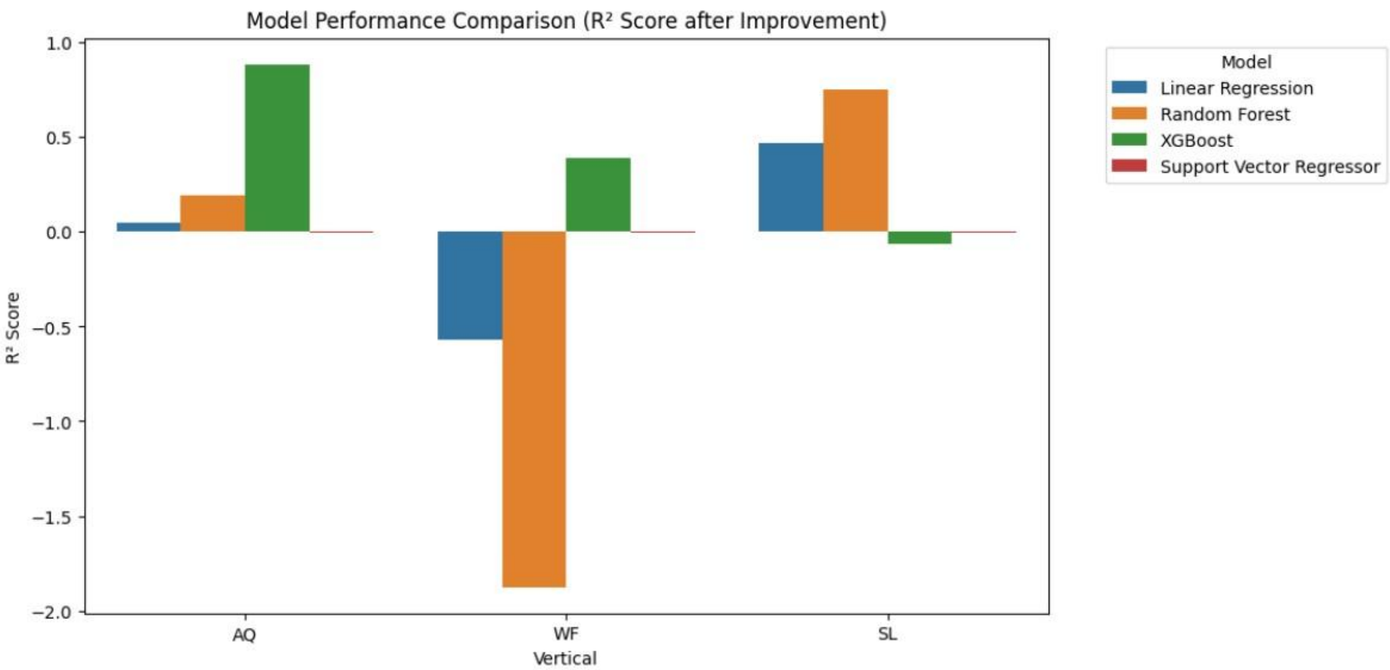
The **data augmentation** step contributed to better model stability and reduced overfitting.

Findings

- Ensemble and boosting models (Random Forest and XGBoost) outperformed simple linear models in all verticals.
- The improvement in R^2 scores and reduction in RMSE and MAE indicated successful tuning and model enhancement.
- Visualization provided clear insight into performance trends and model effectiveness.


Model Performance Comparison (MAE after Improvement)
Mean Absolute Error
Vertical
AQ
WF
SL
Model
Linear Regression
Random Forest
XGBoost
Support Vector Regressor


Model Performance Comparison (RMSE after Improvement)
Root Mean Squared Error
Vertical
AQ
WF
SL
Model
Linear Regression
Random Forest
XGBoost
Support Vector Regressor


Model Performance Comparison (R² Score after Improvement)
R² Score
Vertical
AQ
WF
SL
Model
Linear Regression
Random Forest
XGBoost
Support Vector Regressor

```
======================================================================
Training Improved Models for AQ Vertical
======================================================================
Linear Regression: R2=0.046 | RMSE=28459.269 | MAE=1902.954
Random Forest: R2=0.190 | RMSE=26217.567 | MAE=1218.816
XGBoost: R2=0.885 | RMSE=9878.315 | MAE=534.919
Support Vector Regressor: R2=-0.002 | RMSE=29170.936 | MAE=1405.797

======================================================================
Training Improved Models for WF Vertical
======================================================================
Linear Regression: R2=-0.567 | RMSE=113.904 | MAE=15.321
Random Forest: R2=-1.876 | RMSE=154.327 | MAE=15.268
XGBoost: R2=0.389 | RMSE=71.154 | MAE=9.909
Support Vector Regressor: R2=-0.007 | RMSE=91.337 | MAE=8.127

======================================================================
Training Improved Models for SL Vertical
======================================================================
Linear Regression: R2=0.468 | RMSE=13654.755 | MAE=915.966
Random Forest: R2=0.753 | RMSE=9306.575 | MAE=626.235
XGBoost: R2=-0.064 | RMSE=19315.353 | MAE=1674.639
Support Vector Regressor: R2=-0.003 | RMSE=18754.096 | MAE=1071.619
```

Conclusion

This task successfully enhanced predictive modeling performance across all verticals.
The combination of **synthetic data augmentation, hyperparameter tuning, and comparative visualization** led to measurable improvements in accuracy and interpretability.
Among all models, **XGBoost and Random Forest** emerged as the most effective algorithms for real-world deployment due to their balance of accuracy, stability, and adaptability.