AIR QUALITY ANALYSIS IN TAMIL NADU

INNOVATION:

The incoperate machine learning algorithms to improve the accuracy of the predictive model for analysing pollution of air quality.

Algorithm:

Step -1: Data Preprocessing.

* Collect and clean your data, handling missing values and outliers.
* Encode categorical variables and scale numerical features if necessary.
* Split your data into training, validation, and test sets.

Step – 2: Feature Engineering

* Analyze and select relevant features.
* Create new features if they can provide additional information.
* Perform feature selection to eliminate less important features.

Step – 3: Model Selection

* Choose an appropriate algorithm or ensemble of algorithms based on your problem (e.g., linear regression, decision trees, random forests, neural networks, etc.).

Step – 4: Hyperparameter Tuning

* Use techniques like grid search or random search to find the best hyperparameters for your chosen model.

Step – 5: Cross-Validation

* Implement cross-validation to assess the model's performance and ensure it generalizes well to unseen data.

Step – 6: Model Training

* Train your model on the training data using the optimal hyperparameters.

Step - 7: Evaluation

* Evaluate the model on the validation set to fine-tune and make adjustments.

Step – 8: Regularization

* + Apply regularization techniques (e.g., L1 or L2 regularization) to prevent overfitting.

Step – 9: Ensemble Methods

* + Consider ensemble methods like bagging or boosting to combine multiple models for improved accuracy.

Step – 10: Feature Importance Analysis

* + Analyze feature importances to understand which features have the most impact on predictions.

Step – 11: Error Analysis

* + Investigate the model's errors on the validation set to identify patterns and potential areas for improvement.

Step - 12: Iterate

* + Repeat steps 4 through 11, adjusting hyperparameters, feature engineering, and model selection as needed.

Step - 13: Final

Hence the output

Algorithm with source code:

# Step 1: Data Preprocessing

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.impute import SimpleImputer

# Load your dataset

data = pd.read\_csv("your\_dataset.csv")

# Handle missing values

imputer = SimpleImputer(strategy="mean")

data = imputer.fit\_transform(data)

# Encode categorical variables if needed

# scaler = StandardScaler()

# data[numeric\_features] = scaler.fit\_transform(data[numeric\_features])

# Split the data into training, validation, and test sets

X = data.drop("target\_column", axis=1)

y = data["target\_column"]

X\_train, X\_temp, y\_train, y\_temp = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

X\_val, X\_test, y\_val, y\_test = train\_test\_split(X\_temp, y\_temp, test\_size=0.5, random\_state=42)

# Step 2: Feature Engineering (You can add your own feature engineering code here)

# Step 3: Model Selection

from sklearn.ensemble import RandomForestRegressor

# Choose a model (e.g., Random Forest)

model = RandomForestRegressor()

# Step 4: Hyperparameter Tuning

from sklearn.model\_selection import GridSearchCV

param\_grid = {

"n\_estimators": [100, 200, 300],

"max\_depth": [None, 10, 20, 30],

}

grid\_search = GridSearchCV(model, param\_grid, cv=5)

grid\_search.fit(X\_train, y\_train)

best\_model = grid\_search.best\_estimator\_

# Step 5: Cross-Validation (Optional, but recommended)

from sklearn.model\_selection import cross\_val\_score

cv\_scores = cross\_val\_score(best\_model, X\_train, y\_train, cv=5)

# Step 6: Model Training

best\_model.fit(X\_train, y\_train)

# Step 7: Evaluation

validation\_score = best\_model.score(X\_val, y\_val)

# Step 8: Regularization (if needed)

# Step 9: Ensemble Methods (if needed)

# Step 10: Feature Importance Analysis

# Step 11: Error Analysis

# Step 12: Iterate as needed

# Step 13: Final Evaluation

test\_score = best\_model.score(X\_test, y\_test)

# Print final results

print("Validation Score:", validation\_score)

print("Test Score:", test\_score)