

Comparative study of Machine Learning algorithms for predicting the air pollution data

*Faculty of Engineering & Technology, Jadavpur University in the fulfilment of
the requirements for the degree of Master of Computer Application*

Submitted by

RAJASHREE SARKAR

Registration Number: 133665 of 2015-16

Class Roll Number: 001510503003

Examination Roll No: MCA186003

Under the Supervision of

Dr. Sarbani Roy

Associate Professor, Dept. of Computer Science & Engineering

Jadavpur University

Dept. of Computer Science & Engineering

Faculty of Engineering and Technology

Jadavpur University

May 2018

Declaration of Originality &

Compliance of Academics Ethics

I hereby declare that this project report contains literature survey and original research work done by me, as part of my MCA studies. All information in this document have been obtained and presented in accordance with academic rules and ethical conduct.

Name : Rajashree Sarkar

Registration Number : 133665 of 2015-2016

Class Roll Number : 001510503003

Examination Roll Number : MCA186003

Project Title : Comparative study of Machine Learning algorithms for predicting the air pollution data

Rajashree Sarkar

Department of Computer Science & Engineering
Faculty of Engineering & Technology

Jadavpur University

To Whom It May Concern,

This is to certify that RAJASHREE SARKAR, Registration Number: 133665 of 2015-16, Class Roll Number: 001510503003, Examination Roll Number: MCA186003, a student of MCA, from the Department of Computer Science & Engineering, under the Faculty of Engineering and Technology, Jadavpur University has done a report under my supervision, entitled as "Comparative study of Machine Learning algorithms for predicting the air pollution data" The project report is approved for submission towards the fulfilment of the requirements for the degree of Master of Computer Application, from the Department of Computer Science & Engineering, Jadavpur University for the session 2017-18.

Dr. Sarbani Roy

(Supervisor)

Associate Professor

Department of Computer Science and Engineering

Jadavpur University

Dr. Ujjwal Maulik

(Head of the Department)

Professor

Department of Computer Science and Engineering

Jadavpur University

Dr. Chiranjib Bhattacharjee

(Dean)

Professor

Faculty of Engineering and Technology

Jadavpur University

Certificate of Approval

(Only in case the report is approved)

The forgoing project report is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve this thesis only for the purpose for which it is submitted.

Signature of the Examiner

Signature of the Examiner

Date: _____

Date: _____

Acknowledgement

I would like to express my deepest gratitude to my advisor and guide Dr. Sarbani Roy, for her excellent guidance, caring, patience, and providing me with an excellent atmosphere for doing research. I strongly believe that I will get a lot of encouragement and inspiration from her throughout the project work. With her invaluable guidance, this work will be a successful one. I am equally grateful to Dr. Ujjwal Maulik, Head of The Department, Computer Science & Engineering, Jadavpur University, for his support towards our department. Last but not least, I would like to thank my parents and all respected teachers for their valuable suggestions and helpful discussions.

Regards,

Rajashree Sarkar

Department of Computer Science & Engineering

MCA

Jadavpur University

ABSTRACT

The field of optimization and machine learning are increasingly interplayed and optimization in different problems leads to the use of machine learning approaches. Machine learning algorithms work in reasonable computational time for specific classes of problems and have important role in extracting knowledge from large amount of data. In this report three different types of machine learning algorithm have been applied to a data which contains air quality information of India. So₂ and No₂ have been predicted using linear regression, Support Vector Machine (SVM) and regression tree. Several graphs has been plotted to identify the algorithm which works better than the other. Also it is observed in what percentage of training data set these algorithm gives the better result.

Contents

Chapter 1

INTRODUCTION	3
1.1 BACKGROUND.....	3
1.2 MOTIVATION	3
1.3 OBJECTIVE.....	4
1.4 ORGANIZATION OF THE PROJECT:.....	4

Chapter 2

LITERATURE SURVEY.....	5
------------------------	---

Chapter 3

WORKFLOW.....	7
3.1 PROBLEM DESCRIPTION	7
3.2 DATA COLLECTION	7
3.3 DATA PREPARATION	7
3.4 MACHINE LEARNING METHOD.....	10
3.4.1 TRAIN DATA AND TEST DATA	10
3.4.2 RMS ERROR	11
3.4.3 LINEAR REGRESSION	11
3.4.4 SUPPORT VECTOR MACHINE.....	17
3.4.5 REGRESSION TREE	24

Chapter 4

EXPERIMENTAL DATA	29
4.1. CALCULATION ON AIR POLLUTION DATA 37 STATES OF INDIA:.....	29
4.2 CALCULATION FOR THE STATE MAHARASHTRA	33

