

Application of Machine Learning on Air Quality

Ramya Nataraj¹ and Guillermo Goldsztein[#]

¹Green Hope High School, Cary, NC, USA

[#]Advisor

ABSTRACT

Air pollution can cause serious health problems and millions of people live in areas with poor air quality. Understanding air quality is an important topic of research which in turn helps understand health and global climate. In this paper, I applied machine learning techniques to the predict the air quality. Using a data set collected in urban cities with information on air pollutants, supervised learning was applied to gain insights and predict quality. Neural networks are used to determine the specific factors that impact the air quality index and the accuracy of the prediction. The research was helpful in identifying specific factors that affected air quality and accuracy of prediction improved when increasing hidden layers and epoch values within the neural networks.

Introduction

Global Air Quality in 2019 was responsible for 7 million premature deaths, extensive crop loss and declines in biodiversity across Europe, North America and East Asia [1]. Air pollutants also contribute directly to climate change through changes in the energy balance of the planet by some of the gaseous pollutants, notably ozone and also by particulate matter [1]. Both indoor and outdoor air quality can be overlooked and cause many long and short-term health effects on people. WHO data show that almost all of the global population (99%) breathe air that exceeds WHO guideline limits and contains high levels of pollutants, with low- and middle-income countries suffering from the highest exposures [2].

Given the vast data found about air quality and effects of different pollutants on air quality, machine learning is helpful in forecasting particulate matter in air and resultant impact to air quality. As explained in [3], "Machine learning is programming computers to optimize a performance criterion using example data or past experience. We have a model defined up to some parameters, and learning is the execution of a computer program to optimize the parameters of the model using the training data or past experience. The model may be predictive to make predictions in the future, or descriptive to gain knowledge from data, or both." As discussed in [4], "Machine learning is usually divided into two main types. In the predictive or supervised learning approach, the goal is to learn a mapping from inputs x to outputs y , given a labeled set of input-output pairs $D = \{(x_i, y_i)\}_{i=1}^N$. Here D is called the training set, and N is the number of training examples." As outlined in [5], "Machine learning is fundamentally about generalization. As an example, the standard supervised learning scenario consists of using a finite sample of labeled examples to make accurate predictions about unseen examples." Quite possibly the most important part in the machine learning process is understanding the data you are working with and how it relates to the task you want to solve [6].

Using machine learning, I will be making predictions on air quality based on data set containing information about air pollutants that affect air quality index. The use of the machine learning mechanism supervised learning is widely used. Supervised learning is the use of labeled datasets to train algorithms to classify data and predict outcomes of the data set accurately. This sub-category of machine learning, learns from the training dataset by repeatedly making predictions on the data set, training sets, and adjusting the values, validation sets, for the lowest error. Continuously monitoring air quality is important in identifying the polluted areas, levels of pollution, and the overall air quality index. There are many pollutants that impact the air quality including

particulate matter found in air (PM2.5, PM10), Carbon monoxide(CO), gas emissions from automobiles like O2, NOx, etc. Datasets that include these factors are available about air quality. When these are used in machine learning, data are classified as the features, while the resultant computation of effective Air Quality Index (AQI) is classified as the label.

Data Set

For this research, I selected a data set available in Kaggle [7] that encompasses air quality information collected from hourly and daily level of various stations across multiple cities across India. Data set included 12 features that affect air quality (including PM2.5, P10, CO, NOx, NH3, etc.), as well as details about location/city and date where the sample was collected, and resultant label that speaks to air quality index. In total the dataset had 29531 samples (rows). Each data set also had AQI values and resultant classification as a AQI bucket (whether quality is good or satisfactory). In terms of machine learning, AQI value is treated as the label in this given data set. Following table shows a sample of the data.

