CS771: Introduction to Machine Learning

Assignment 3

Sana Chaitanya, Shaijal Tripathi, Kunal Singh, Anirudh, Bhavya Gupta, Lakshmipravallika

Part 1:

Testing different Linear Models:

We had many possible approaches when it came to using linear regression models for predicting NO2 and O3 levels for the given dataset.

A list of linear regression techniques that are viable options are as follows:

1. **Simple Linear Regression**: Models the relationship between a single feature and the target variable using a straight line.

2. **Multiple Linear Regression**: Extends simple linear regression to model the relationship between multiple features and the target variable using a linear function.

3. **Ridge Regression**: A variation of multiple linear regression that includes a regularization term (L2 penalty) to reduce overfitting by shrinking the coefficients of less important features.

4. **Lasso Regression**: Similar to ridge regression, but uses an L1 penalty instead of L2, which can lead to sparse solutions (some feature coefficients become exactly zero).

5. **Elastic Net Regression**: Combines both L1 and L2 penalties, providing a balance between ridge and lasso regression. This can be particularly useful when dealing with correlated features.

6. **Support Vector Regression**: It is a type of machine learning algorithm used for regression tasks. It is based on the concept of Support Vector Machines (SVMs), which are primarily used for classification problems. The objective of SVR is to find a function that best approximates the relationship between input features and output targets, while minimizing the error between the predicted values and actual values.

We worked on the last two methods mentioned above: **Elastic net regression** (because it combines both L1 and L2 penalties) and **support vector regression** (Robustness to outliers.

We will explain the working for SVR as follows:

- The SVR algorithm is imported from the sklearn.svm module and used to predict the OZONE and NO2 levels based on the input features 'no2op1', 'no2op2', 'o3op1', and 'o3op2'. Two separate SVR models with linear kernels are trained, one for each target (OZONE and NO2 levels).

- The input features are standardized using StandardScaler from the sklearn.preprocessing module to ensure that the scale of the features does not negatively impact the performance of the model.

- After standardizing the input features, the SVR models are trained using the fit method. **The kernel parameter is set to "linear" in this case, indicating that a linear function will be used** to approximate the relationship between input features and output targets. The models are then used to make predictions on the training and testing datasets using the predict method.

By default, the SVR model in scikit-learn uses the ε-insensitive loss function and L2 regularization.

We got the following MAE values for SVR:

```
OZONE - SVR
Mean Absolute Error Train: 5.6124745632928725
Mean Absolute Error Test: 6.326993099680213


NO2 - SVR
Mean Absolute Error Train NO2: 6.106504903008368
Mean Absolute Error Test NO2: 5.858266194522602
```

Now, we describe how we used elastic net regression in our study:

- Elastic Net regression is used alongside SVR to predict the OZONE and NO2 levels based on the input features 'no2op1', 'no2op2', 'o3op1', and 'o3op2'. Elastic Net is a regularized linear regression model that combines both L1 and L2 regularization techniques. It is particularly useful when dealing with multicollinearity in the dataset or when a large number of features are present.

- Elastic Net regression is imported from the sklearn.linear_model module, and two separate Elastic Net models are trained, one for each target (OZONE and NO2 levels). The alpha parameter controls the overall regularization strength, with larger values resulting in stronger regularization. The l1_ratio parameter adjusts the balance between L1 and L2 regularization, with a value of 0.5 indicating equal weight for both types of regularization.

- The Elastic Net models are then trained using the fit method with the standardized input features and target values. Once the models are trained, they are used to make predictions on both the training and testing datasets using the predict method.

We got the following MAE values for Elastic net regression:

```
OZONE - Elastic Net
Mean Absolute Error Train: 10.402465746402443
Mean Absolute Error Test: 10.914864263172857


NO2 - Elastic Net
Mean Absolute Error Train NO2: 7.146229850304003
Mean Absolute Error Test NO2: 6.8564352193147755
```

We can clearly see that the MAE values are smaller for Linear SVR and thus, **we conclude that from our comparative study, linear SVR (with linear kernel) is the best "linear" model that we trained.**

Part 2:

Non-linear models:

We have considered the following non-linear models for this assignment:

1. **Gradient Boosting Regressor**: Gradient Boosting Regressor is an ensemble learning technique that builds a series of weak learners (typically shallow decision trees) in a stage-wise manner, optimizing the mean squared error (MSE) loss function. Each tree is trained to learn from the residuals (errors) of the previous tree, hence boosting the overall performance. It supports regularization with shrinkage (learning rate) and subsampling, which help control the model complexity and prevent overfitting.

