Air Pollutant Concentration Prediction over Ahmedabad Using Machine Learning

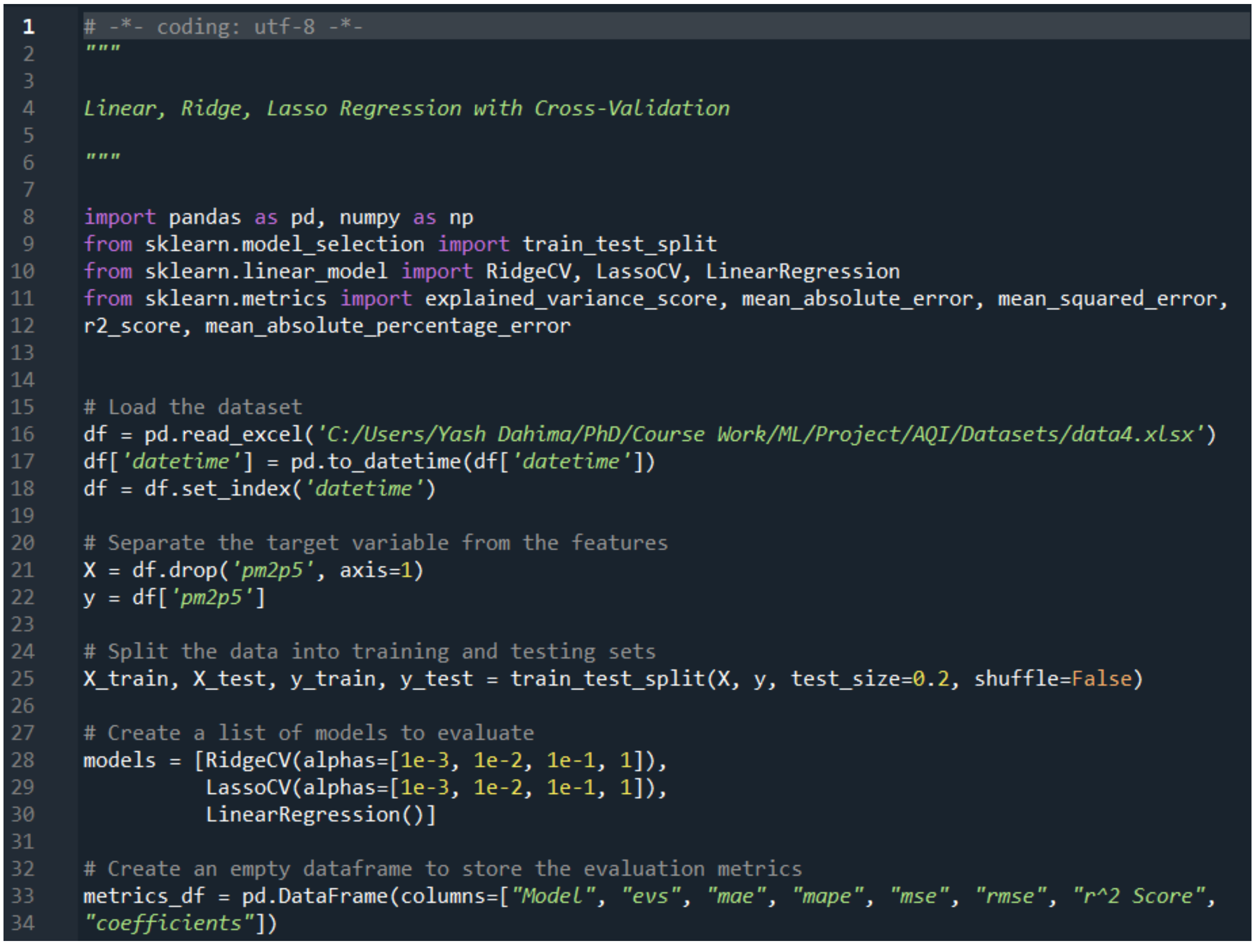
CSE523 - Machine Learning
Winter Semester 2023
Weekly Report - 1/4/2023