Chapter 5

CONCLUSION AND FUTURE WORK	37
5.1 CONCLUSION	37
5.2 FUTURE WORK	37
REFERENCES.....	38

1.1 BACKGROUND

The effects of rapid growth of the world's population are reflected in the overuse and scarcity of natural resources, deforestation, climate change, and especially environmental pollution. Adverse health impacts from exposure to outdoor air pollutants are complicated functions of pollutant compositions and concentrations. Major outdoor air pollutants in cities include ozone (O₃), particle matter (PM), sulfur dioxide (SO₂), carbon monoxide (CO), nitrogen oxides (NO_x), volatile organic compounds (VOCs), pesticides, and metals, among others. A recent study using a global atmospheric chemistry model estimated that 3.3 million annual premature deaths worldwide are linked to outdoor air pollution, which is expected to double by 2050, mostly due to anthropogenic fine particulate matter (aerodynamic diameter < 2.5 μm; PM_{2.5}) [2]. Since industrialization, there has been an increasing concern about environmental pollution. As mentioned in the WHO report 7 million premature deaths annually linked to air pollution, air pollution is the world's largest single environmental risk. Moreover as reported in the NY Times article, India's Air Pollution Rivals China's as World's Deadliest it has been found that India's air pollution is deadlier than even China's. Using this dataset, one can explore India's air pollution levels at a more granular scale.

1.2 MOTIVATION

New computing technologies, machine learning today is not like machine learning of the past. It was born from pattern recognition and the theory that computers can learn without being programmed to perform specific tasks; researchers interested in artificial intelligence wanted to see if computers could learn from data. The iterative aspect of machine learning is important because as models are exposed to new data, they are able to independently adapt. They learn from previous computations to produce reliable, repeatable decisions and results.

It's a science that's not new – but one that has gained fresh momentum[6]. So in this paper So₂ and No₂ have been predicted using linear regression, Support Vector Machine (SVM) and regression tree. Several graphs had been plotted to identify the algorithm which works better than the other. Also it is observed in what percentage of training data set these algorithm gives the better result.

1.3 OBJECTIVE

Main objective of this project is to find the machine learning algorithm which works better and using that algorithm to predict the different component of air . This report also contains the percentage of data to be trained to get better result.

1.4 ORGANIZATION OF THE PROJECT:

Chapter 2 of the project contains the related work of this project. Then in Chapter 3 the report says the methodology applied to complete the project. This portion includes the problem statement, then the source from where the dataset is collected. Next it mentions how the data is prepared for using the different machine learning algorithm .Next few pages describe the three different machine learning algorithm . After that experimental data are attached in Chapter 4 which includes several graphs and at the end in Chapter 5 conclusion and the future works are present.

LITERATURE SURVEY

A large number of work has been done previously to apply machine learning algorithms onto air quality predictions. Some researchers aimed to predict targets into discretized levels. Kalapanidas et al. elaborated effects on air pollution only from meteorological features such as temperature, wind, precipitation, solar radiation, and humidity, and classified air pollution into different levels (low, med,high, alarm) by using a lazy learning approach, Case Based Reasoning (CBR) system[1]. Athanasiadis et al. employed *s*-fuzzy lattice neurocomputing classifier to predict and categorize O₃ concentration into 3 levels (low, mid, and high) based on meteorological features and other pollutants like SO₂,NO, NO₂ and so on. Kunwar et al. utilized principle component analysis (PCA) and ensemble learning models to predict categorized air quality index (AQI) and combined air quality index (CAQI). However, the process of converting regression tasks to classification tasks is problematic, as it ignores the magnitude of the numeric data and consequently is inaccurate.

Other researchers worked on predicting concentrations of pollutants. Corani worked on training neural network models to predict hourly O₃ and PM₁₀ concentration based on data from the previous day. Performances of Feed Forward Neural Network (FFNN) and Pruned Neural Network (PNN) were mainly compared. More efforts have been made on FFNN, Fu et al. applied a rolling mechanism and gray model to improve traditional FFNN models. Jiang et al. explored multiple models (physical & chemical model, regression model, multiple layer perceptron) on the air pollutant prediction task and their result shows statistical models are competitive to the classical physical & chemical models. Ni, X. Y. et al. compared multiple statistical models based on PM_{2.5} data around Beijing, which implies linear regression models sometimes can be better than the other models. Multi-task Learning (MTL) focuses on learning multiple tasks that have commonalities together, which can improve the efficiency and accuracy of the models. It has achieved tremendous successes in many fields such as: natural language processing, image recognition, bioinformatics, marketing prediction, etc. A variety of regularizations can be utilized to enhance the commonalities of the tasks including $\ell_{2,1}$ -norm, nuclear-norm, spectral norm Frobenius norm etc. However,