df

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	S02	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	Ahmedabad	2015-01-01	NaN	NaN	0.92	18.22	17.15	NaN	0.92	27.64	133.36	0.00	0.02	0.00	NaN	NaN
1	Ahmedabad	2015-01-02	NaN	NaN	0.97	15.69	16.46	NaN	0.97	24.55	34.06	3.68	5.50	3.77	NaN	NaN
2	Ahmedabad	2015-01-03	NaN	NaN	17.40	19.30	29.70	NaN	17.40	29.07	30.70	6.80	16.40	2.25	NaN	NaN
3	Ahmedabad	2015-01-04	NaN	NaN	1.70	18.48	17.97	NaN	1.70	18.59	36.08	4.43	10.14	1.00	NaN	NaN
4	Ahmedabad	2015-01-05	NaN	NaN	22.10	21.42	37.76	NaN	22.10	39.33	39.31	7.01	18.89	2.78	NaN	NaN
...
29526	Visakhapatnam	2020-06-27	15.02	50.94	7.68	25.06	19.54	12.47	0.47	8.55	23.30	2.24	12.07	0.73	41.0	Good
29527	Visakhapatnam	2020-06-28	24.38	74.09	3.42	26.06	16.53	11.99	0.52	12.72	30.14	0.74	2.21	0.38	70.0	Satisfactory
29528	Visakhapatnam	2020-06-29	22.91	65.73	3.45	29.53	18.33	10.71	0.48	8.42	30.96	0.01	0.01	0.00	68.0	Satisfactory
29529	Visakhapatnam	2020-06-30	16.64	49.97	4.05	29.26	18.80	10.03	0.52	9.84	28.30	0.00	0.00	0.00	54.0	Satisfactory
29530	Visakhapatnam	2020-07-01	15.00	66.00	0.40	26.85	14.05	5.20	0.59	2.10	17.05	NaN	NaN	NaN	50.0	Good

29531 rows x 16 columns

Applying Machine Learning

In previous research on water potability [9], we had taken specific steps to cleanse the data, determining the feature set that impact the results, and so on, Similar methodology is applied in this project to determine air quality.

Data Cleansing

Data cleaning is a critically important step in any machine learning project [8]. Using the dataset on air quality, I first examined the dataset and recognized null values for some of the features - i.e., numbers/data points do not exist for some features in certain data samples. Each feature was analyzed to determine which cells have null/empty data. In terms of programming in Python, I implemented code `df = df[df['feature']. isnull ()]` to remove empty cell and get the resultant data frame after all null cells are removed. The shape of the data frame reduced to an effective 6236 data samples (rows), with 16 columns (including 12 features that impact quality). This was deemed to be good basis to continue the analysis.

```

(df.isnull()).sum()
City      0
Date      0
PM2.5    4598
PM10     11140
NO        3582
NO2       3585
NOx       4185
NH3       10328
CO         2059
SO2       3854
O3         4022
Benzene   5623
Toluene   8041
Xylene    18109
AQI       4681
AQI_Bucket 4681
dtype: int64

df = df[-df['PM2.5'].isnull()]
df.shape
(24933, 16)

(df.isnull()).sum()
City      0
Date      0
PM2.5    0
PM10     0
NO        0
NO2       0
NOx       0
NH3       0
CO         0
SO2       0
O3         0
Benzene   0
Toluene   0
Xylene    0
AQI       0
AQI_Bucket 0
dtype: int64

```

```
df.head()
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
2123	Amaravati	2017-11-25	81.40	124.50	1.44	20.50	12.08	10.72	0.12	15.24	127.09	0.20	6.50	0.06	184.0	Moderate
2124	Amaravati	2017-11-26	78.32	129.06	1.26	26.00	14.85	10.28	0.14	26.96	117.44	0.22	7.95	0.08	197.0	Moderate
2125	Amaravati	2017-11-27	88.76	135.32	6.60	30.85	21.77	12.91	0.11	33.59	111.81	0.29	7.63	0.12	198.0	Moderate
2126	Amaravati	2017-11-28	64.18	104.09	2.56	28.07	17.01	11.42	0.09	19.00	138.18	0.17	5.02	0.07	188.0	Moderate
2127	Amaravati	2017-11-29	72.47	114.84	5.23	23.20	16.59	12.25	0.16	10.55	109.74	0.21	4.71	0.08	173.0	Moderate

Scaling Variables

Improving the accuracy of the dataset involves creating variables using different parts of the dataset. For example, the variable X is used to show the relevant features like PM10, CO, NOx, NO2, while the variable Y is set to the label of the dataset, i.e., the AQI values. Creating separate variables depicting different feature set (e.g., X_PM10 to depict PM10 values) allows for better readability of the resultant data and graphs.

Before beginning the use of the variables, the training and validation sets are created using `scaler_variable = preprocessing.StandardScaler()` line.

Training sets are data sets on which training and experimenting takes place. However, the validation set helps to improve the model performance after each epoch and provides us with the final accuracy of the model.

To scale the variable, it is necessary to flatten the y training set to a 1D array of elements to match the X training set. This allows for the model to be used accurately while also not having large differences in the range of data values used.

