AML Group 24
Andrea Lopez (apl2171), Naishadh Parmar (nnp2118), Guanyu Ho (gh2646), Duru Kahyaoglu (dk2565), Daniel Chang-Dae Pak (dcp2149)

Applied Machine Learning Project Report

Introduction

Air pollution is the contamination of air due to the presence of substances that causes damage to the climate or to living organisms. The Environmental Protection Agency has identified 6 primary pollutants as criteria for air pollution, one of which is ozone. Ozone is beneficial at higher elevation in the atmosphere as it can absorb radiation from the sun, however at lower elevations, it can harm the respiratory systems of living organisms. (Ground-level Ozone Basics | US EPA).

The objective of this analysis is to create a model that can predict ozone concentration framed as a regression style problem. Following the supervised learning framework, we will perform exploratory data analysis, data preprocessing, feature extraction, and model evaluation for a set of models. By having a way to predict ozone concentrations, this can lead to changes in public policy and actions to combat air pollution.

Background of Data

The dataset used is from Clean Air Status and Trends Network (CASTNET), a national atmospheric monitoring program located across North America that records and publishes ozone and atmospheric measurements. There are many available flat files in CASTNET, however we elected to use 3 of them: site information, hourly gas measurements, and meteorological measurements. Our label ozone is contained in the meteorological measurements table. To get 1 holistic dataset, the 3 individual ones were joined on the respective primary keys/foreign keys.

Preprocessing

We combined 3 datasets from CASTNET between the years 2013 to 2022: site, hourly gas, and meteorological data. We first narrowed down to several features from the site dataset, meteorological dataset, and hourly gas dataset and then joined them on site ID and datetime where applicable. We then filtered the data to only valid measurements with a high quality assurance level. We dropped irrelevant columns (i.e. quality assurance columns), dropped highly correlated columns, one hot encoded categorical variables, dropped the datetime column and created month and year features, and scaled the data. We did structured splitting by year so that the most recent years (2021-2022) were the test data, second most recent (2019-2020) were validation data, and oldest majority of the data (2013-2018) was the training data. We were left with 22 features: temperature, relative humidity, solar radiation, precipitation, wind direction, flow rate, windspeed, shelter temperature, NO, NOY, NOYDIF, SO2_GA, latitude, longitude, elevation, month, Agric, Forest, Range, Complex, Flat, and Rolling.

Evaluation Methods/Model Performance

We chose R^2 score as our evaluation metric because it is commonly used for regression problems, clearly indicates the variance captured by the regression model, and makes it easy to compare different models.

Model	R^2 Scores of Optimal Model		
	Train Set	Validation Set	Test Set
Linear Regression	0.613	0.581	0.596
Elastic Net Regression	0.613	0.581	0.596
XGBoost	0.904	0.775	0.763
Random Forest	0.976	0.776	0.777
Multi Layer Perceptron	0.729	0.772	0.772

We used SHAP to find the most important features for each model rather than just using coefficient weights because it captures more complex and detailed insight into feature importance.

Linear Regression

A baseline linear regression model returned the following R^2 scores: 0.581 for the validation set and 0.596 for the test set. The most important features were temperature, month, and relative humidity. In order to improve upon this, we trained an elastic net regression model and tuned the following parameters using grid search: alpha with values [0.001, 0.01, 0.1, 1, 10], max iterations with values [10000, 100000] and L1 ratio with values [0.1, 0.25, 0.5, 0.75, 0.9]. The best model had an alpha of 0.01, max iterations of 10,000 and L1 ratio 0.75. Its most important features were relative humidity, month, and elevation. The R^2 scores were 0.581 for the validation set and 0.596 for the test set. These are the same scores as the baseline regression model, which shows elastic net is not any more accurate. This led to our suspicion that our data would be better captured by a nonlinear model.

Random Forest

The default random forest regressor returned the following R^2 scores: 0.986 for the training set, 0.761 for the validation set, and 0.752 for the test set. We tuned the following hyperparameters using random search: number of estimators ([10,57,105,152,200]) , max depth ([5,11,17,23,30]), min samples leaf ([1,3,5,7,10]), and max features ([sqrt, log2]). The most optimal results were achieved with the following set of hyperparameters: n_estimators = 105, min_samples_leaf = 1, max_features = log2, and max_depth =23. With these hyperparameters, we achieved an R^2 of 0.976 for the train set, 0.776 for the validation set, and 0.777 for the test set. To search for the best hyperparameters we did cross validation with time series split with one fold (i.e. one year is for training and one data for validating) and used random search to reduce run time and computational resources used.

To determine the most important features in this model, we first checked for the default feature importances and found out that the top 3 most important features were relative humidity (0.257), temperature (0.092), and month (0.082). We also tried using SHAP and Permutation for more accuracy;

however, due to the resource heavy nature of these computations and the high number of features in our model, we weren't able to run these operations.

XGBoost

The initial XGBoost model had R^2 scores as follows: 0.877 for the training set, 0.763 for the validation set, and 0.744 for the test set. Using grid search, we tuned 3 hyperparameters with the corresponding tuning sets: learning rate [0.05, 0.10, 0.15, 0.2, 0.25], number of estimators [50, 100, 150, 200, 250], and maximum depth [5, 10, 15, 20, 25]. The most optimal model was achieved with the following set of hyperparameters: n_estimators = 100, learning_rate = 0.05, and max_depth =10. Our optimal model had improved scores: 0.904 for the training set, 0.775 for the validation set, and 0.763 for the test set. To determine the most important features in the XGBoost model, we first checked for the default feature importances and found out that the top 3 most important features were elevation (0.729), relative humidity (0.099), and complex territory (0.034). We also tried SHAP for feature importance and the top 3 most important features were relative humidity, elevation, and month.

Neural Network

For the final model, we decided to use a multi layered perceptron neural network to see if it outperformed our previous models. As a neural network has a vastly larger amount of hyperparameters to tune due to the hidden layers and ways to train, a high R^2 score on the test set was the main objective while changing the hyperparameters. The final model, which can be found in our repo, had 4641 total parameters, 4381 which were trainable. This model also utilized BatchNormalization and Dropout to try and prevent overfitting, and a learning rate schedule for the Adam optimizer. When performing SHAP on this model, the top 5 features are relative humidity, month, forest, temperature, and elevation.

Conclusions

Our best performing models were the random forest regression model and the neural network. Our worst performing models were the two linear regression models. This suggests that the trends and relationships in our data are nonlinear and therefore not effectively captured by a linear model. The random forest and neural network were able to capture the nonlinear and complex relationships in our data. The most important features we saw across all models were relative humidity, month, temperature, and elevation.

Future Work

We can split different ozone concentrations into different labels depending on how harmful they are and turn this into a classification model. Additionally, rather than removing all of the rows and columns with missing data, we can incorporate feature engineering by generating more data. This way, we can further explore the significance of the features (such as gas concentrations) that were initially eliminated from our cleaned data.