most of former machine learning works on air pollutant prediction don't consider the similarities between the models and only focus on improving model performance for a single task[1]. There are also many related work using machine learning algorithm such as : A malware detection system based on the data mining and machine learning technique has been proposed in . Malware represents a serious threat to the security of computer systems. Traditional malware detection techniques like signature-based, heuristic-based, Specification-based detection are used to detect the known malware. These techniques detect the known malware accurately, but unable to detect the new, unknown malware. The proposed method in consists of disassemble process, feature extraction process and feature selection process. Three classification algorithms were employed on dataset to generate and train the classifiers named as Ripper, C4.5, IBk. The goal of study was to find an effective machine learning method for classifying ElectroMyoGram (EMG) signals by applying de-noising, feature extraction and classifier. The study presented a framework for classification of EMG signals using multi-scale principal component analysis for de-noising, discrete wavelet transform for feature extraction and decision tree algorithms for classification. The presented framework automatically classified the EMG signals as myopathic, ALS or normal, using CART, C4.5 and random forest decision tree algorithms. Decision tree algorithms are extensively used in machine learning field to classify biomedical signals. De-noising and feature extraction methods were also utilized to get higher classification accuracy. Since the application of microarray data for cancer classification is important, researchers have tried to analyze gene expression data using various computational intelligence methods. A novel method for gene selection has been proposed in . This method utilizing particle swarm optimization, is combined with a decision tree as the classifier to select a small number of informative genes from the thousands of genes in the data that can contribute in identifying cancers. A class of relevant speech signal processing algorithms as probabilistic inference problems has been described in . Starting with an observation model that relates all involved random variables, the authors converted the respective joint probability density function into its Bayesian network representation in order to infer the desired signal estimates[4]. Therefore, we decide to use meteorological and pollutant data to do prediction for So₂ and No₂ based on linear models. In this work, we focus on three different prediction model such as linear regression, Support Vector Machine(SVM) and regression tree using R language for making a comparison between these methods and tried to find out the percentage of data to be trained to get the better result.

3.1 PROBLEM DESCRIPTION

In this project we have compared the different machine learning algorithm such as Linear Regression, SVM and Regression tree while predicting the value of SO₂ and NO₂ in the states of India based on historical data. Next we have found The root mean square distance (rms distance) with the predicted value and the given value. Several Graphs of Percentage of trained data Vs rms error using the three above mentioned machine learning algorithm are prepared using R language for all the states data and state-specific data.

3.2 DATA COLLECTION

We collected air pollutant data from *kaggle* website[7].

This data is combined (across the years and states) and largely clean version of the Historical Daily Ambient Air Quality Data released by the Ministry of Environment and Forests and Central Pollution Control Board of India under the National Data Sharing and Accessibility Policy (NDSAP).

3.3 DATA PREPARATION

We had prepared the dataset in the following way.

STEP1:

The data set contains 435742 observation of 13 variables. The column contains the following information .

stn_code - Mentions the station code of different area.

sampling_date- Mentions the month and a corresponding code when the data is collected

state-	Mentions the name of the state
location-	Mentions the particular location of an individual state.
agency-	Mentions the name of the agency who has collected the data.
type-	Mentions the type of area, whether industrial or residential etc.
so2-	Mentions the amount of sulphur di oxide.
no2-	Mentions the amount of nitrogen di oxide.
rspm-	Mentions the amount of respirable suspended particulate matter.
spm-	Mentions the amount of suspended particulate matter.
location_monitoring_station-	Mentions address of the monitoring station.
pm2_5-	Mentions the amount of pm2_5.
date -	Mentions the date of collecting data .

NA REMOVAL:

In the data set many fields contain Not Applicable[N.A.] value or *Missing Values*. So we first replaced all the N.A by 0 . For this purpose we wrote yhe below-mentioned command in R language.

```
dataset$column_name[which(is.na(dataset$column_name))]<-0
```

STEP2:

In the data set there was more than one value present against a single date. So we found the mean of the records present against the single date. In this way a csv file was created which contain the values of *so2, no2, spm, rspm, pm2_5* against each date.