```
scaler_X.fit(X_train)
scaler_Y.fit(y_train.reshape(-1,1))

StandardScaler()

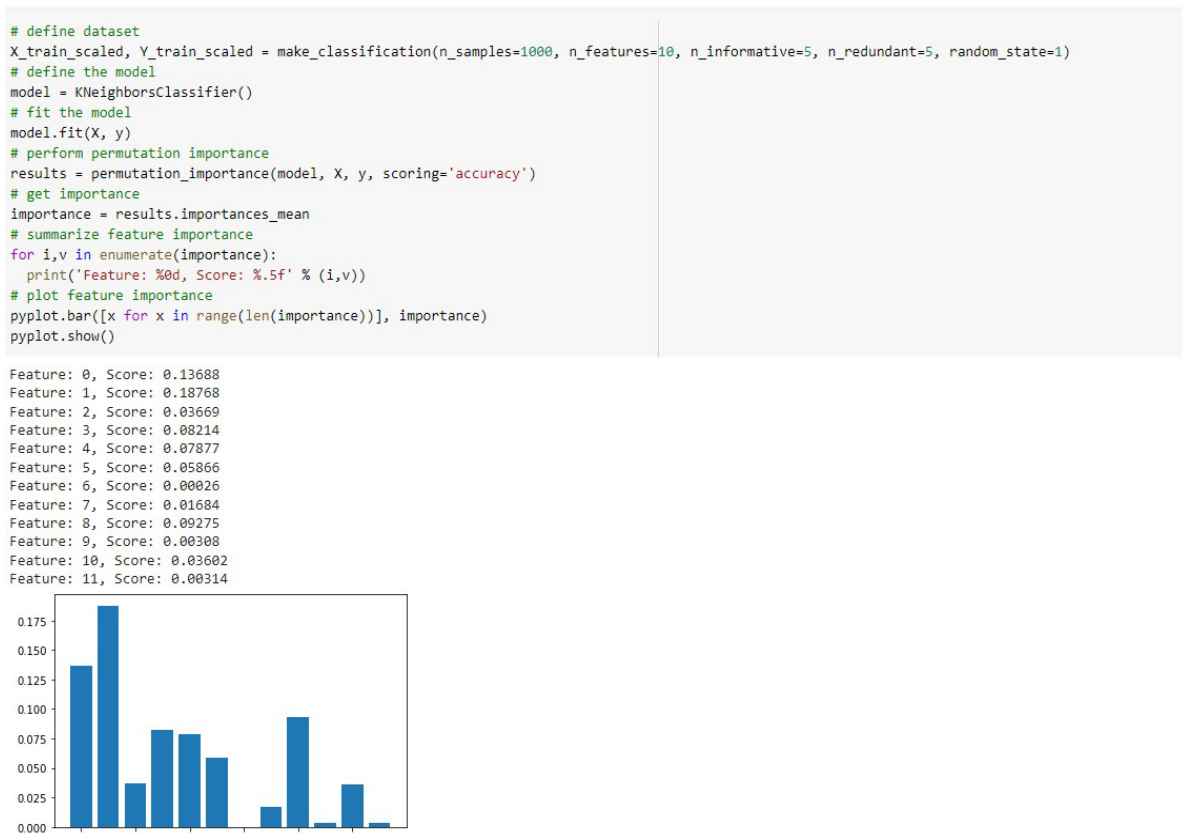
X_train_scaled = scaler_X.transform(X_train)
X_val_scaled = scaler_X.transform(X_val)