2. **Random Forest Regressor:** Random Forest Regressor is an ensemble learning method that constructs multiple decision trees during training and averages their predictions. It uses bagging (bootstrap aggregating) and feature randomness to create diverse trees, reducing the variance of the model and improving generalization. The loss function for this regressor is also MSE, and regularization is implicitly handled by the bagging process, which reduces overfitting.

3. **Decision Tree Regressor**: A Decision Tree Regressor is a non-parametric model that recursively splits the feature space into regions based on feature values, aiming to minimize the MSE at each split. It tends to overfit the data without proper pruning or depth control. The loss function is the MSE, and regularization can be applied by controlling the tree's depth, minimum samples per leaf, or the complexity parameter (cost-complexity pruning).

4. **kNN Regressor**: k-Nearest Neighbors (kNN) Regressor is a non-parametric, lazy learning algorithm that predicts the target value by averaging the k nearest training samples' values based on a distance metric (e.g., Euclidean distance). No explicit loss function or regularization is used, but the choice of k and distance metric can impact model complexity and generalization.

5. **Polynomial Regressor**: Polynomial Regressor is a linear regression model that fits a polynomial of degree 'n' to the input data. It uses the linear regression's least squares loss function, which minimizes the sum of squared residuals. Regularization can be applied by using Ridge (L2) or Lasso (L1) techniques to penalize high-degree polynomial coefficients and prevent overfitting.

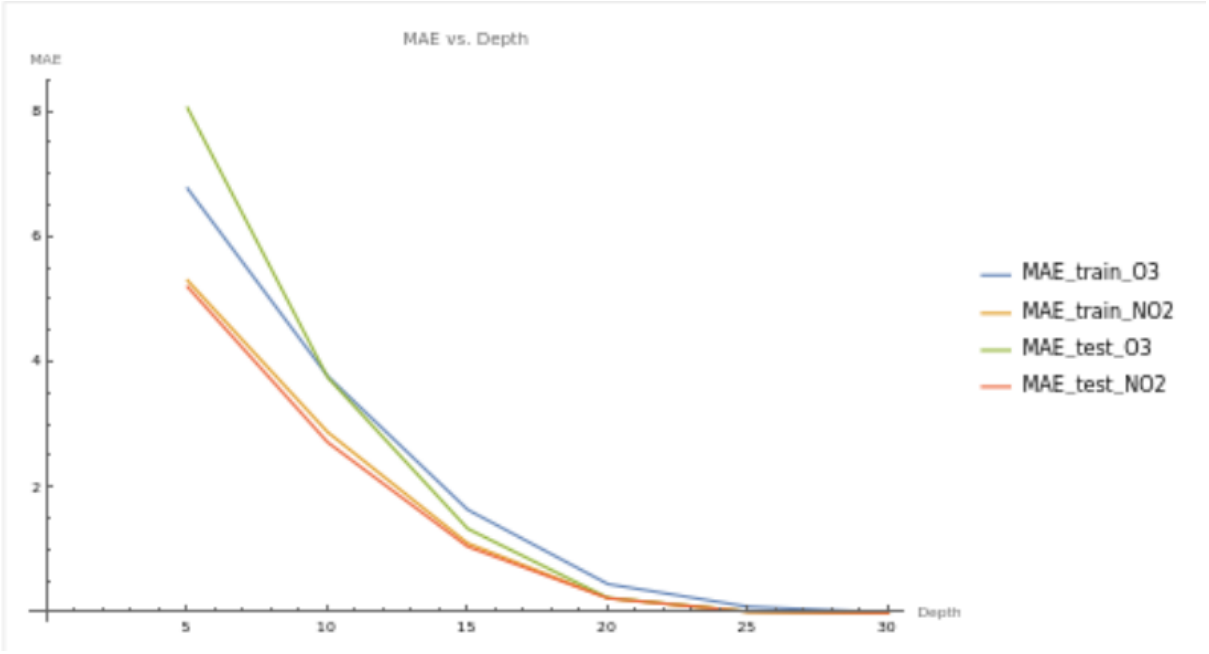
6. **SVR (kernel = 'rbf')**: Support Vector Regression (SVR) with Radial Basis Function (RBF) kernel is a kernelized version of linear regression that maps input data into a higher-dimensional space. It aims to find the best-fitting hyperplane with a maximum margin, minimizing the ε-insensitive loss function. Regularization is controlled by the hyperparameter 'C', which determines the trade-off between fitting the training data and minimizing the model complexity.

