CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning.

The name machine learning was coined in 1959 by Arthur Samuel. Tom M. Mitchell provided a widely quoted, more formal definition of the algorithms studied in the machine learning field: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." This definition of the tasks in which machine learning is concerned offers a fundamentally operational definition rather than defining the field in cognitive terms.

Machine learning is effectively a method of data analysis that works by automating the process of building data models. Machine learning in business and other fields such as healthcare and governmental departments is not simply another term for AI (Artificial Intelligence). AI is the broad term given for machines emulating human abilities while machine learning is a particular branch of AI where machines are trained to learn.

1.1.1 Application of Machine Learning

- Web Search Engine: One of the reasons why search engines like Google, Bing etc. work so well is because the system has learnt how to rank pages through a complex learning algorithm.
- Photo tagging Applications: Be it Facebook or any other photo tagging application, the ability to tag friends makes it even more happening. It

is all possible because of a face recognition algorithm that runs behind the application.

- Spam Detector: Our mail agent like Gmail or Hotmail does a lot of hard work for us in classifying the mails and moving the spam mails to spam folder. This is again achieved by a spam classifier running in the

  back end of mail application.
- Database Mining for growth of automation: Typical applications include Web-click data for better user experience, Medical records for better automation in healthcare, biological data and many more.

1.2 SUPERVISED LEARNING

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers function from labeled training data consisting of a set of training examples. In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way.

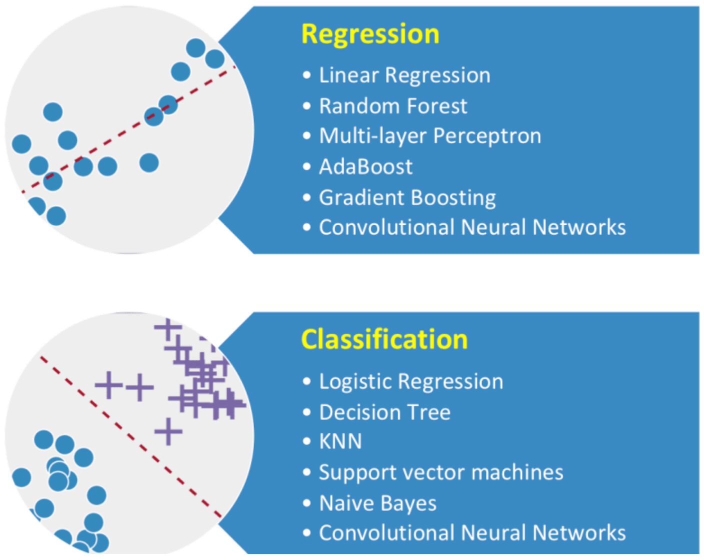


Figure 1.1 Supervised Learning

1.2.1 Regression in Supervised Learning

Regression models are used to predict a continuous value. Predicting prices of a house given the features of house like size, price etc is one of the common examples of Regression. It is a supervised technique.

In machine learning, regression algorithms attempt to estimate the mapping function (f) from the input variables (x) to numerical or continuous output variables (y).In this case, y is a real value, which can be an integer or a floating point value. Therefore, regression prediction problems are usually quantities or sizes.For example, when provided with a dataset about houses, and you are asked to predict their prices, that is a regression task because price will be a continuous output**.**

1.2.2 Classification in Supervised Learning

In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. This data set may simply be bi-class or it may be multi-class too.Some examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification etc.

On the other hand, classification algorithms attempt to estimate the mapping function (f) from the input variables (x) to discrete or categorical output variables (y). In this case , y is a category that the mapping function predicts.

CHAPTER 2

LITERATURE SURVEY

In Air Quality Prediction: Big Data and Machine Learning Approaches (2018) , Monitoring and preserving air quality has become one of the most essential activities in many industrial and urban areas today. The quality of air is adversely affected due to the various forms of pollution caused by transportation , electricity , fuel uses etc. The deposition of harmful gases is creating a serious threat for the quality of life in smart cities. With increasing air pollution , we need to implement the efficient air quality monitoring models which collect information about the concentration of air pollutants and provide assessment of air pollution in each area. Hence, air quality evaluation and prediction has become an important research area.

Recently ,many researchers began to use the big data analytics approach due to advancements in big data applications and availability of environmental sensing networks and sensor data.

In the frame of air quality monitoring of urban areas the task of short-term prediction of key-pollutants concentrations is a daily activity of major importance. Automation of this process is desirable but development of reliable predictive models with good performance to support this task in operational basis presents many difficulties.

In Atmospheric Environment(2019), As air pollution becomes more and more severe, air quality prediction has become an important approach for air pollution management and prevention. In recent years, a number of methods have been proposed to predict air quality, such as deterministic methods, statistical methods as well as machine learning methods. However, these methods have some limitations. Deterministic methods require expensive computations and specific knowledge for parameter identification, while the forecasting performance of statistical methods is limited due to the linear assumption and the multi-collinearity problem. Most of the machine learning methods, on the other hand, cannot capture the time series patterns or learn from the long-term dependencies of air pollutant concentrations. Furthermore, there is a lack of methods that could generate high prediction accuracy for air quality forecasting at larger temporal resolutions, such as daily and weekly or even monthly.

In Indian air quality predictions and analysis using machine learning(2019), We forecast the air quality of India by using machine learning to predict the air quality index of a given area. Air quality index of India is a standard measure used to indicate the pollutant levels over a period. We developed a model to predict the air quality index based on historical data of previous years and predicting over a particular upcoming year as a Gradient decent boosted multivariable regression problem. we improve the efficiency of the model by applying cost Estimation for our predictive problem.

In A Novel Method for Improving Air Pollution Prediction Based on Machine Learning Approaches:(2019), Environmental pollution has mainly been attributed to urbanization and industrial developments across the globe. Air pollution has been marked as one of the major problems of metropolitan areas around the world, especially in Tehran, the capital of Iran, where its administrators and residents have long been struggling with air pollution damage such as the health issues of its citizens. As far as the study area of this research is concerned, a considerable proportion of Tehran air pollution is attributed to PM10 and PM2.5 pollutants.

Therefore, the present study was conducted to determine the prediction models to determine air pollutions based on PM10 and PM2.5 pollution concentrations in Tehran. To predict the air-pollution, the data related to day of week, month of year, topography, meteorology, and pollutant rate of two nearest neighbors as the input parameters and machine learning methods were used. These methods include a regression support vector machine,

geographically weighted regression, artificial neural network and auto-regressive nonlinear neural network with an external input as the machine learning method for the air pollution prediction. A prediction model was then proposed to improve the afore-mentioned methods, by which the error percentage has been reduced and improved by 57%, 47%, 47% and 94%, respectively.

In Detection and Prediction of Air Pollution using Machine Learning Models(May 2018),Governments consider the regulation of air as a major task. The meteorological and traffic factors, burning of fossil fuels, industrial parameters such as power plant emissions play significant roles in air pollution. Among all the particulate matter that determine the quality of the air, Particulate matter (PM 2.5) needs more attention. When it’s level is high in the air, it causes serious issues on people’s health. Hence, controlling it by constantly keeping a check on its level in the air is important.

CHAPTER 3

PROPOSED WORK

3.1 PROPOSED METHOD

Three types of classification algorithm is used in this model. The classification algorithm is used to find the accuracy rate and best fit of the parameters. The classification algorithms used here are

- Linear Regression
- Support Vector Machine
- Decision Tree

3.2 LINEAR REGRESSION

Simple linear regression is useful for finding relationship between two continuous variables. One is predictor or independent variable and other is response or dependent variable. It looks for statistical relationship but not deterministic relationship. Relationship between two variables is said to be deterministic if one variable can be accurately expressed by the other.For example, using temperature in degree Celsius it is possible to accurately predict Fahrenheit. Statistical relationship is not accurate in determining relationship between two variables.

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

3.2.1 Hypothesis Function For Linear Regression

$$y = \theta_1 + \theta_2 . x$$

Figure 3.1 Equation of Linear Regression

While training the model it is given,

X:input training data(univariate-one input variable (parameter))

Y:Labels to data(supervised learning)

When training the model - it fits the best to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ1 and θ2 values.

θ1: intercept

θ2: coefficient of x

Once we find the best θ1 and θ2 values, we get the best fit line. So when we are finally using our model for prediction, it will predict the value of y for the input value of x.

The regression score of X_train ,y_train and the regression score of X_test,y_test is pretty much similar. The regression scores of training data and test data tells the accuracy rate of the algorithm. Based on the accuracy rate we select the algorithm that best fits our model.

3.2.3 Pros and Cons associated with Linear Regression

- Pros**:**
  - The modeling of the predictions as a weighted sum makes it transparent how predictions are produced.

- Mathematically, it is straightforward to estimate the weights and you have a guarantee to find optimal weights .

- Cons**:**

  - Linear regression models can only represent linear relationships, i.e. a weighted sum of the input features.
  - Each nonlinearity or interaction has to be hand-crafted and explicitly given to the model as an input feature.
  - Linear models are also often not that good regarding predictive **performance**, because the relationships that can be learned are so restricted and usually oversimplify how complex reality is.

3.2 SUPPORT VECTOR MACHINE

"Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges.

However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate.

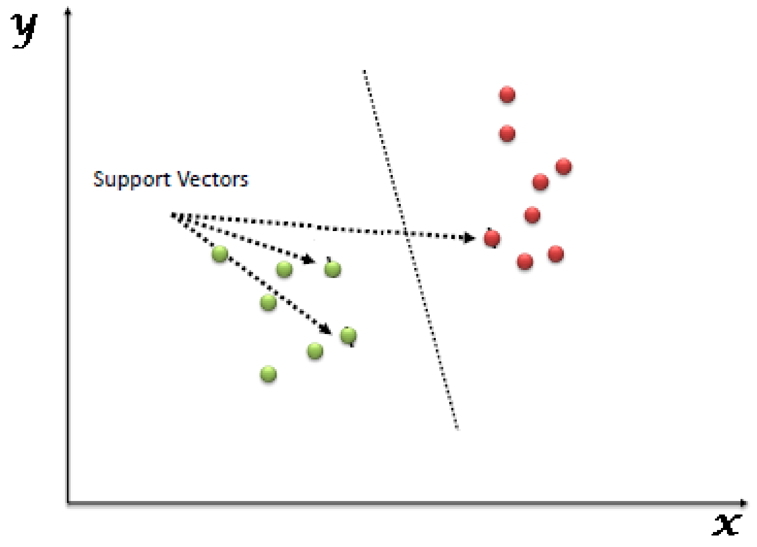


Figure 3.4 Support Vector Machine Hyperplane

Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

3.3.1 Hyperplanes and Support Vectors

Hyperplanes are decision boundaries that help classify the data points.