Method used :

```
tapply(dataframe$column_name, dataframe$date, mean)
```

tapply() generally apply a function to each cell of a ragged array, that is to each (non-empty) group of values given by a unique combination of the levels of certain factors.

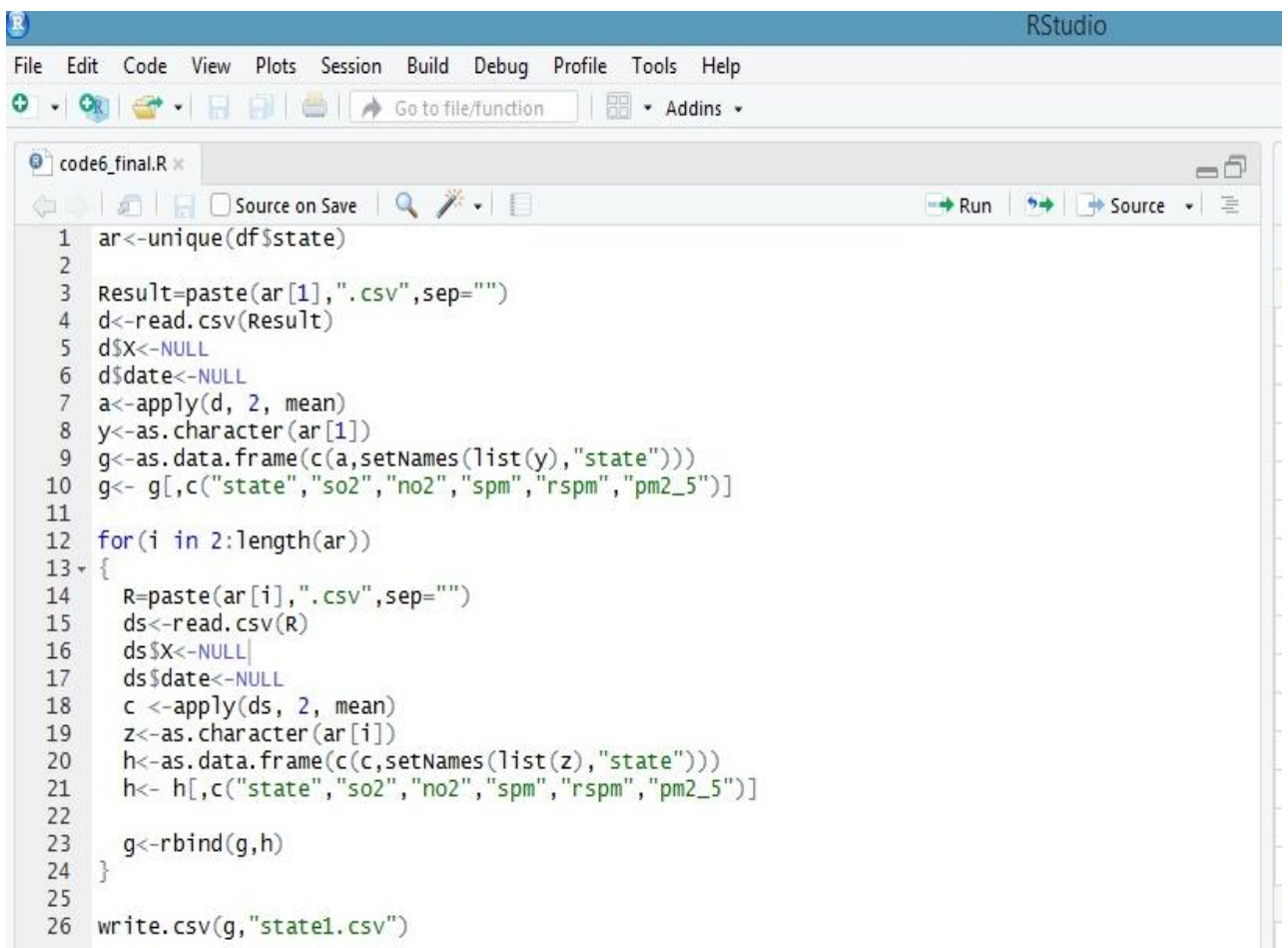
STEP3:

This process is repeated for the 37 states individually, and 37 csv files were created and each file was named after the states which were present in the main data sheet each file contained 6 variable- *date, SO2, NO2, spm, rspm, pm2_5*

STEP4:

Next the mean value for SO₂, NO₂, SPM, RSPM and PM_{2.5} is found against each state. Thus a new csv file named "State.csv" is prepared whose contents are: - *state,SO₂,NO₂,spm, rspm, pm_{2.5}* .