Y_train_scaled = scaler_Y.transform(y_train.reshape(-1,1))
Y_val_scaled = scaler_Y.transform(y_val.reshape(-1,1))
```

Feature Importance

To understand the ones with the highest importance, feature importance assigns a score to each feature, which allows us to predict modeling projects, and the key features selected to analyze for the air quality. The models then are created with various combinations of the features to analyze and understand the dataset more. The below image is of the code used to output the importance scores of the features, with features 1 (PM10), 0 (PM2.5) and, 8 (O3), resulting with highest scores.

Therefore, features PM10, PM2.5 and O3 are the most relevant features for the model.



Creating Predictive Models

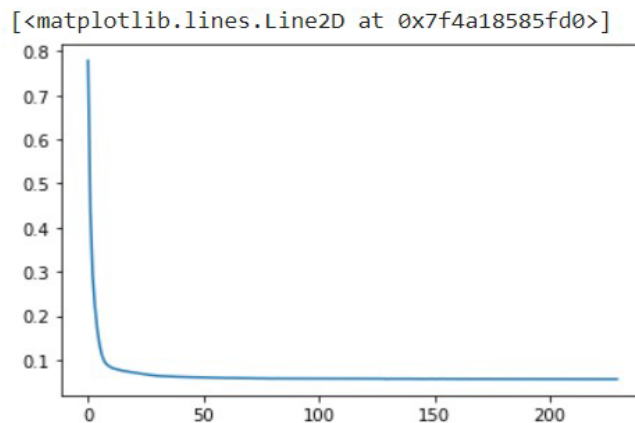
Given the importance scores the process of feature importance provided it can be modeled to understand which features are the highest impacting independent and in combination of other variables. Models are created to use one or more algorithms to predict outcomes or make decisions by trying to understand the relationship between

multiple inputs of varying type. Amongst all the models the only activation function used is 'relu'. Additionally, it is used to not allow all the neurons to activate at the same time, while also used to output the input only positively or as zero.

Using Full Data Set

The first model analyzes the full dataset, df, the data for all features and the label. Provided in the table below it determined that seven attempts had to be taken to gain the mean absolute error close to 0.15. If epochs were increased to above 230 or layers were increased to above 3,4 the model loses the linear shape it needs to maintain.

```
model = 0
model = Sequential()
model.add(Dense(4,activation='relu'))
model.add(Dense(3,activation='relu'))
model.add(Dense(1))
model.compile(loss='MSE')
model.fit(X_train_scaled,Y_train_scaled,epochs=230,verbose=0)
J = model.history.history['loss']
plt.plot(J)
```



Error Reduction

The type of errors used for linear regression models compared to binary crossentropy models are mean squared error (MSE) and mean absolute error (MBA). This is because all values within the dataset are numerical compared to binary problems that use 1 and 0 to initialize the label of the dataset. MSE represents the mean squared error which measures the average of error squares, meaning the average squared difference between the estimated values and true value. MBA represents the mean absolute error which calculates the average difference between the calculated values and actual value.

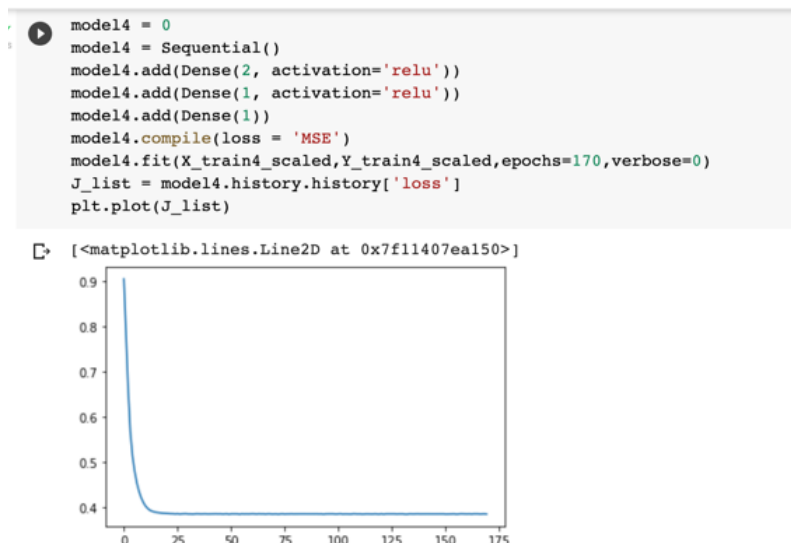
The models created to analyze the dataset uses the scaled X and Y training sets to print out the loss, or error, of the correct set of data. The MBA is changed to come close to or below 0.15, by adding different layers, hidden layers, and the number of epochs, for each of the models created. The table below shows the attempts don't with manipulating the layers and epochs per model to gain a closer MBA lower than 0.15. As seen in the table, most MBA don't reach below 0.15. Epoch of 230 seems optimal and therefore was used with rest of the experiment.

Layers	Mean Squared Error	Mean Absolute Error (try to get below 0.15)
Layer: 1 Hidden Layer: 0 Epochs: 40	Validation set: 0.08674257 Training set: 0.08578993	Validation set: 0.19738232 Training set: 0.19667935
Layer: 1 Hidden Layer: 2 Epochs: 80	Validation set: 0.06457213 Training set: 0.061836477	Validation set: 0.17337222 Training set: 0.17088147
Layer: 1 Hidden Layer: 2 Epochs: 100	Validation set: 0.06876201 Training set: 0.0627156	Validation set: 0.17751281 Training set: 0.17281567
Layer: 1 Hidden Layer: 2, 4 Epochs: 150	Validation set: 0.060155906 Training set: 0.055842843	Validation set: 0.16658312 Training set: 0.16080546
Layer: 1 Hidden Layer: 3, 4 Epochs: 220	Validation set: 0.056651536 Training set: 0.051520515	Validation set: 0.16104814 Training set: 0.15549788
Layer: 1 Hidden Layer: 3, 4 Epochs: 230	Validation set: 0.06256646 Training set: 0.053441215	Validation set: 0.16849169 Training set: 0.16099803
Begin layer table for PM10 and AQI only model		
Layer: 1 Hidden Layer: 2, 3 Epoch: 280	0.14584708 0.14922	0.24708527 0.25484946
Layer: 1 Hidden Layers: 3, 3 Epoch: 320	0.14718047 0.14786877	0.25313428 0.25871933
Attempts not recorded for other models		

Analyzing with Features with Lowest Importance Scores

Before analyzing the effect of all the features simultaneously with the highest importance score, the two lowest features (CO and Benzene) were coded in to understand the effect of them simultaneously on the AQI. This was done in model 5.

It is understood that when features with the lowest importance were played out on the AQI alone, the MSE and MBA resulting are very large, and will not come to below 0.15 as wished. Therefore, these features are ruled out of the model.