Data points falling on either side of the hyperplane can be attributed to

different classes. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane.

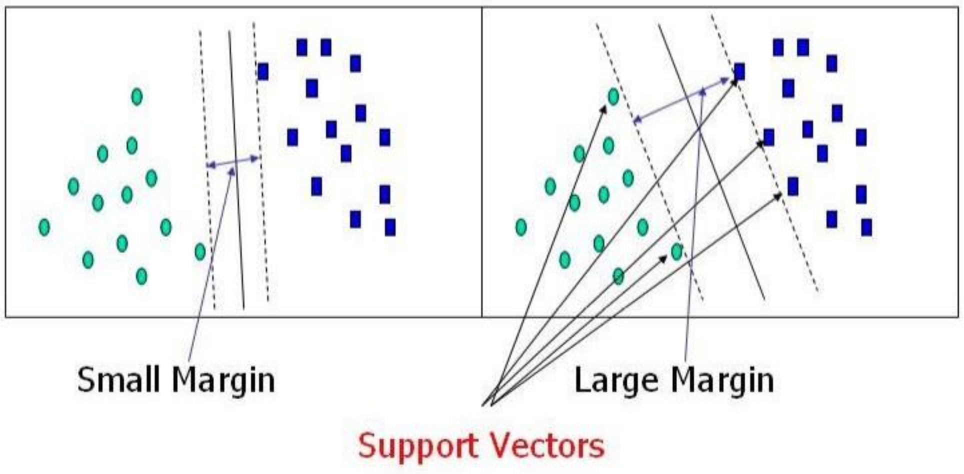


Figure 3.5 Hyperplanes and Support Vectors

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support

vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

3.3.2 Pros and Cons associated with SVM

- Pros**:**
    - It works really well with clear margin of separation.
    - It is effective in high dimensional spaces.
    - It is effective in cases where number of dimensions is greater than the number of samples.
    - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

- Cons:
    - It doesn’t perform well, when we have large data set because the required training time is higher.
    - It also doesn’t perform very well, when the data set has more noise i.e. target classes are overlapping.

3.3 DECISION TREE

Decision Tree Analysis is a general, predictive modelling tool that has applications spanning a number of different areas. In general, decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

A decision tree is a tree-like graph with nodes representing the place where we pick an attribute and ask a question; edges represent the answers the to the question; and the leaves represent the actual output or class label. They are used in non-linear decision making with simple linear decision surface.

A general algorithm for a decision tree can be described as follows:

1. Pick the best attribute/feature. The best attribute is one which best splits or separates the data.
2. Ask the relevant question.
3. Follow the answer path.
4. Go to step 1 until you arrive to the answer.

The best split is one which separates two different labels into two sets.

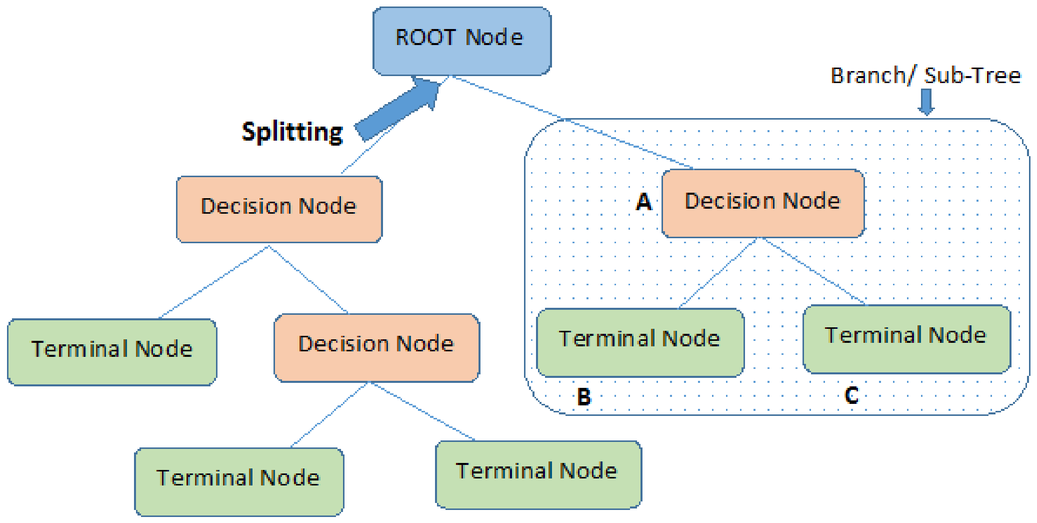


Figure 3.7 Decision Tree

3.4.1 Using Decision Tree:

- At the beginning, we consider the whole training set as the root. values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
- On the basis of attribute values records are distributed recursively.
- We use statistical methods for ordering attributes as root or the internal node.

3.6 Pros and Cons associated with Decision Tree

- Pros:
  - Simple to understand, interpret, visualize.
  - Decision trees implicitly perform variable screening or feature selection.
  - Can handle both numerical and categorical data. Can also handle multi-output problems.

- Cons
  - Decision-tree learners can create over-complex trees that do not generalize the data well. This is called overfitting.
  - Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This is called variance, which needs to be lowered by methods like bagging and boosting.
  - Greedy algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees, where the features and samples are randomly sampled with replacement.
  - Decision tree learners create biased trees if some classes dominate.

CHAPTER 4

SYSTEM REQUIREMENT

4.1 ANACONDA INSTALLATION ON WINDOWS

Anaconda is a package manager, an environment manager, and Python distribution that contains a collection of many open source packages. This is advantageous as when you are working on a data science project, you will find that you need many different packages (numpy, scikit-learn, scipy, pandas to name a few), which an installation of Anaconda comes preinstalled with.

1. Go to the Anaconda Website and choose a Python 3.x graphical installer (A) or a Python 2.x graphical installer (B). If you aren't sure which Python version you want to install, choose Python 3. Do not choose both.

2. Locate your download and double click it.

3. click on Next.Read the license agreement and click on I Agree.

4. Click on Next.Note your installation location and then click Next.

5. This is an important part of the installation process. The recommended approach is to not check the box to add Anaconda to your path. This means you will have to use Anaconda Navigator or the Anaconda Command Prompt (located in the Start Menu under "Anaconda") when you wish to use Anaconda (you can always add Anaconda to your PATH later if you don't check the box). If you want to be able to use Anaconda in your command prompt (or git bash, cmder, powershell etc), please use the alternative approach and check the box.

6. Click on Finish.

4.1.1 Hardware requirements

- CPU: 2 x 64-bit 2.8 GHz 8.00 GT/s CPUs
- RAM: 32 GB (or 16 GB of 1600 MHz DDR3 RAM)
- Storage: 300 GB.
- Internet access to download the files from Anaconda Cloud or a USB drive containing all of the files you need with alternate instructions .

4.1.2 Software & system requirements

- Ubuntu users may need to install cURL.
- Client environment may be Windows, macOS or Linux
- MongoDB 2.6 (provided)
- Anaconda Repository license file

4.2 JUPYTER NOTEBOOK

The Jupyter Notebook application allows you to create and edit documents that display the input and output of a Python or R language script. Once saved, you can share these files with others.

While Jupyter runs code in many programming languages, Python is a requirement (Python 3.3 or greater, or Python 2.7) for installing the

JupyterLab or the classic Jupyter Notebook.

Jupyter Notebook is a web application that allows you to create and share documents that contain:

- live code (e.g. Python code)

- visualizations
- explanatory text (written in markdown syntax)

4.2.1 Launching Jupyter Notebook

The Jupyter Notebook App can be launched by clicking on the Jupyter Notebook icon installed by Anaconda in the start menu (Windows) or by typing in a terminal (cmd on Windows):

- Launch the Jupyter Notebook App (see previous section).
- In the Notebook Dashboard navigate to find the notebook: clicking on its name will open it in a new browser tab.
- Click on the menu Help -> User Interface Tour for an overview of the Jupyter Notebook App user interface.
- You can run the notebook document step-by-step (one cell a time) by pressing shift + enter.
- You can run the whole notebook in a single step by clicking on the menu Cell -> Run All.
- To restart the kernel (i.e. the computational engine), click on the menu Kernel -> Restart. This can be useful to start over a computation from scratch (e.g. variables are deleted, open files are closed, etc…).

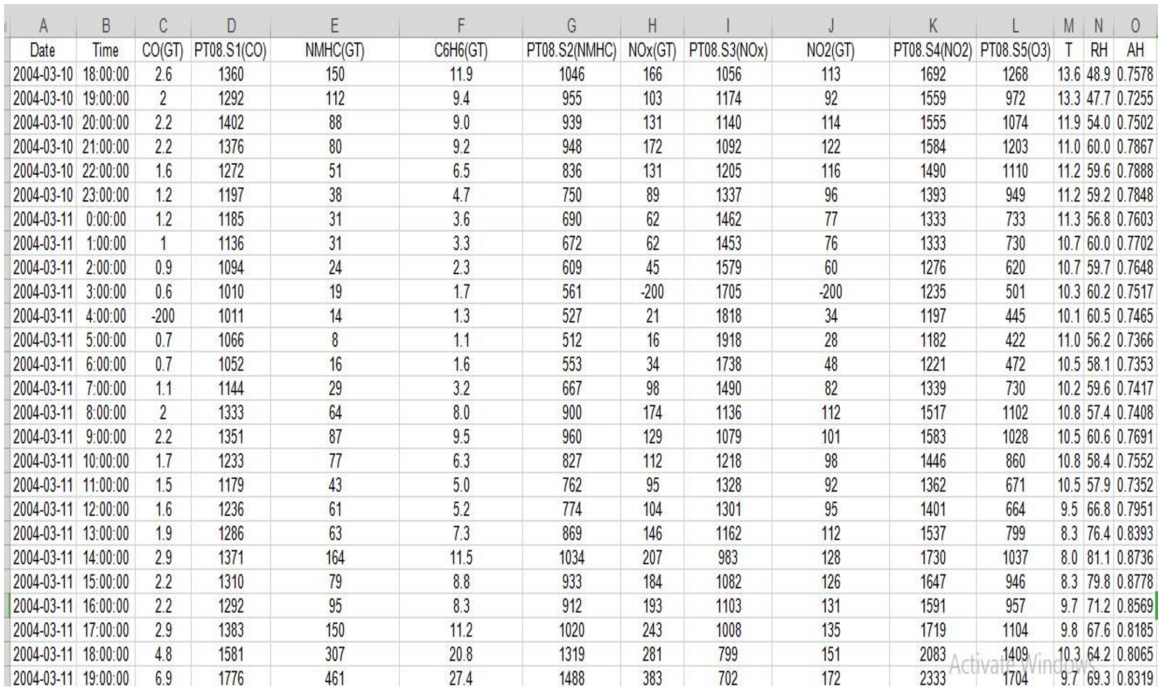
CHAPTER 5

IMPLEMENTATION MODULES

5.1 DATA DESCRIPTION

This dataset contains the responses of a gas multisensor device deployed on the field in an Italian city. Hourly responses averages are recorded along with gas concentrations references from a certified analyzer. The dataset contains 9358 instances of hourly averaged responses from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device. The device was located on the field in a significantly polluted area, at road level, within an Italian city. Data was recorded from March 2004 to February 2005 (one year) representing the longest freely available recordings of on field deployed air quality chemical sensor devices responses. Ground Truth hourly averaged concentrations for CO, Non Metanic Hydrocarbons, Benzene, Total Nitrogen Oxides (NOx) and Nitrogen Dioxide (NO2) are provided by a co-located reference certified analyzer.