Method used in R:

The image shows a screenshot of the RStudio interface. The title bar at the top reads "RStudio". Below it is a menu bar with options: File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help. A toolbar contains icons for file operations and a search bar labeled "Go to file/function". The main editor window shows a script named "code6_final.R" with the following R code:

```
1 ar<-unique(df$state)
2
3 Result=paste(ar[1],".csv",sep="")
4 d<-read.csv(Result)
5 d$X<-NULL
6 d$date<-NULL
7 a<-apply(d, 2, mean)
8 y<-as.character(ar[1])
9 g<-as.data.frame(c(a,setNames(list(y),"state")))
10 g<- g[,c("state","so2","no2","spm","rspm","pm2_5")]
11
12 for(i in 2:length(ar))
13 {
14   R=paste(ar[i],".csv",sep="")
15   ds<-read.csv(R)
16   ds$X<-NULL
17   ds$date<-NULL
18   c <-apply(ds, 2, mean)
19   z<-as.character(ar[i])
20   h<-as.data.frame(c(c,setNames(list(z),"state")))
21   h<- h[,c("state","so2","no2","spm","rspm","pm2_5")]
22
23   g<-rbind(g,h)
24 }
25
26 write.csv(g,"state1.csv")
```

FIG-3.1

The above code[as shown in the FIG-3.1] is performed for 37 states and then they are merged together.

apply() function returns a vector or array or list of values obtained by applying a function to margins of an array or matrix.

STEP5:

The above mentioned steps(i.e step1-step-4) is done for the state " Maharashtra" as it contains the maximum amount of data among the all states. Thus we got another csv file(named MR.csv) whose content are

date SO2,NO2,spm,rspm,pm2_5

Thus after preparing the data set we used the three machine learning algorithm

3.4 MACHINE LEARNING METHOD

This section consist of the details description of the three machine learning method we have applied to the data sat

3.4.1 TRAINED DATA AND TEST DATA:

Basically there is three data sets: training, validation and testing.

One can train the classifier using 'training set', tune the parameters using 'validation set' and then test the performance of our classifier on unseen 'test set'. An important point to note is that during training the classifier only the training and/or validation set is available. The test set must not be used during training the classifier. The test set will only be available during testing the classifier.

There is no 'one' way of choosing the size of training/testing set and people apply heuristics such as 10% testing and 90% training. However, doing so can bias the classification or regression results and the results may not be generalizable. A well accepted method is N-Fold cross validation, in which one can randomize the dataset and create N (almost) equal size partitions. Then choose Nth partition for testing and N-1 partitions for training the classifier. Within the training set we can further employ another K-fold cross validation to create a validation set and find the best parameters. And repeat this process N times to get an average of the metric. Since we want to get rid of classifier 'bias' we repeat this above process M times (by randomizing data and splitting into N fold) and take average of the metric. Cross-validation is almost unbiased, but it can also be misused if training and

validation set comes from different populations and knowledge from training set is used in the test set.

3.4.2 RMS ERROR:

The regression line predicts the average y value associated with a given x value. Note that is also necessary to get a measure of the spread of the y values around that average. To do this, we use the root-mean-square error (r.m.s. error).

To construct the r.m.s. error, we first need to determine the residuals. Residuals are the difference between the actual values and the predicted values. I denoted them by $\hat{y}_i - y_i$, where y_i is the observed value for the ith observation and \hat{y}_i is the predicted value. They can be positive or negative as the predicted value under or over estimates the actual value. Squaring the residuals, averaging the squares, and taking the square root gives us the r.m.s error. We then use the r.m.s. error as a measure of the spread of the y values about the predicted y value[8].

$$RMSE_{errors} = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}}$$

3.4.3 LINEAR REGRESSION:

Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable. For example, a modeler might want to relate the weights of individuals to their heights using a linear regression model.

Before attempting to fit a linear model to observed data, a modeler should first determine whether or not there is a relationship between the variables of interest. This does not necessarily imply that one variable *causes* the other (for example, higher SAT scores do not *cause* higher college grades), but that there is some significant association between the two variables. A scatterplot can be a helpful tool in determining the strength of the relationship between two variables. If there appears to be no association between the proposed explanatory and dependent variables (i.e., the scatterplot does not indicate any

increasing or decreasing trends), then fitting a linear regression model to the data probably will not provide a useful model. A valuable numerical measure of association between two variables is the correlation coefficient, which is a value between -1 and 1 indicating the strength of the association of the observed data for the two variables.