```

Y_hat_val_scaled = model4.predict(X_val4_scaled)
Y_hat_train_scaled = model4.predict(X_train4_scaled)
print('MSEerror on validation set',mse(Y_val4_scaled,Y_hat_val_scaled).numpy())
print('MSEerror on training set',mse(Y_train4_scaled,Y_hat_train_scaled).numpy())
print('MAError on validation set',mba(Y_val4_scaled,Y_hat_val_scaled).numpy())
print('MAError on training set',mba(Y_train4_scaled,Y_hat_train_scaled).numpy())

MSEerror on validation set 0.3624178
MSEerror on training set 0.3853752
MAError on validation set 0.4369407
MAError on training set 0.4484307

```

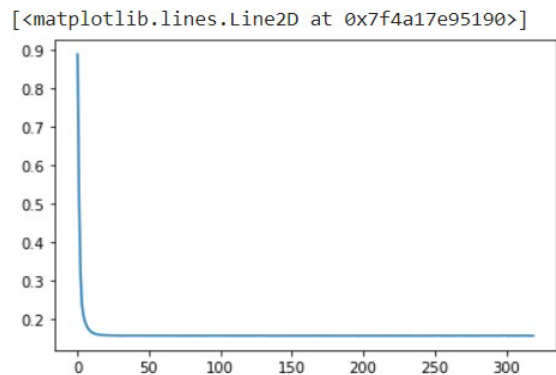
Using only PM10

Model 2 is created to analyze the effect PM10 separately on the AQI, since it gained the highest value of importance. This model continued to use the relu activation function for the layers to determine the effect of PM10 on AQI. After seven tries, with the entire df it was seen that only 3 tries were taken to get the MSE and MBA lower than 0.15. This was one less hidden layer, with 320 epochs taken in total.