Figure 5.1 shows the dataset in excel format . Training data and testing data are separated. Training data is used to train the model and testing data is used for testing the model .

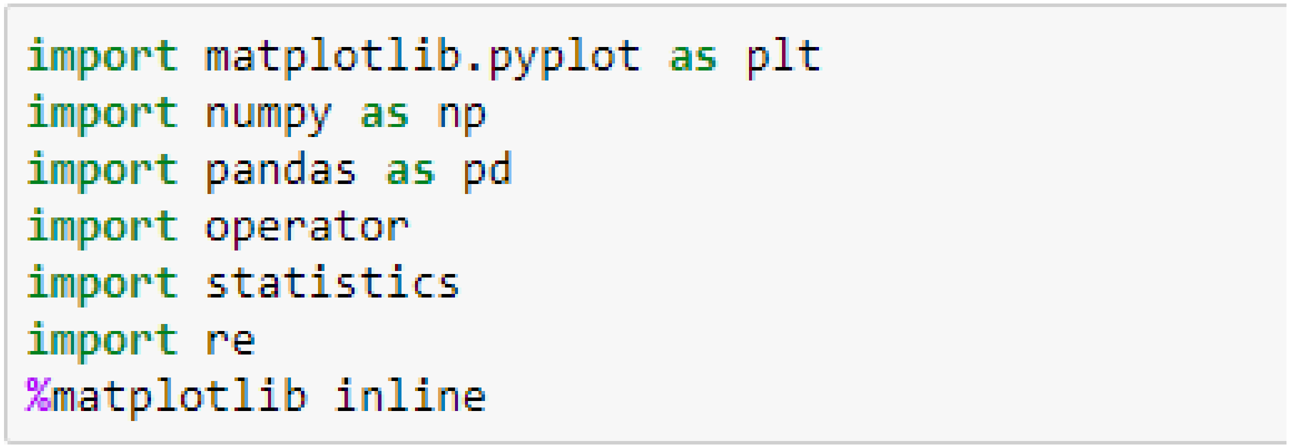


A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	T	RH	AH
2004-03-10	18:00:00	2.6	1360	150	11.9	1046	166	1056	113	1692	1268	13.6	48.9	0.7578
2004-03-10	19:00:00	2	1292	112	9.4	955	103	1174	92	1559	972	13.3	47.7	0.7255
2004-03-10	20:00:00	2.2	1402	88	9.0	939	131	1140	114	1555	1074	11.9	54.0	0.7502
2004-03-10	21:00:00	2.2	1376	80	9.2	948	172	1092	122	1584	1203	11.0	60.0	0.7867
2004-03-10	22:00:00	1.6	1272	51	6.5	836	131	1205	116	1490	1110	11.2	59.6	0.7888
2004-03-10	23:00:00	1.2	1197	38	4.7	750	89	1337	96	1393	949	11.2	59.2	0.7848
2004-03-11	0:00:00	1.2	1185	31	3.6	690	62	1462	77	1333	733	11.3	56.8	0.7603
2004-03-11	1:00:00	1	1136	31	3.3	672	62	1453	76	1333	730	10.7	60.0	0.7702
2004-03-11	2:00:00	0.9	1094	24	2.3	609	45	1579	60	1276	620	10.7	59.7	0.7648
2004-03-11	3:00:00	0.6	1010	19	1.7	561	-200	1705	-200	1235	501	10.3	60.2	0.7517
2004-03-11	4:00:00	-200	1011	14	1.3	527	21	1818	34	1197	445	10.1	60.5	0.7465
2004-03-11	5:00:00	0.7	1066	8	1.1	512	16	1918	28	1182	422	11.0	56.2	0.7366
2004-03-11	6:00:00	0.7	1052	16	1.6	553	34	1738	48	1221	472	10.5	58.1	0.7353
2004-03-11	7:00:00	1.1	1144	29	3.2	667	98	1490	82	1339	730	10.2	59.6	0.7417
2004-03-11	8:00:00	2	1333	64	8.0	900	174	1136	112	1517	1102	10.8	57.4	0.7408
2004-03-11	9:00:00	2.2	1351	87	9.5	960	129	1079	101	1583	1028	10.5	60.6	0.7691
2004-03-11	10:00:00	1.7	1233	77	6.3	827	112	1218	98	1446	860	10.8	58.4	0.7552
2004-03-11	11:00:00	1.5	1179	43	5.0	762	95	1328	92	1362	671	10.5	57.9	0.7352
2004-03-11	12:00:00	1.6	1236	61	5.2	774	104	1301	95	1401	664	9.5	66.8	0.7951
2004-03-11	13:00:00	1.9	1286	63	7.3	869	146	1162	112	1537	799	8.3	76.4	0.8393
2004-03-11	14:00:00	2.9	1371	164	11.5	1034	207	983	128	1730	1037	8.0	81.1	0.8736
2004-03-11	15:00:00	2.2	1310	79	8.8	933	184	1082	126	1647	946	8.3	79.8	0.8778
2004-03-11	16:00:00	2.2	1292	95	8.3	912	193	1103	131	1591	957	9.7	71.2	0.8569
2004-03-11	17:00:00	2.9	1383	150	11.2	1020	243	1008	135	1719	1104	9.8	67.6	0.8185
2004-03-11	18:00:00	4.8	1581	307	20.8	1319	281	799	151	2083	1409	10.3	64.2	0.8065
2004-03-11	19:00:00	6.9	1776	461	27.4	1488	383	702	172	2333	1704	9.7	69.3	0.8319

Figure 5.1 Dataset in Excel Sheet

5.2 IMPORTING PACKAGES AND PACKAGES

In programming, a module is a piece of software that has a specific functionality. For example, when building a ping pong game, one module would be responsible for the game logic, and another module would be responsible for drawing the game on the screen. Each module is a different file, which can be edited separately.



```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import operator
import statistics
import re
%matplotlib inline
```

Figure 5.2 Importing Packages

5.2.1 Matplotlib

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.

One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations. They’re typically instruments for reasoning about quantitative information.

5.2.2 Numpy

NumPy is a Python package which stands for ‘Numerical Python’. It is the core library for scientific computing, which contains a powerful n-dimensional array object, provide tools for integrating C, C++ etc. It is also useful in linear algebra, random number capability etc. NumPy array can also be used as an efficient multi-dimensional container for generic data.

Numpy array is a powerful N-dimensional array object which is in the form of rows and columns. We can initialize numpy arrays from nested Python lists and access it elements.

5.2.3 Pandas

Pandas is the most popular python library that is used for data analysis. It provides highly optimized performance with back-end source code is purely written in C or Python.

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.

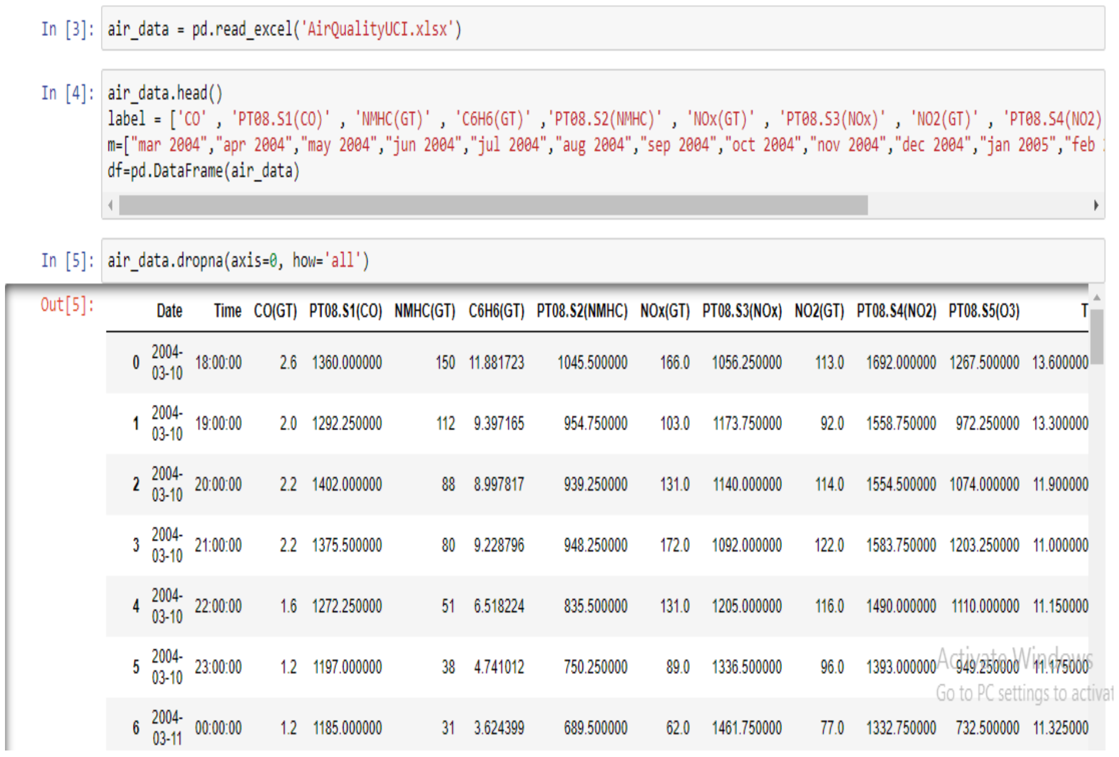
5.2.4 Regular Expression

Regular expressions use the backslash character ('\') to indicate special forms or to allow special characters to be used without invoking their special meaning. This collides with Python's usage of the same character for the same purpose in string literals; for example, to match a literal backslash, one might have to write '\\\\' as the pattern string, because the regular expression must be \\, and each backslash must be expressed as \\ inside a regular Python string literal. Also, please note that any invalid escape sequences in Python's usage of the backslash in string literals now generate a DeprecationWarning and in the future this will become a SyntaxError. This behaviour will happen even if it is a valid escape sequence for a regular expression.