Yash Dahima - AU2129002

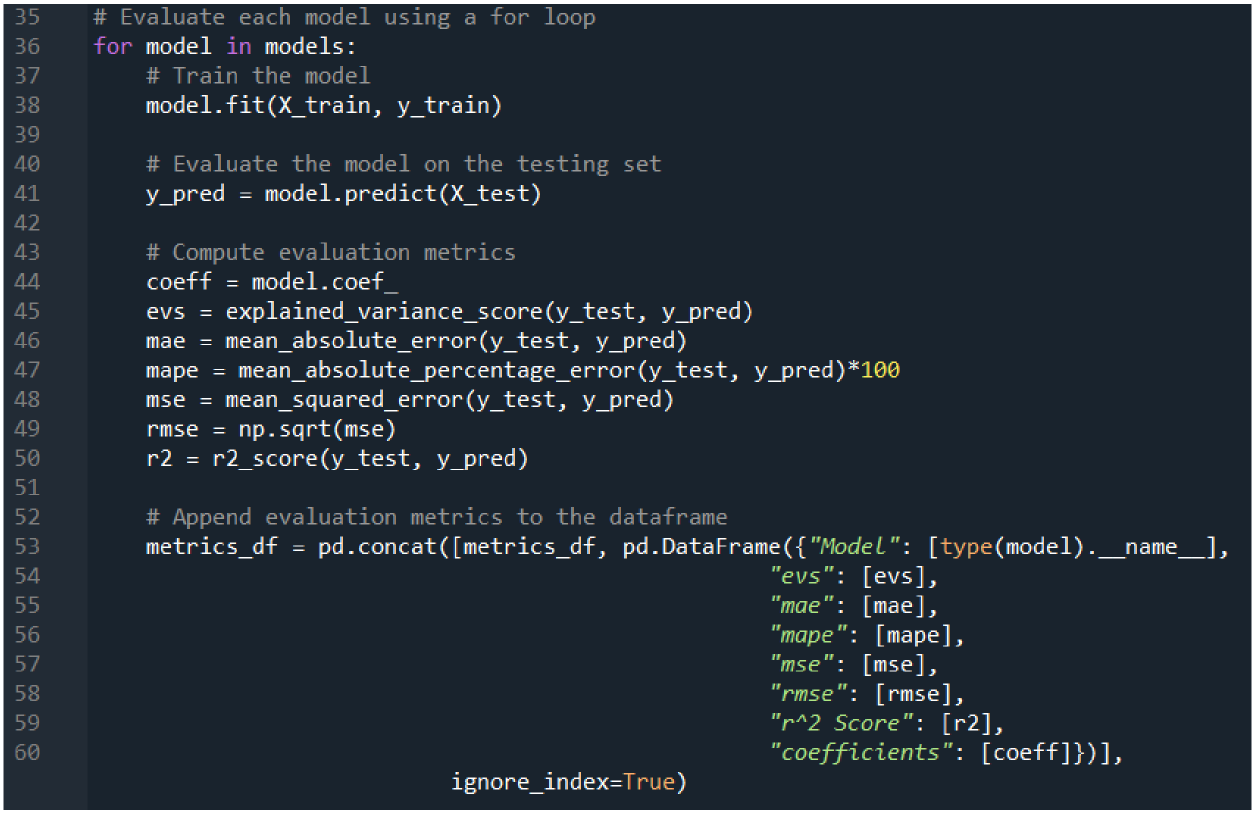
The task of ***model development*** using linear models has been performed this week.

Linear Models:

Linear Regression (OLS - Ordinary Least Square), Ridge Regression, and Lasso Regression from scikit-learn python library have been used to develop models. Their performance is evaluated using metrics such as mean absolute error, mean squared error, r2 score, etc. Models - Ridge Regression and Lasso Regression were developed with cross-validation by trying different regularizer strengths and data was fit with the best value of regularizer.



```python
# -*- coding: utf-8 -*-
"""

Linear, Ridge, Lasso Regression with Cross-Validation

"""

import pandas as pd, numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import RidgeCV, LassoCV, LinearRegression
from sklearn.metrics import explained_variance_score, mean_absolute_error, mean_squared_error,
r2_score, mean_absolute_percentage_error


# Load the dataset
df = pd.read_excel('C:/Users/Yash Dahima/PhD/Course Work/ML/Project/AQI/Datasets/data4.xlsx')
df['datetime'] = pd.to_datetime(df['datetime'])
df = df.set_index('datetime')

# Separate the target variable from the features
X = df.drop('pm2p5', axis=1)
y = df['pm2p5']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)

# Create a list of models to evaluate
models = [RidgeCV(alphas=[1e-3, 1e-2, 1e-1, 1]),
          LassoCV(alphas=[1e-3, 1e-2, 1e-1, 1]),
          LinearRegression()]

# Create an empty dataframe to store the evaluation metrics
metrics_df = pd.DataFrame(columns=["Model", "evs", "mae", "mape", "mse", "rmse", "r^2 Score",
"coefficients"])
```



```python
# Evaluate each model using a for loop
for model in models:
    # Train the model
    model.fit(X_train, y_train)

    # Evaluate the model on the testing set
    y_pred = model.predict(X_test)

    # Compute evaluation metrics
    coeff = model.coef_
    evs = explained_variance_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    mape = mean_absolute_percentage_error(y_test, y_pred)*100
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)
    r2 = r2_score(y_test, y_pred)

    # Append evaluation metrics to the dataframe
    metrics_df = pd.concat([metrics_df, pd.DataFrame({"Model": [type(model).__name__],
                                                   "evs": [evs],
                                                   "mae": [mae],
                                                   "mape": [mape],
                                                   "mse": [mse],
                                                   "rmse": [rmse],
                                                   "r^2 Score": [r2],
                                                   "coefficients": [coeff]})],
                           ignore_index=True)
```

Important model performance evaluation metrics are shown in the table below:

Models	Linear (OLS)	Lasso	Ridge
Mean Absolute Error	21.695	21.701	21.685
Mean Absolute Percentage Error	36.031	36.064	36.005
Root Mean Squared Error	28.333	28.345	28.325
R2 Score	0.560	0.559	0.560

As we can see, the model performances are not satisfactory as they are linear in nature and there is some periodicity present in the data. Hence, the models with periodic kernel functions will be developed in the next week.