```

model1 = 0
model1 = Sequential()
model1.add(Dense(3, activation='relu'))
model1.add(Dense(3, activation='relu'))
model1.add(Dense(1))
model1.compile(loss = 'MSE')
model1.fit(X_train_scale,Y_train_scale,epochs=320,verbose=0)
j_list = model1.history.history['loss']
plt.plot(j_list)

```



Using only PM2.5

Then, Model 3 is created to analyze PM2.5, the next highest affecting feature, and its effect on the AQI. PM2.5 separate resulted in either in a nonlinear model, or an MBA of values higher than 0.15, because of its feature's effect separately.

```

model2 = 0
model2 = Sequential()
model2.add(Dense(4, activation='relu'))
model2.add(Dense(4, activation='relu'))
model2.add(Dense(1))
model2.compile(loss = 'MSE')
model2.fit(X_train2_scale,Y_train2_scale,epochs=440,verbose=0)
J_list = model2.history.history['loss']
plt.plot(J_list)

```

```

Y_hat_val_scaled = model2.predict(X_val2_scale)
Y_hat_train_scaled = model2.predict(X_train2_scale)
print('MSEError on validation set',mse(Y_val2_scale,Y_hat_val_scaled).numpy())
print('MSEError on training set',mse(Y_train2_scale,Y_hat_train_scaled).numpy())
print('MAError on validation set',mba(Y_val2_scale,Y_hat_val_scaled).numpy())
print('MAError on training set',mba(Y_train2_scale,Y_hat_train_scaled).numpy())

```

```

MSEError on validation set 0.09688229
MSEError on training set 0.09397832
MAError on validation set 0.21362598
MAError on training set 0.20966184

```

Using PM10 and PM2.5

Model 4 does the analysis on the effect of PM10 and PM2.5 on AQI.

```

model3 = 0
model3 = Sequential()
model3.add(Dense(3, activation='relu'))
model3.add(Dense(2, activation='relu'))
model3.add(Dense(1))
model3.compile(loss = 'MSE')
model3.fit(X_train2_scaled,Y_train2_scaled,epochs=220,verbose=0)
J_list = model3.history.history['loss']
plt.plot(J_list)

```

```

Y_hat_val_scaled = model3.predict(X_val2_scaled)
Y_hat_train_scaled = model3.predict(X_train2_scaled)
print('MSEError on validation set',mse(Y_val2_scaled,Y_hat_val_scaled).numpy())
print('MSEError on training set',mse(Y_train2_scaled,Y_hat_train_scaled).numpy())
print('MAError on validation set',mba(Y_val2_scaled,Y_hat_val_scaled).numpy())
print('MAError on training set',mba(Y_train2_scaled,Y_hat_train_scaled).numpy())

```

```

MSEError on validation set 0.077634186
MSEError on training set 0.07163842
MAError on validation set 0.18473433
MAError on training set 0.18076372

```

Modeling with Most Relevant Features – PM10, PM2.5 And O3

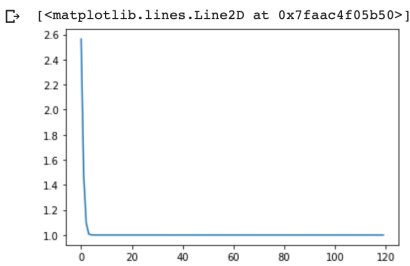
Model 6 does the analysis on the features that affected the AQI the most, e.g., the ones with the highest feature importance. This would be PM10, PM2.5, and O3.

In the air quality learning model, it is now determined that this combination of features (PM10, PM2.5 and O3) has the best impact in determining Air Quality, which is represented through AQI.

```

model5 = 0
model5 = Sequential()
model5.add(Dense(1, activation='relu'))
model5.add(Dense(1, activation='relu'))
model5.add(Dense(1))
model5.compile(loss = 'MSE')
model5.fit(X_train3_scaled,Y_train3_scaled,epochs=120,verbose=0)
J_list = model5.history.history['loss']
plt.plot(J_list)

```



```

[76] Y_hat_val_scaled = model5.predict(X_val3_scaled)
Y_hat_train_scaled = model5.predict(X_train3_scaled)
print('MSEerror on validation set',mse(Y_val3_scaled,Y_hat_val_scaled).numpy())
print('MSEerror on training set',mse(Y_train3_scaled,Y_hat_train_scaled).numpy())
print('MAError on validation set',mba(Y_val3_scaled,Y_hat_val_scaled).numpy())
print('MAError on training set',mba(Y_train3_scaled,Y_hat_train_scaled).numpy())

```