5.3 LOADING DATASETS

Data was recorded from March 2004 to February 2005 (one year) representing the longest freely available recordings of on field deployed air quality chemical sensor devices responses. Ground Truth hourly averaged concentrations for CO, Non Metanic Hydrocarbons, Benzene, Total Nitrogen Oxides (NOx) and Nitrogen Dioxide (NO2) are provided by a co-located reference certified analyzer.

The name of the dataset used here is AirQualityUCI. The dataset contains 9358 instances of hourly averaged responses from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device.



```
In [3]: air_data = pd.read_excel('AirQualityUCI.xlsx')

In [4]: air_data.head()
        label = ['CO' , 'PT08.S1(CO)' , 'NMHC(GT)' , 'C6H6(GT)' ,'PT08.S2(NMHC)' , 'NOx(GT)' , 'PT08.S3(NOx)' , 'NO2(GT)' , 'PT08.S4(NO2)
        m=["mar 2004","apr 2004","may 2004","jun 2004","jul 2004","aug 2004","sep 2004","oct 2004","nov 2004","dec 2004","jan 2005","feb 
        df=pd.DataFrame(air_data)

In [5]: air_data.dropna(axis=0, how='all')

Out[5]:
       Date      Time  CO(GT)  PT08.S1(CO)  NMHC(GT)  C6H6(GT)  PT08.S2(NMHC)  NOx(GT)  PT08.S3(NOx)  NO2(GT)  PT08.S4(NO2)  PT08.S5(O3)          T
   0  2004-  18:00:00     2.6  1360.000000       150  11.881723    1045.500000    166.0   1056.250000    113.0   1692.000000  1267.500000  13.600000
      03-10
   1  2004-  19:00:00     2.0  1292.250000       112   9.397165     954.750000    103.0   1173.750000     92.0   1558.750000   972.250000  13.300000
      03-10
   2  2004-  20:00:00     2.2  1402.000000        88   8.997817     939.250000    131.0   1140.000000    114.0   1554.500000  1074.000000  11.900000
      03-10
   3  2004-  21:00:00     2.2  1375.500000        80   9.228796     948.250000    172.0   1092.000000    122.0   1583.750000  1203.250000  11.000000
      03-10
   4  2004-  22:00:00     1.6  1272.250000        51   6.518224     835.500000    131.0   1205.000000    116.0   1490.000000  1110.000000  11.150000
      03-10
   5  2004-  23:00:00     1.2  1197.000000        38   4.741012     750.250000     89.0   1336.500000     96.0   1393.000000   949.250000  11.175000
      03-10
   6  2004-  00:00:00     1.2  1185.000000        31   3.624399     689.500000     62.0   1461.750000     77.0   1332.750000   732.500000  11.325000
      03-11
```

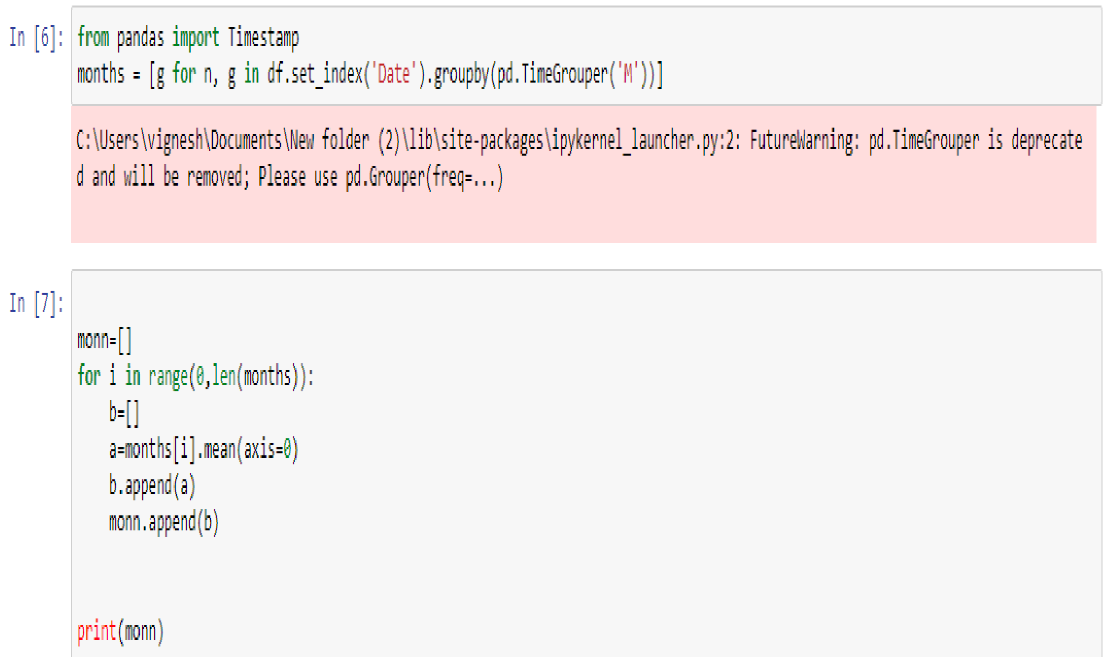
Figure 5.3 Loaded Dataset

5.3.1 Dropna( )

The dropna() function is used to remove a row or a column from a dataframe which has a NaN or no values in it. Pandas dropna() method allows the user to analyze and drop Rows/Columnswith Null values in different ways. subset: It&#39;s an array which limits the dropping process to passed

rows/columns through list. inplace: It is a boolean which makes the changes in data frame itself if True.

5.4 SEGREGATING DATSET



```
In [6]: from pandas import Timestamp
        months = [g for n, g in df.set_index('Date').groupby(pd.TimeGrouper('M'))]

C:\Users\vignesh\Documents\New folder (2)\lib\site-packages\ipykernel_launcher.py:2: FutureWarning: pd.TimeGrouper is deprecate
d and will be removed; Please use pd.Grouper(freq=...)

In [7]: 
        monn=[]
        for i in range(0,len(months)):
            b=[]
            a=months[i].mean(axis=0)
            b.append(a)
            monn.append(b)


        print(monn)
```

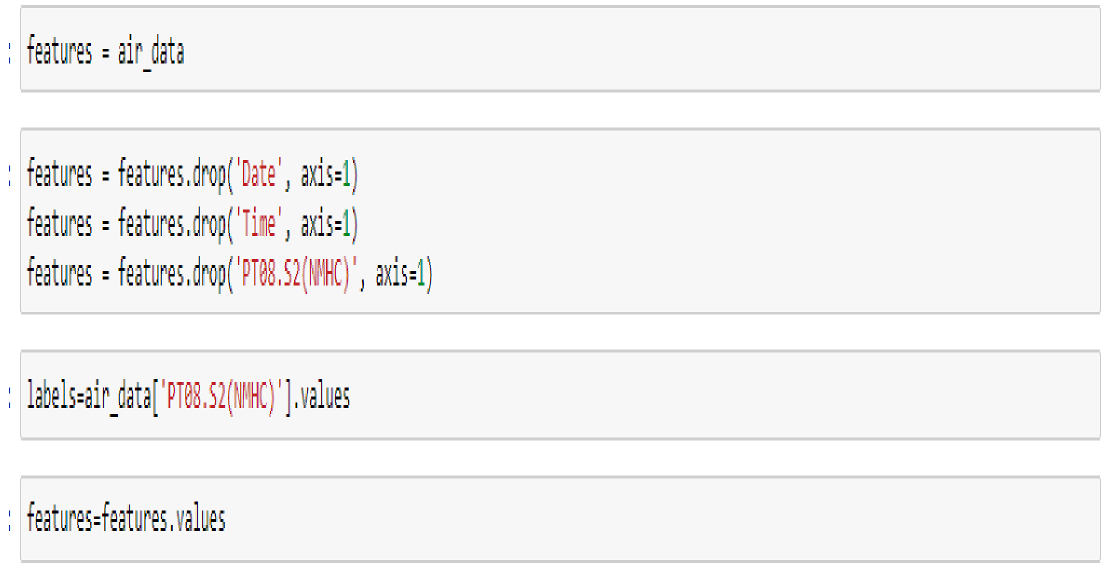
Figure 5.4 Segregated Dataset

5.4.1 Timestamp

Timestamp is the pandas equivalent of python’s Datetime and is interchangeable with it in most cases. It’s the type used for the entries that make up a DatetimeIndex, and other timeseries oriented data structures in pandas.

