Water Consumption Prediction

Data Preprocessing and Feature Engineering

A structured and modular approach was used for data preprocessing to ensure an optimized dataset for model training. This involved handling missing values, treating outliers, encoding categorical variables, and selecting the most relevant features using statistical techniques, including **Ordinary Least Squares (OLS), Variance Inflation Factor (VIF) analysis, and correlation analysis**.

1. Handling Missing Values

A targeted strategy was applied to handle missing values, depending on the nature of each feature:

- **Categorical Features:**
  - `ApartmentType` and `IncomeLevel`: Missing values in these categorical columns were filled with **"Unknown"** to preserve the presence of these records rather than discarding them. This approach ensures that missing categories are still accounted for in the model without introducing bias.
- **Numerical Features:**
  - **General Approach:** Missing values in most numerical features were replaced with **-1**, serving as an indicator for missingness while still retaining the data points.
  - **Temperature:** Since this variable had no significant outliers, the missing values were imputed using the **mean** of the existing values. Mean imputation was chosen due to its stability and minimal impact on variance.
  - **Amenities:** This feature had over **5000 missing values**, requiring a more advanced imputation method. A **Random Forest Regressor** was trained to predict the missing values by leveraging relationships with other features. This method was chosen over simple imputation techniques (such as mean or median) due to its ability to capture complex interactions between variables.

2. Outlier Treatment

Outliers were identified and treated systematically to prevent them from negatively affecting model performance. The approach followed:

- **Statistical Thresholding:** The **upper and lower bounds** were set based on the **1.5 × IQR (Interquartile Range) rule**, ensuring that extreme values did not distort the data distribution.
- **Winsorization:** Instead of removing outliers, extreme values were **clipped to the lower and upper bounds**, preserving data integrity while reducing noise.

3. One-Hot Encoding

- Categorical features such as **`ApartmentType` and `IncomeLevel`** were encoded using **`pd.get_dummies()`**, ensuring that they were converted into a format suitable for machine learning models.
- This method was chosen because it allows models to interpret categorical data without assuming ordinal relationships between categories.

4. Feature Engineering and Selection

Several statistical techniques were applied to assess feature importance and reduce redundancy:

OLS Regression Analysis:

- **Ordinary Least Squares (OLS)** was used to analyze feature significance in the model.
- This helped in determining which features contributed significantly to the target variable.

Correlation and VIF Analysis:

- **Correlation Analysis:**
  - Highly correlated features were identified and examined to **eliminate redundancy** and **prevent multicollinearity**.
- **Variance Inflation Factor (VIF):**
  - Features with high VIF scores were re-evaluated and either transformed or removed.
  - **Initially, several features had high VIF values**, but after eliminating **date-based features**, the VIF values of the remaining features dropped below 10, resolving multicollinearity.

Date-Based Feature Engineering and Removal:

- **Cyclical Transformation:**
  - Time-based features (`Year`, `Month`, `Day`, `Hour`) were transformed into **sine and cosine components** to capture periodicity.
  - However, after experimentation, it was found that these transformations did **not significantly improve predictive power**.
  - As a result, **date-based features were removed from the dataset.**

Final Feature Selection Decisions:

- **Date Parts Removal:** After testing cyclical transformations, it was found that **date parts did not significantly contribute to predictive power**. As a result, `Year`, `Month`, `Day`, and `Hour` were removed.
- **Humidity Removal:** The `Humidity` feature was found to have low relevance and was removed.

- **Final Feature Set:** After removing date-based features, **VIF values of remaining features dropped below 10**, indicating that multicollinearity was resolved. Important features such as **Temperature, Water Price, and Period Consumption Index were retained** as they played a crucial role in model prediction.

Model Training

For model training, we experimented with multiple regression models to determine the best-performing algorithm based on RMSE (Root Mean Squared Error). The following models were trained and evaluated:

- **Decision Tree Regressor**
- **Random Forest Regressor**
- **Gradient Boosting Regressor**
- **Linear Regression**
- **Support Vector Regressor (SVR)**
- **XGBoost Regressor** (`XGBRegressor` with `objective='reg:squarederror'`)
- **AdaBoost Regressor**

To optimize performance, we applied **GridSearchCV** for hyperparameter tuning on these models. The best parameters were selected based on the lowest RMSE score. Each model was evaluated using cross-validation, and GridSearchCV was employed to find the optimal hyperparameters. Based on performance metrics such as RMSE, MAE, and R^2, the best-performing model was selected for deployment

Artifacts Folder

In the **Artifacts** folder, we have organized all the essential files related to our machine learning workflow, from **data ingestion** to **model training** and beyond. This folder serves as a comprehensive repository for all outputs generated during the project. Here's a breakdown of its contents:

- **Data Ingestion**: Raw data files, along with pre-processed datasets, are stored to track all stages of data preparation. This ensures the entire pipeline can be retraced if necessary.
- **Model Training**: The folder contains serialized models (such as using `pickle`) that represent the final, trained versions of the models after tuning. This also includes configuration files for hyperparameter settings used during the training process, ensuring full reproducibility.
- **Test and Train Performances**: Performance metrics, such as RMSE, R^2, and other relevant evaluation scores, are saved for both training and testing datasets. These allow for easy tracking and comparison of model performance across different stages.
- **Predictions**: The predictions made by the best-performing model (determined by the lowest RMSE score) on the test data are also stored in the folder. This serves as a valuable resource to assess how the model would behave on real-world inputs.

Conclusion

• Missing **Values Handling**: Categorical missing values were filled with "Unknown," and numerical missing values were imputed with -1, except for `Temperature`, which was filled with the mean value. For `Amenities`, a Random Forest model was used for imputation due to its large number of missing values.

• Error **Corrections**: Errors like negative values in the `Number of Guests` and `Water Price` columns were identified and corrected, as they were nonsensical.

• Feature **Selection**: Features like `Humidity` were removed due to weak impact, and cyclical transformations for date-related features were tested but ultimately not used.

• Model **Selection**: After hyperparameter tuning of various models, the best model achieved an R^2 of 95.58% and a low RMSE, making it the final choice for water consumption predictions.