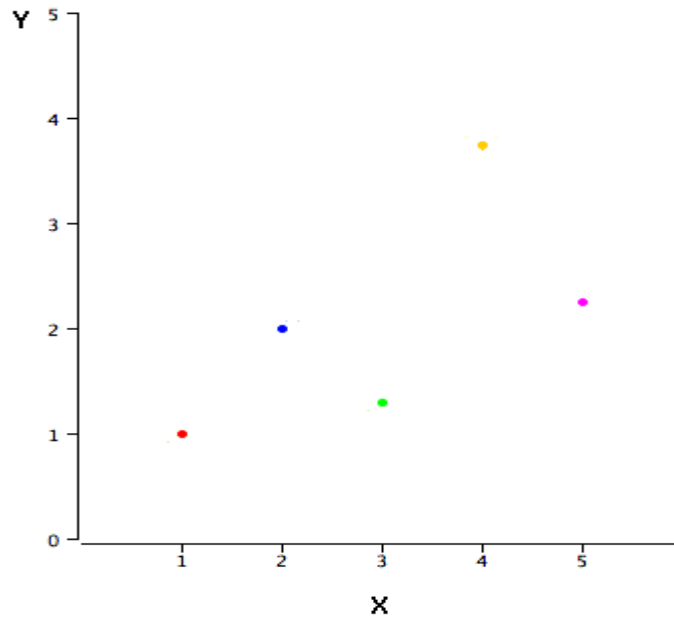
A linear regression line has an equation of the form $Y = a + bX$, where X is the explanatory variable and Y is the dependent variable. The slope of the line is b , and a is the intercept (the value of y when $x = 0$) [9].

In simple linear regression, we predict scores on one variable from the scores on a second variable. The variable we are predicting is called the *criterion variable* and is referred to as Y . The variable we are basing our predictions on is called the *predictor variable* and is referred to as X . When there is only one predictor variable, the prediction method is called *simple regression*. In simple linear regression, the topic of this section, the predictions of Y when plotted as a function of X form a straight line.

The example data in Table 1 are plotted in Figure 1. We can see that there is a positive relationship between X and Y . If we were going to predict Y from X , the higher the value of X , the higher our prediction of Y .

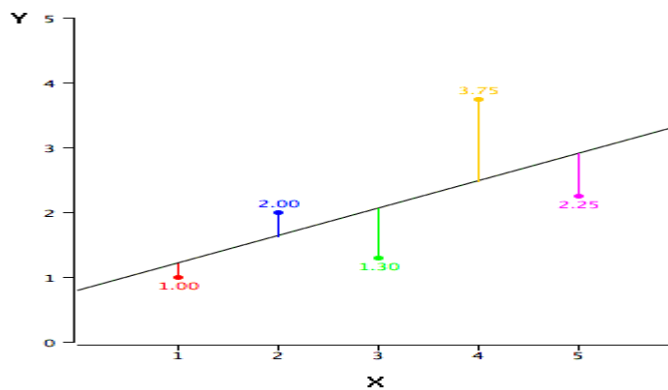
Table 1. Example data.

X	Y
1.00	1.00
2.00	2.00
3.00	1.30
4.00	3.75
5.00	2.25



A scatter plot of the example data[FIG - 3.2]

Linear regression consists of finding the best-fitting straight line through the points. The best-fitting line is called a *regression line*. The black diagonal line in Figure 2 is the regression line and consists of the predicted score on Y for each possible value of X. The vertical lines from the points to the regression line represent the errors of prediction. As we can see, the red point is very near the regression line; its error of prediction is small. By contrast, the yellow point is much higher than the regression line and therefore its error of prediction is large.



A scatter plot of the example data. [FIG - 3.3]

The black line consists of the predictions, the points are the actual data, and the vertical lines between the points and the black line represent errors of prediction.

[A] ERROR PREDICTION:

The error of prediction for a point is the value of the point minus the predicted value (the value on the line). Table 2 shows the predicted values (Y') and the errors of prediction ($Y-Y'$). For example, the first point has a Y of 1.00 and a predicted Y (called Y') of 1.21. Therefore, its error of prediction is -0.21.

Table 2. Example data.

X	Y	Y'	$Y-Y'$	$(Y-Y')^2$
1.00	1.00	1.210	-0.210	0.044
2.00	2.00	1.635	0.365	0.133
3.00	1.30	2.060	-0.760	0.578
4.00	3.75	2.485	1.265	1.600
5.00	2.25	2.910	-0.660	0.436

[B] BEST FITTING LINE:

By far, the most commonly-used criterion for the best-fitting line is the line that minimizes the sum of the squared errors of prediction. That is the criterion that was used to find the line in Figure 2. The last column in Table 2 shows the squared errors of prediction. The sum of the squared errors of prediction shown in Table 2 is lower than it would be for any other regression line.