5.5 FEATURES AND LABELS

Defining our features and ignore those that might not be of help in our prediction. For example, date is not a very useful feature that can assist in predicting the future values.



```
features = air_data
```

```
features = features.drop('Date', axis=1)
features = features.drop('Time', axis=1)
features = features.drop('PT08.S2(NMHC)', axis=1)
```

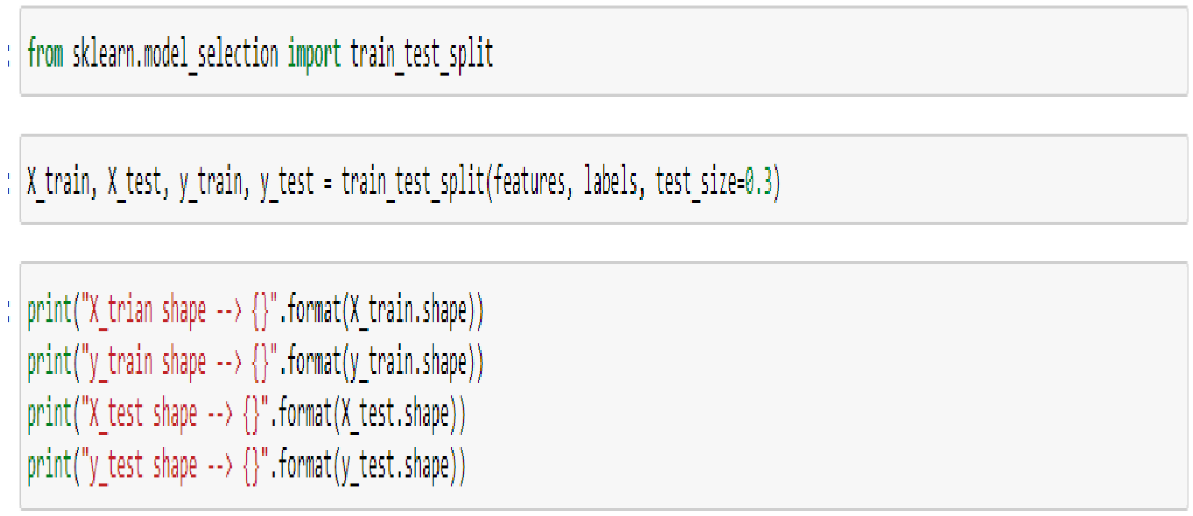
```
labels=air_data['PT08.S2(NMHC)'].values
```

```
features=features.values
```

Figure 5.5 Features and Labels

5.5 TRAIN AND SPLIT

Working with datasets, a **machine learning algorithm** works in two stages. Usually split the data around 20%-80% between testing and training stages. Under supervised learning, I split a dataset into a training data and test data in Python ML.



```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.3)
```

```
print("X_trian shape --> {}".format(X_train.shape))
print("y_train shape --> {}".format(y_train.shape))
print("X_test shape --> {}".format(X_test.shape))
print("y_test shape --> {}".format(y_test.shape))
```

Figure 5.6 Train and Test Data

Here the model is trained with 70% of the data from the dataset and 30% of the data is used for testing.

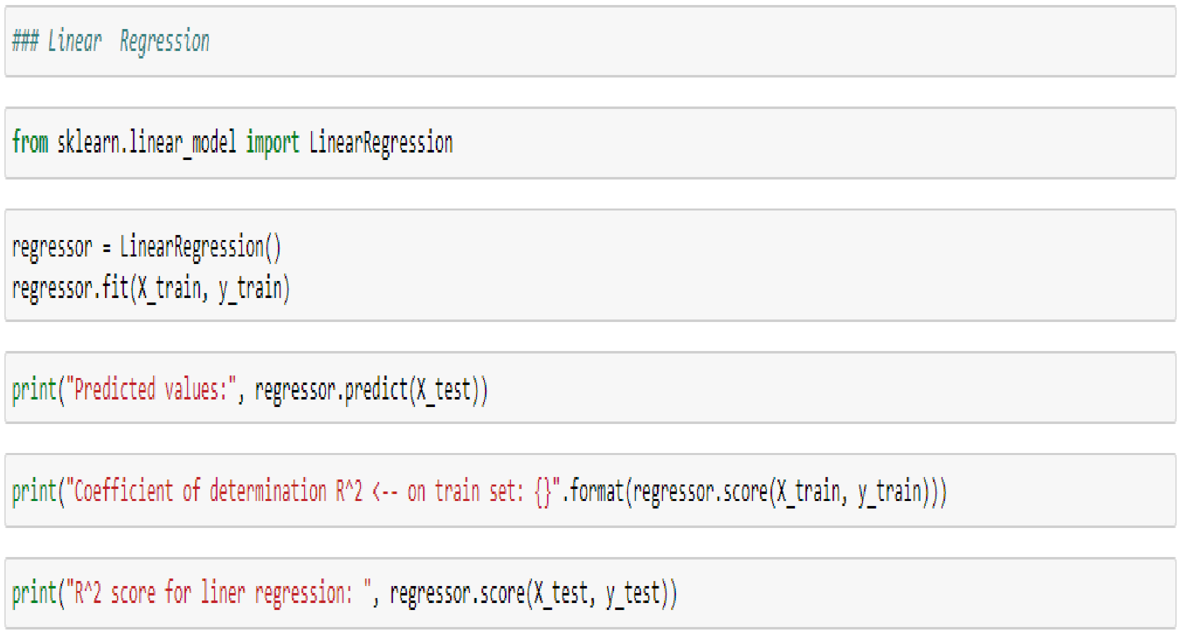
5.7 PREDICTING ACCURACY

Here we predict the accuracy of the model using three main algorithms.

- Linear Regression
- Support Vector Machine
- Decision Tree

5.7.1 Linear Regression

Simple linear regression is useful for finding relationship between two continuous variables. One is predictor or independent variable and other is response or dependent variable.



```
### Linear  Regression
```

```
from sklearn.linear_model import LinearRegression
```

```
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
print("Predicted values:", regressor.predict(X_test))
```

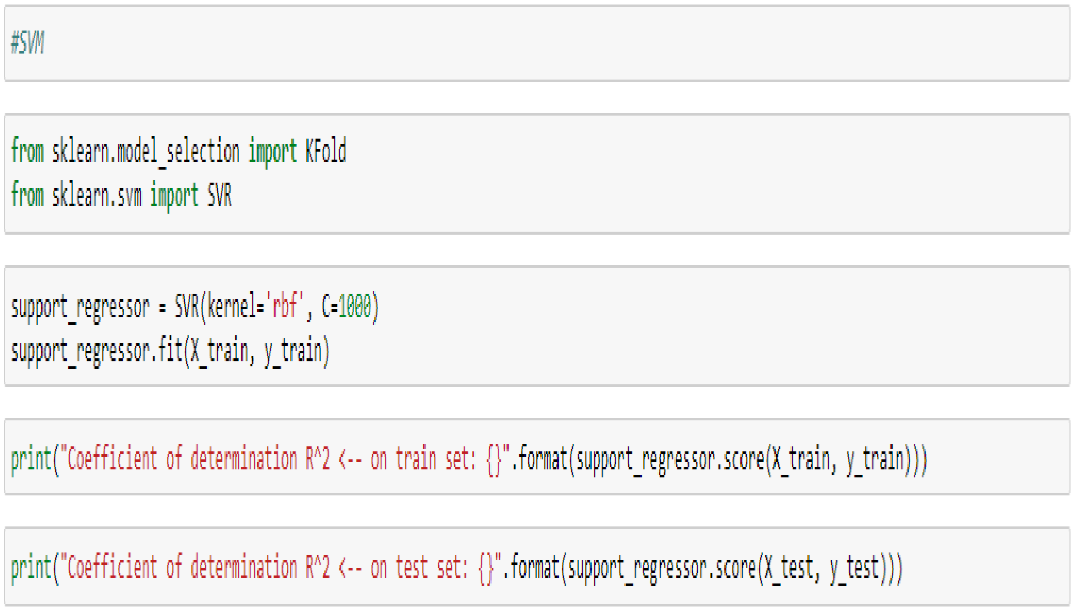
```
print("Coefficient of determination R^2 <-- on train set: {}".format(regressor.score(X_train, y_train)))
```

```
print("R^2 score for liner regression: ", regressor.score(X_test, y_test))
```

Figure 5.7 Linear Regression

5.7.2 Support Vector Machine

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.



```
#SVM
```

```
from sklearn.model_selection import KFold
from sklearn.svm import SVR
```

```
support_regressor = SVR(kernel='rbf', C=1000)
support_regressor.fit(X_train, y_train)
```

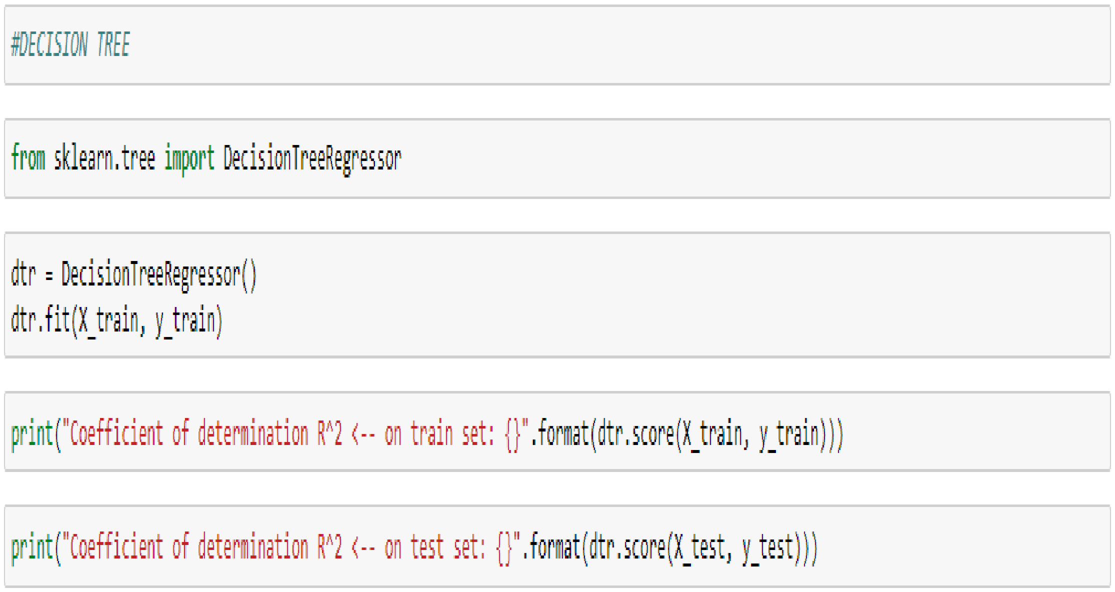
```
print("Coefficient of determination R^2 <-- on train set: {}".format(support_regressor.score(X_train, y_train)))
```

```
print("Coefficient of determination R^2 <-- on test set: {}".format(support_regressor.score(X_test, y_test)))
```

5.8 Support Vector Machine

5.7.3 Decision Tree

The general motive of using Decision Tree is to create a training model which can use to predict class or value of target variables by learning decision rules inferred from prior data(training data).



```
#DECISION TREE
```

```
from sklearn.tree import DecisionTreeRegressor
```

```
dtr = DecisionTreeRegressor()
dtr.fit(X_train, y_train)
```

```
print("Coefficient of determination R^2 <-- on train set: {}".format(dtr.score(X_train, y_train)))
```