We tested different approaches and the MAE values are listed below:

Type of regressor	MAE value for O3 calibration	MAE value for NO2 calibration
Gradient Boosting	5.6369	3.9883
KNearestNeighbour (n_neighbors=35)	4.2060	2.8695
Polynomial regression: PolynomialFeatures(degree =2)	5.3111	4.4091
SVR(kernel='rbf')	5.0744	24.4027
Decision Tree (max depth = 20)	**0.2415**	**0.2349**
Random Forest (No of trees in a forest = 10)	1.4247	0.8784

If we analyze the above table, we can clearly see that the decision tree regressor with max depth 20 gives the lowest MAE and random forest regressor provides the second-lowest values.

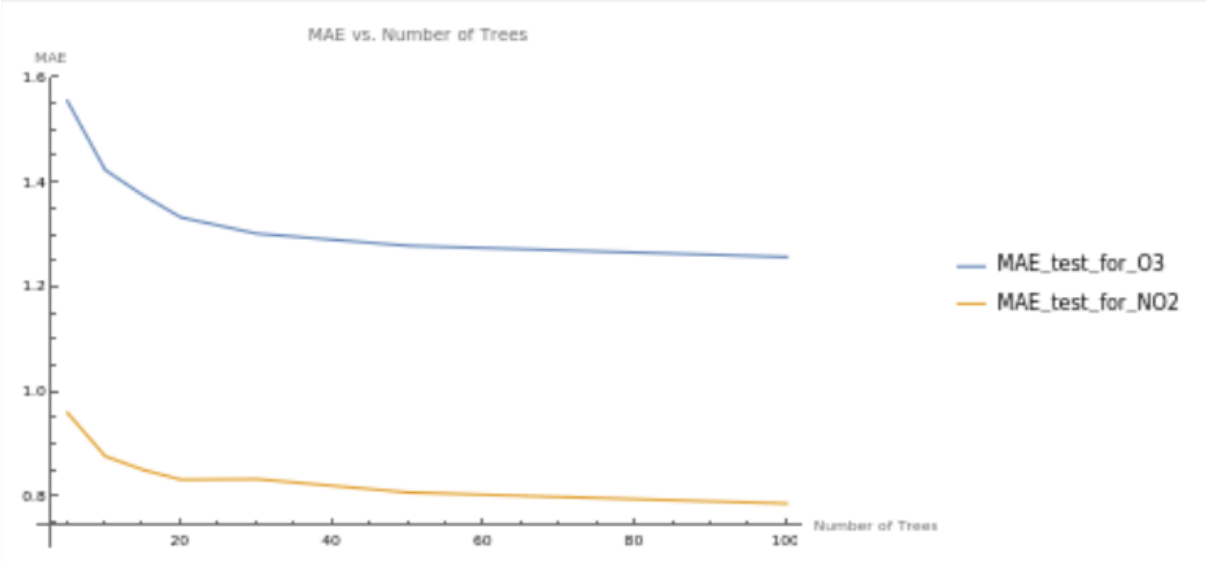
However, when we tested the decision tree model for different “max depth values”, we got the following chart:



Insights from the above chart:

- The mean MAE values for both training and testing datasets decrease significantly as the depth increases.
- The standard deviation of the MAE values also decreases with increasing depth.
- The MAE values for O3 are generally higher than those for NO2, suggesting that the model may have more difficulty predicting O3 levels compared to NO2 levels.

Now, we also **experimented with random forests** and different numbers of trees in a forest. We present our data in the form of the following graph:



Insights are as follows:

- Decreasing MAE with More Trees
- Stabilization of MAE

However, as we increased the number of trees, the training time also increased.

Conclusion:

It appears that the decision tree regressor is overfitting to the training and dummy test data as we increase the depth. Overfitting occurs when a model becomes too complex and starts capturing noise in the data rather than the underlying pattern. In this case, the model will not generalize well to unseen data, like the secret test data.

On the other hand, the random forest regressor is an ensemble method that helps reduce overfitting by averaging the predictions of multiple decision trees. This usually leads to better generalization to new data, even if the MAE on the training and dummy test data is not as low as with a single decision tree.
Our primary goal is to perform well on the secret test data, thus we use the random forest regressor as the best model for this assignment.