The formula for a regression line is

$$Y' = bX + A$$

where Y' is the predicted score, b is the slope of the line, and A is the Y intercept. The equation for the line in Figure 2 is

$$Y' = 0.425X + 0.785$$

For $X = 1$,

$$Y' = (0.425)(1) + 0.785 = 1.21.$$

For $X = 2$,

$$Y' = (0.425)(2) + 0.785 = 1.64.$$

[C] COMPUTING THE REGRESSION LINE

In the age of computers, the regression line is typically computed with statistical software. However, the calculations are relatively easy, and are given here for anyone who is interested. The calculations are based on the statistics shown in Table 3. M_X is the mean of X, M_Y is the mean of Y, s_X is the standard deviation of X, s_Y is the *standard deviation* of Y, and r is the *correlation* between X and Y.

Table 3. Statistics for computing the regression line

M_X	M_Y	s_X	s_Y	r
3	2.06	1.581	1.072	0.627

The slope (b) can be calculated as follows:

$$b = r s_Y / s_X$$

and the intercept (A) can be calculated as

$$A = M_Y - bM_X.$$

For these data,

$$b = (0.627)(1.072) / 1.581 = 0.425$$

$$A = 2.06 - (0.425)(3) = 0.785$$

The calculations have all been shown in terms of sample statistics rather than population parameters. The formulas are the same; simply use the parameter values for means, standard deviations, and the correlation[10].

[D] MULTIPLE REGRESSION[11]:

Multiple linear regression is the most common form of linear regression analysis. As a predictive analysis, the multiple linear regression is used to explain the relationship between one continuous dependent variable and two or more independent variables. The independent variables can be continuous or categorical .

In this project we have used Multiple Regression, because while predicting SO₂ and NO₂, we used the other 4 variables such as spm , rspm, pm_{2_5},SO₂/NO₂(for NO₂/SO₂) using R.

Implementation:

Using R we have trained the 60% of data while predicting so₂ and plotted a graph State vs rms error , and again from the "State.csv" file and state specific data "MR.csv" we have plotted a graph Percentage of trained data Vs rms value.

in R,

```
m1<-lm(so2 ~ no2 + spm + rspm + pm2_5,data=st_train)
```

where st_train is the trained data set

Here m1 is the trained model, this model is further used to predict the data .

lm is used to fit linear models. It can be used to carry out regression, single stratum analysis of variance and analysis of covariance .

and for predicting ,we used the function,

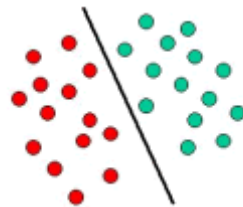
```
p1<- (predict(m1,data.frame("no2"=st_test[,2], "spm"=st_test[,3],"rspm"=st_test[,4],  
"pm2_5"=st_test[,5])))
```

"predict" is a generic function for predictions from the results of various model fitting functions. The function invokes particular *methods* which depend on the class of the first argument.

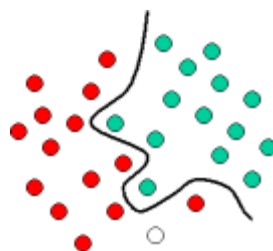
3.4.4 SUPPORT VECTOR MACHINE

Support Vector Machines (SVM) : Introductory Overview

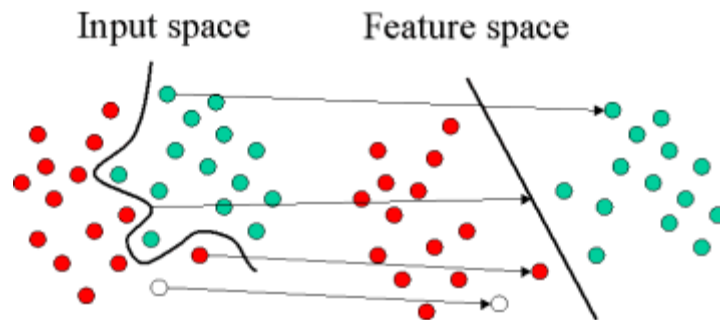
Support Vector Machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships. A schematic example is shown in the illustration below. In this example, the objects belong either to class GREEN or RED. The separating line defines a boundary on the right side of which all objects are GREEN and to the left of which all objects are RED. Any new object (white circle) falling to the right is labeled, i.e., classified, as GREEN (or classified as RED should it fall to the left of the separating line).