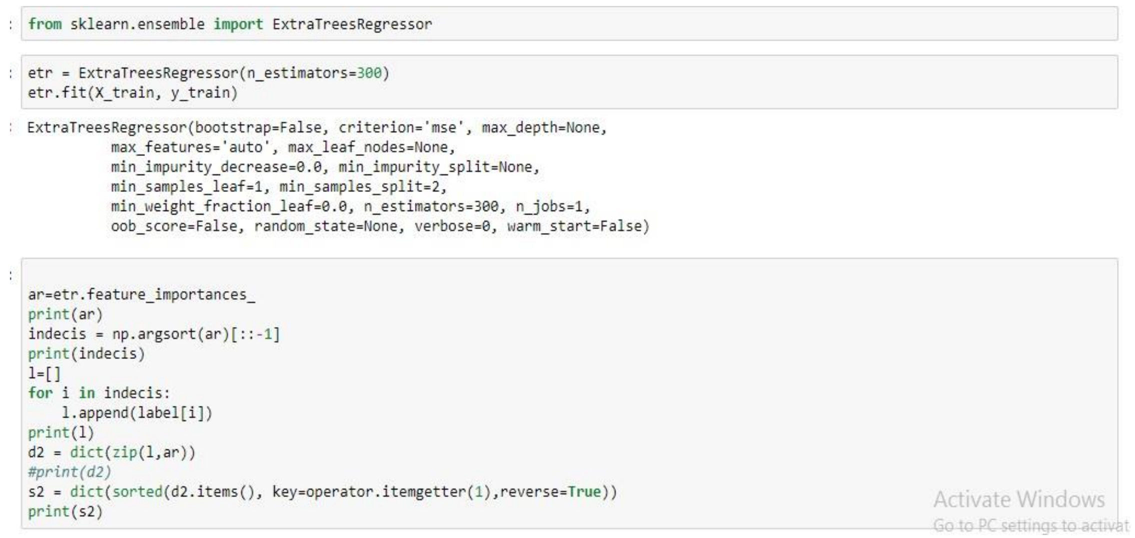
```
print("Coefficient of determination R^2 <-- on test set: {}".format(dtr.score(X_test, y_test)))
```

Figure 5.9 Decision Tree

5.8 FEATURE SELECTION

Feature Selection is one of the core concepts in machine learning which hugely impacts the performance of your model. The data features that you use to train your machine learning models have a huge influence on the performance you can achieve.

Feature selection and Data cleaning should be the first and most important step of your model designing.Feature Selection is the process where you automatically or manually select those features which contribute most to your prediction variable or output in which you are interested in.



```
from sklearn.ensemble import ExtraTreesRegressor
```

```
etr = ExtraTreesRegressor(n_estimators=300)
etr.fit(X_train, y_train)
```

```
ExtraTreesRegressor(bootstrap=False, criterion='mse', max_depth=None,
          max_features='auto', max_leaf_nodes=None,
          min_impurity_decrease=0.0, min_impurity_split=None,
          min_samples_leaf=1, min_samples_split=2,
          min_weight_fraction_leaf=0.0, n_estimators=300, n_jobs=1,
          oob_score=False, random_state=None, verbose=0, warm_start=False)
```

```
ar=etr.feature_importances_
print(ar)
indecis = np.argsort(ar)[::-1]
print(indecis)
l=[]
for i in indecis:
    l.append(label[i])
print(l)
d2 = dict(zip(l,ar))
#print(d2)
s2 = dict(sorted(d2.items(), key=operator.itemgetter(1),reverse=True))
print(s2)
```

Figure 5.10 Feature Selection

CHAPTER 6

SNAPSHOT OF MODULES

6.1 MONTH WISE PLOTTING THE INFLUENCING GASES

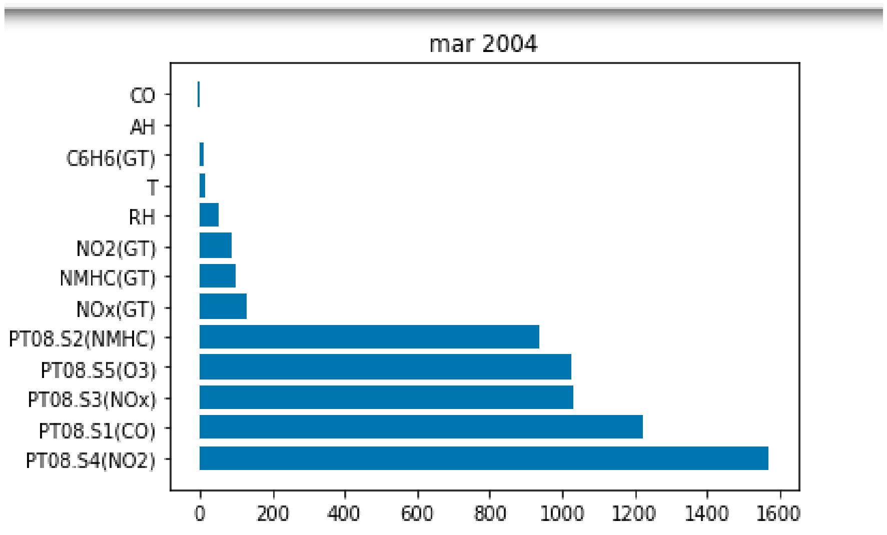


Figure 6.1 March 2004

This graph shows the distribution of gases of March 2004.

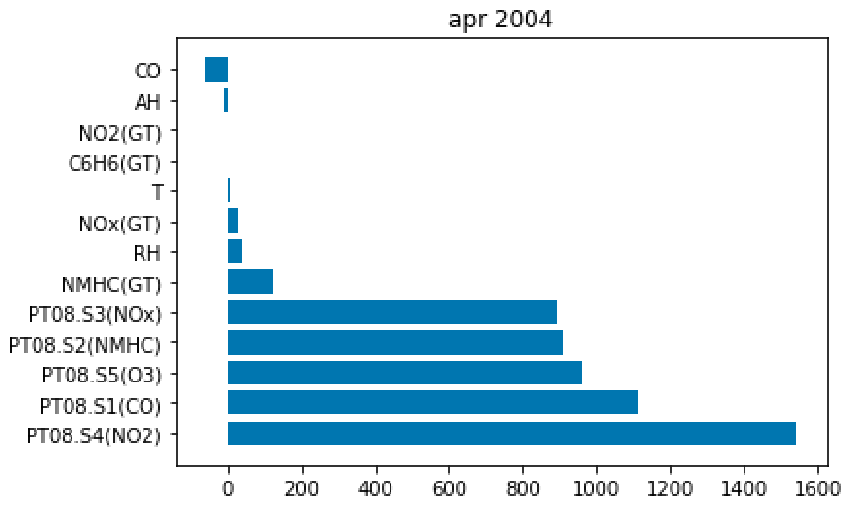


Figure 6.2 April 2004

This graph shows the distribution of gases of April 2004.

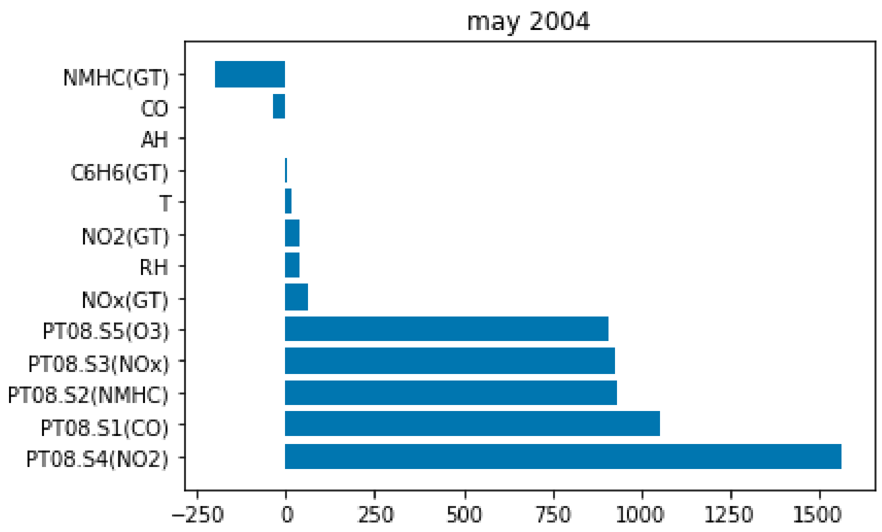


Figure 6.3 May 2004

This graph shows the distribution of gases of May 2004.

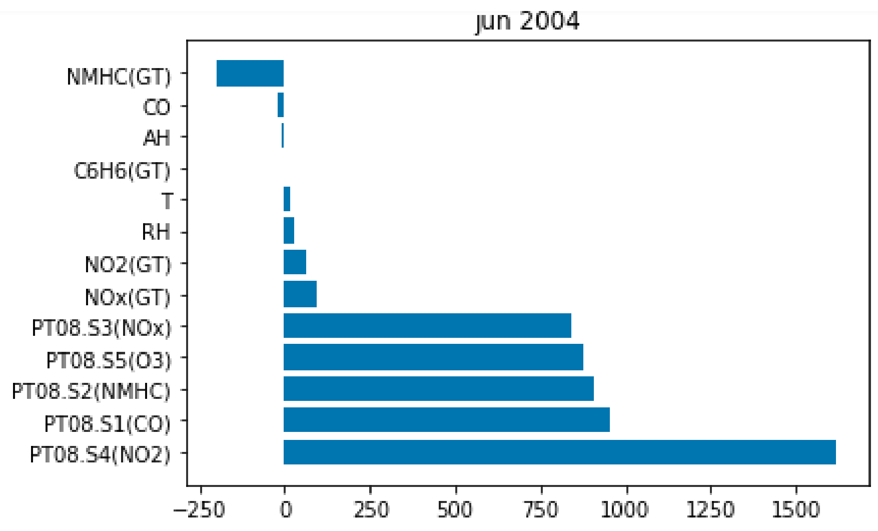


Figure 6.4 June 2004

This graph shows the distribution of gases of June 2004.

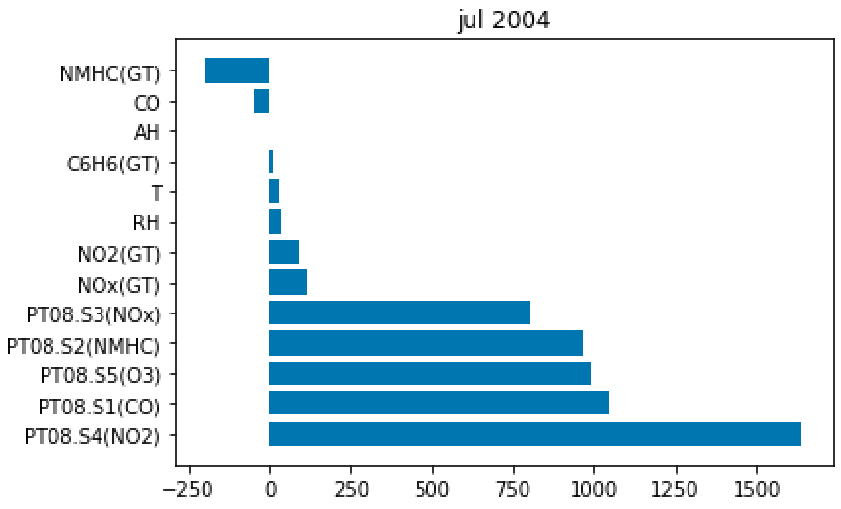


Figure 6.5 July 2004

This graph shows the distribution of gases of July 2004.