```

MSEerror on validation set 0.8866017
MSEerror on training set 1.0000124
MAError on validation set 0.71178675
MAError on training set 0.7422226

```

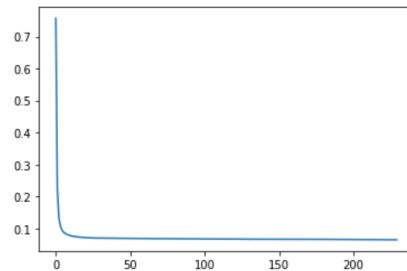
What is to be observed is that the MAError is high – 0.74. Therefore, hidden layers and epochs have to be added to improve the accuracy of the model. Adding one more hidden layer and increasing epochs from 120 to 230 helped reduce the MAError to 0.17.

```

model5 = 0
model5 = Sequential()
model5.add(Dense(4, activation='relu'))
model5.add(Dense(3, activation='relu'))
model5.add(Dense(1))
model5.compile(loss = 'MSE')
model5.fit(X_train3_scaled,Y_train3_scaled,epochs=230,verbose=0)
J_list = model5.history.history['loss']
plt.plot(J_list)

```

[matplotlib.lines.Line2D at 0x7faabf5c6490]



```

Y_hat_val_scaled = model5.predict(X_val3_scaled)
Y_hat_train_scaled = model5.predict(X_train3_scaled)
print('MSError on validation set',mse(Y_val3_scaled,Y_hat_val_scaled).numpy())
print('MSError on training set',mse(Y_train3_scaled,Y_hat_train_scaled).numpy())
print('MAError on validation set',mba(Y_val3_scaled,Y_hat_val_scaled).numpy())
print('MAError on training set',mba(Y_train3_scaled,Y_hat_train_scaled).numpy())

```

```

↳ MSError on validation set 0.07190368
MSError on training set 0.06227846
MAError on validation set 0.18124278
MAError on training set 0.1742801

```

Summary

In this research, I applied machine learning techniques to predict the air quality. Based on the data set that includes air pollutants, supervised learning programming was applied to this linear regression model. Artificial neural network mechanisms are applied to determine the significant pollutants that impact air quality and accuracy in predicting air quality. Among the twelve features in the data set, three of the features had higher scores – which pertained to PM10, PM2.5, and O3. Those were selected to train the models. Experiments were done by increasing the hidden layers and increasing the epochs to bring the mean absolute error of the models below 0.15. Accuracy was noticed when hidden layers and epochs were simultaneously changed to cause the model of only the three features to change.

Acknowledgments

I would like to sincerely thank Professor Guillermo Goldsztein from Georgia Tech for teaching me the fundamentals of machine learning, providing me with the tools to further my interest, and guiding me to research the topics of air quality and water potability.

References

[1] Fowler, D., Pyle, J.A., Sutton, M.A. and Williams, M.L., 2020. Global Air Quality, past present and future: an introduction. *Philosophical Transactions of the Royal Society A*, 378(2183), p.20190323.

- [2] Air Pollution, World Health Organization (WHO), https://www.who.int/health-topics/air-pollution#tab=tab_1
- [3] Ethem Alpaydin. 2010. Introduction to Machine Learning (2nd ed.). MIT Press.
- [4] Kevin P. Murphy, 2012, Machine Learning A Probabilistic Perspective, MIT Press.
- [5] Mehryar Mohri, Afshin Rostamizadeh and Ameet Talwalkar, 2018, Foundations of Machine Learning (2nd ed.). MIT Press.
- [6] Andreas C. Müller, Sarah Guido, 2016. Introduction to Machine Learning with Python. O'Reilly Media, Inc.
- [7] Air Quality Data in India (2015-2020), Vopani, 2020, v1, <https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india>
- [8] How to Perform Data Cleaning for Machine Learning with Python, Jason Brownie, <https://machinelearningmastery.com/basic-data-cleaning-for-machine-learning>
- [9] Ramya Nataraj, “Application of machine learning to predict water potability”, Internation Journal of Mathematics and its Applications, Vol 9 Number 4, 2021,59-64
<http://ijmaa.in/index.php/ijmaa/article/view/29>