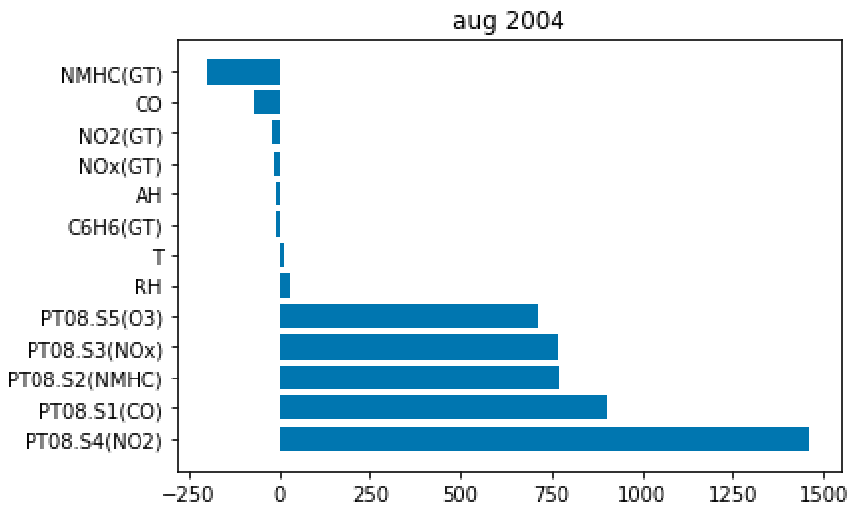


Figure 6.6 August 2004

This graph shows the distribution of gases of August 2004.

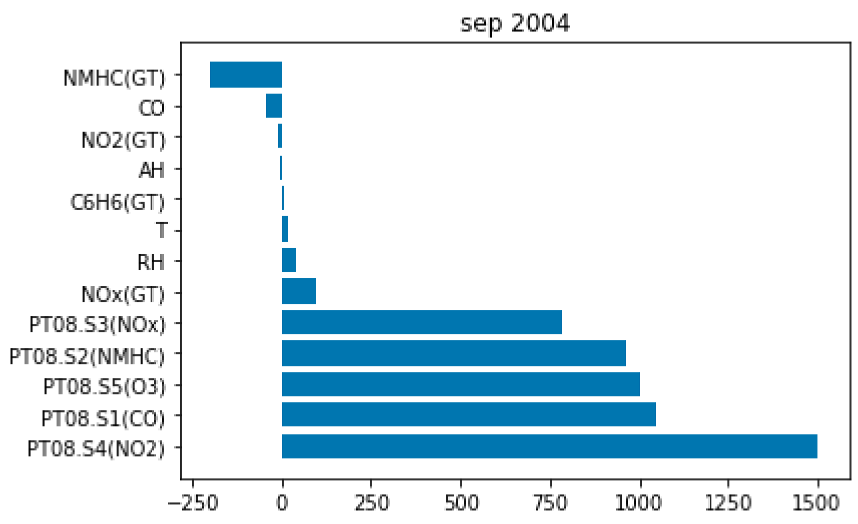


Figure 6.7 September 2004

Figure 6.7 shows the distribution of gases of September 2004.

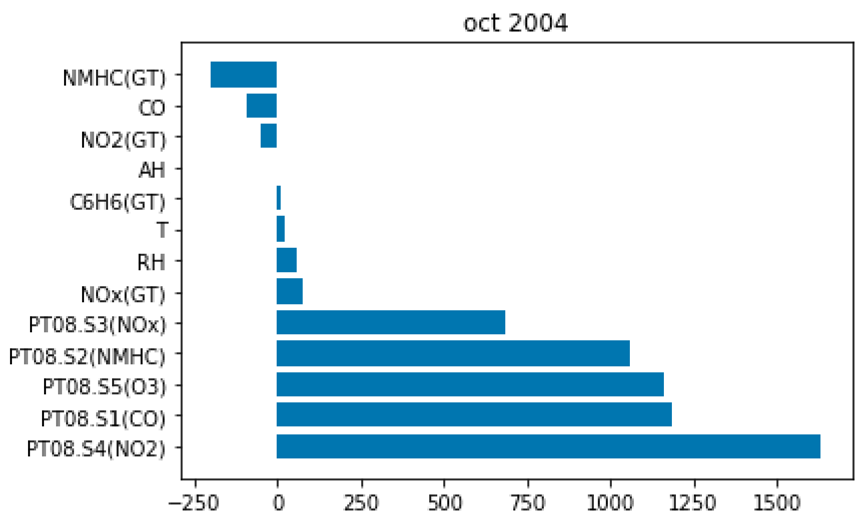


Figure 6.8 October 2004

Figure 6.8 shows the distribution of gases of October 2004.

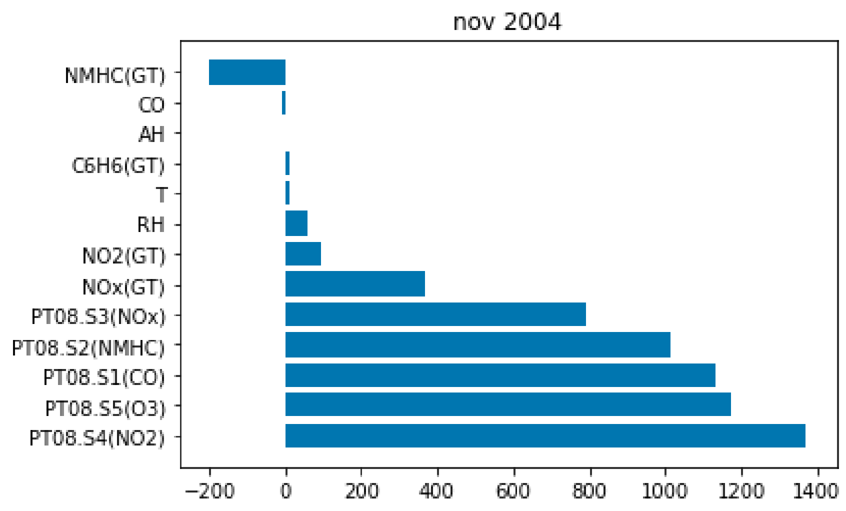


Figure 6.9 November 2004

Figure 6.9 shows the distribution of gases of November 2004.

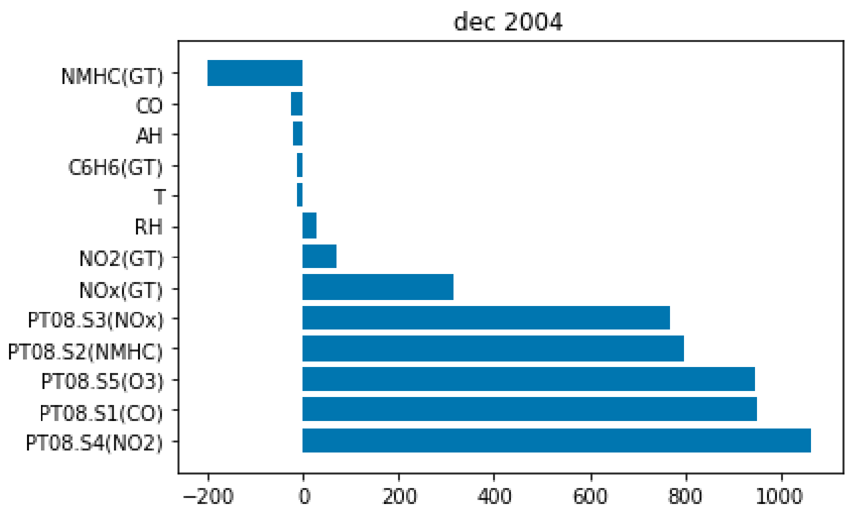


Figure 6.10 December 2004

Figure 6.10 shows the distribution of gases of December 2004.

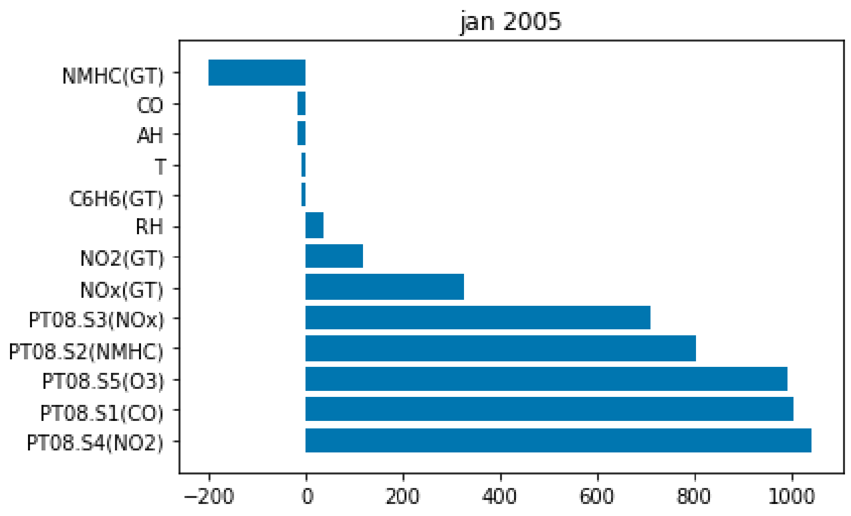


Figure 6.11 January 2005

Figure 6.11 shows the distribution of gases of January 2005.

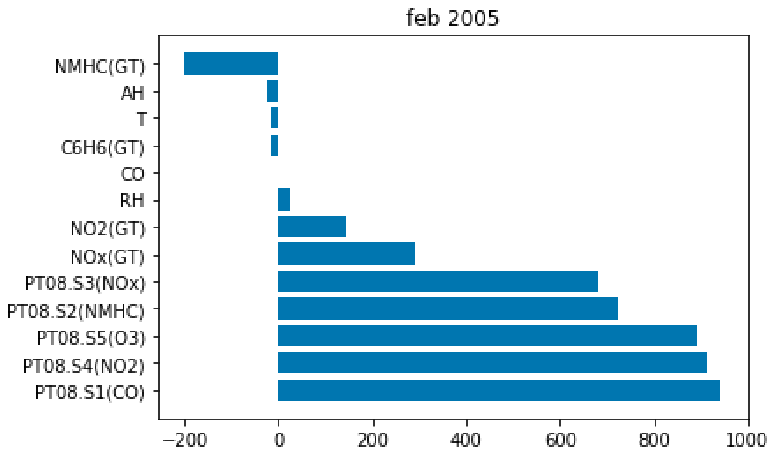


Figure 6.12 February 2005

Figure 6.12 shows the distribution of gases of February 2005.

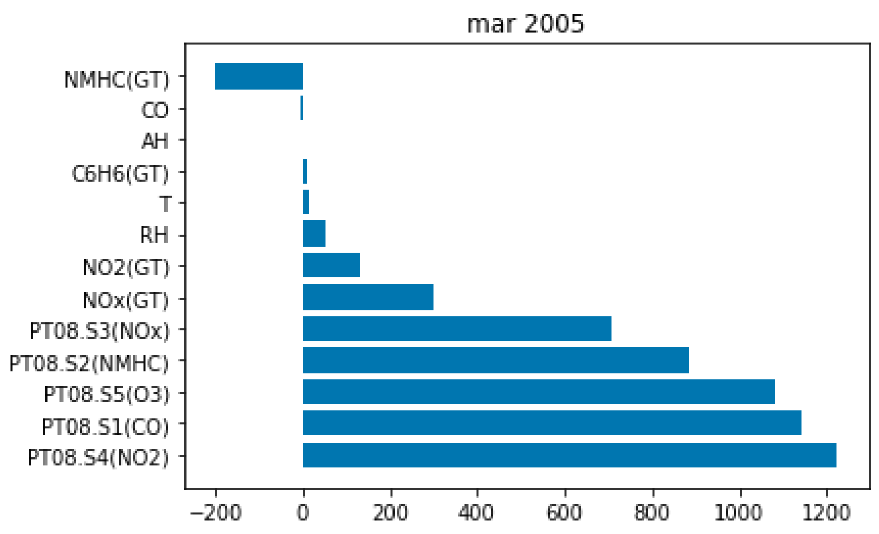


Figure 6.13 March 2005

Figure 6.13 shows the distribution of gases of March 2005.

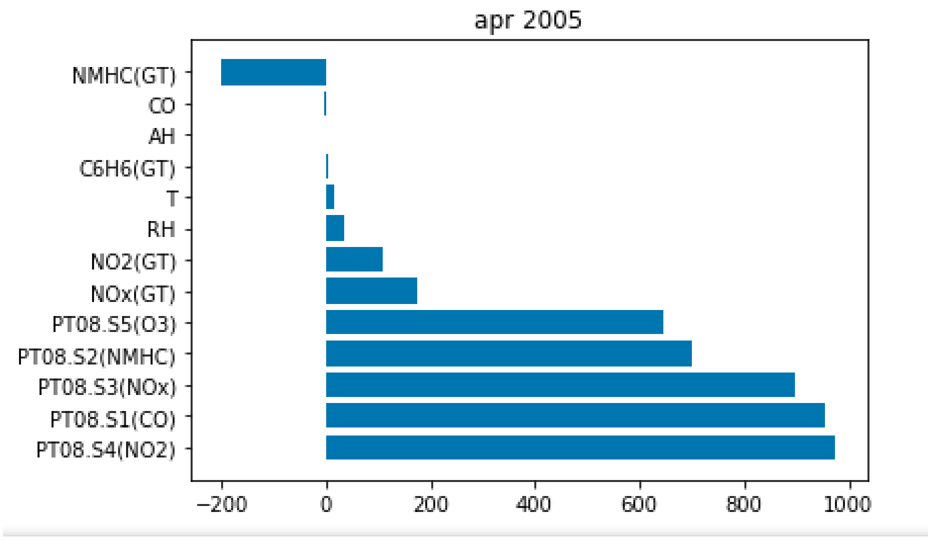


Figure 6.14 April 2005

Figure 6.14 shows the distribution of gases of April 2005.

6.2 FEATURE IMPORTANCE

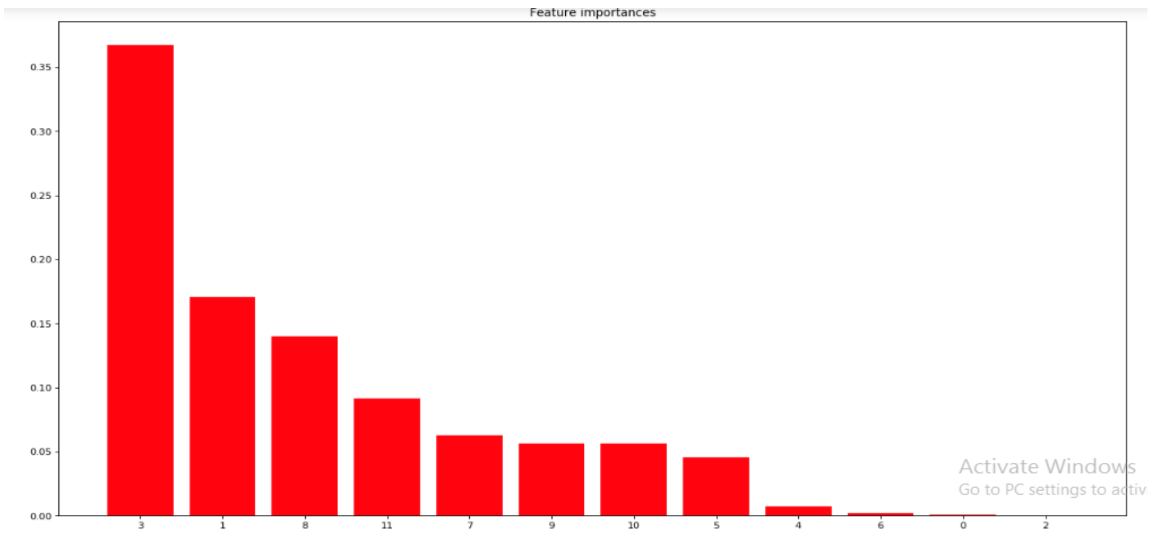


Figure 6.15 Feature Importance

Figure 6.15 shows the important features from the dataset.

6.3 COLUMN AVERAGING

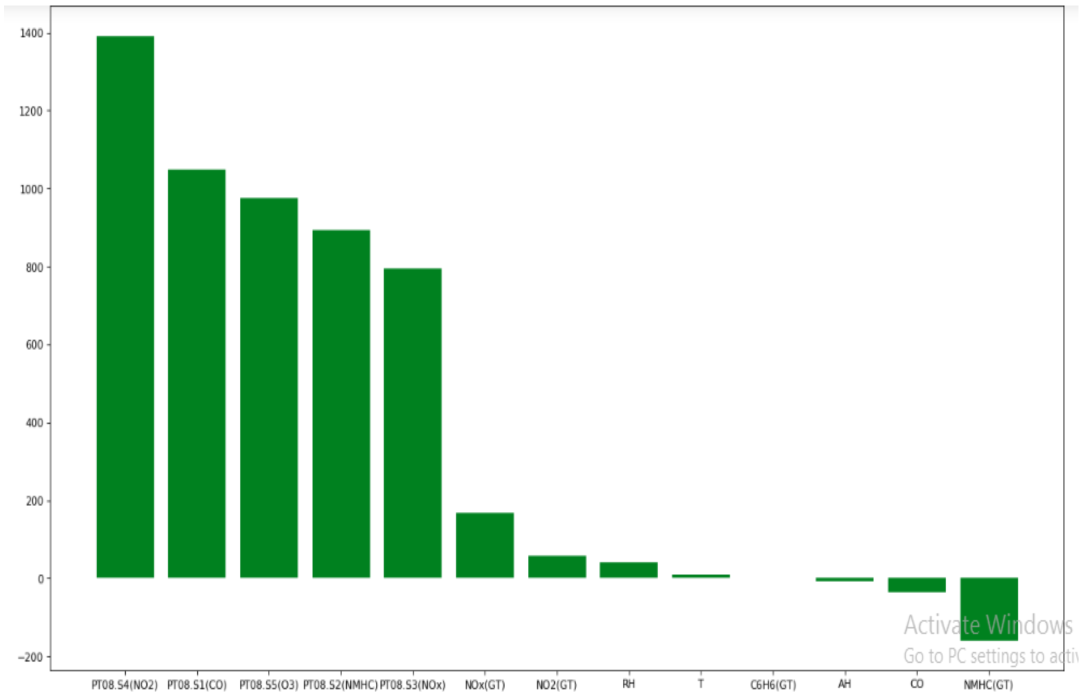


Figure 6.16 Column Averaging

Figure 6.16 shows the column-wise averaging of the gases in the dataset.

CHAPTER 7

CONCLUSION AND FUTURE WORK

The air quality data generation through air quality monitoring network available today, involves large number of monitoring agencies, personal and equipment for sampling, chemical analysis and data reporting etc. The involvement of several agencies increases the probability of variations and personal biases reflecting on the data. Therefore, the air quality data statistics available today is being recognized to be more indicative rather than absolute and perfect.

The future work

- Society looks for a pollution-free globe for happy living. The global warming threat is waiting at the door.
- Government rules, governing pollution control in private sector industries are not implemented that effectively.
- This scenario stresses the need for an efficient monitoring system with the collaboration of users, domain experts, hardware designers and software developers. This study is an attempt in this direction.

REFERENCES

1. Balram Pani, “Sources of Air Pollution,” in Air Pollution, Text book of Environmental Chemistry, I. K. International Publishing House Pvt. Ltd, New Delhi,(2007), PP. 197-198,.

2. Jane K. Hart, Kirk Martinez, “Environmental Sensor Networks: A revolution in the earth system science?,” Earth-Science reviews,(2006) , vol. 78, PP. 177-191 .

3. Tamilnadu Pollution Control Board, Available: www.tnpcb.gov.in.

4. Narain, Cover story, “Carbon Discredit” Science and Environment fortnightly, Down To Earth, December 1,(2008),PP. 41.

5. Centre for Science and Environment, New Delhi, “Green Rating network,” Science and Environment fortnightly, Down To Earth, December 16, (2008), PP. 51.

6. Rajesh Rangarajan, “Air Quality Monitoring Regime in India an Overview,” Pollution Monitoring Series: Briefing Note 1, Centre for Development Finance, April (2010), PP. 1-11.

7. https://www.wikipedia.org

8. https://towardsdatascience.com

9. https://medium.com