The above is a classic example of a linear classifier, i.e., a classifier that separates a set of objects into their respective groups (GREEN and RED in this case) with a line. Most classification tasks, however, are not that simple, and often more complex structures are needed in order to make an optimal separation, i.e., correctly classify new objects (test cases) on the basis of the examples that are available (train cases). This situation is depicted in the illustration below. Compared to the previous schematic, it is clear that a full separation of the GREEN and RED objects would require a curve (which is more complex than a line). Classification tasks based on drawing separating lines to distinguish between objects of different class memberships are known as hyperplane classifiers. Support Vector Machines are particularly suited to handle such tasks.



The illustration below shows the basic idea behind Support Vector Machines. Here we see the original objects (left side of the schematic) mapped, i.e., rearranged, using a set of mathematical functions, known as kernels. The process of rearranging the objects is known as mapping (transformation). Note that in this new setting, the mapped objects (right side of the schematic) is linearly separable and, thus, instead of constructing the complex curve (left schematic), all we have to do is to find an optimal line that can separate the GREEN and the RED objects.



Support Vector Machine (SVM) is primarily a classifier method that performs classification tasks by constructing hyperplanes in a multidimensional space that separates cases of different class labels. SVM supports both regression and classification tasks and can handle multiple continuous and categorical variables. For categorical variables a dummy variable is created with case values as either 0 or 1. Thus, a categorical dependent variable consisting of three levels, say (A, B, C), is represented by a set of three dummy variables:

A: {1 0 0}, B: {0 1 0}, C: {0 0 1}

To construct an optimal hyperplane, SVM employs an iterative training algorithm, which is used to minimize an error function. According to the form of the error function, SVM models can be classified into four distinct groups:

- Classification SVM Type 1 (also known as C-SVM classification)
- Classification SVM Type 2 (also known as nu-SVM classification)
- Regression SVM Type 1 (also known as epsilon-SVM regression)
- Regression SVM Type 2 (also known as nu-SVM regression)

Following is a brief summary of each model.

[A]Classification SVM

CLASSIFICATION SVM TYPE 1

For this type of SVM, training involves the minimization of the error function:

$$\frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i$$

subject to the constraints:

$$y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0, i = 1, \dots, N$$

where C is the capacity constant, w is the vector of coefficients, b is a constant, and ξ_i represents parameters for handling nonseparable data (inputs). The index i labels the N training cases. Note that $y \in \pm 1$ represents the class labels and x_i represents the independent variables. The kernel ϕ is used to transform data from the input (independent) to the feature space. It should be noted that the larger the C , the more the error is penalized. Thus, C should be chosen with care to avoid over fitting.

CLASSIFICATION SVM TYPE 2

In contrast to Classification SVM Type 1, the Classification SVM Type 2 model minimizes the error function:

$$\frac{1}{2} w^T w - \nu \rho + \frac{1}{N} \sum_{i=1}^N \xi_i$$

subject to the constraints:

$$y_i (w^T \phi(x_i) + b) \geq \rho - \xi_i, \xi_i \geq 0, i = 1, \dots, N \text{ and } \rho \geq 0$$

In a regression SVM, we have to estimate the functional dependence of the dependent variable y on a set of independent variables x . It assumes, like other regression problems, that the relationship between the independent and dependent variables is given by a deterministic function f plus the addition of some additive noise.

[B] Regression SVM

$$y = f(x) + \text{noise}$$

The task is then to find a functional form for f that can correctly predict new cases that the SVM has not been presented with before. This can be achieved by training the SVM model on a sample set, i.e., training set, a process that involves, like classification (see above), the sequential optimization of an error function. Depending on the definition of this error function, two types of SVM models can be recognized:

REGRESSION SVM TYPE 1

For this type of SVM the error function is:

$$\frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i + C \sum_{i=1}^N \xi_i^*$$

which we minimize subject to:

$$\begin{aligned} w^T \phi(x_i) + b - y_i &\leq \varepsilon + \xi_i^* \\ y_i - w^T \phi(x_i) - b &\leq \varepsilon + \xi_i \\ \xi_i, \xi_i^* &\geq 0, i = 1, \dots, N \end{aligned}$$

REGRESSION SVM TYPE 2

For this SVM model, the error function is given by:

$$\frac{1}{2} w^T w - C \left(\nu \varepsilon + \frac{1}{N} \sum_{i=1}^N (\xi_i + \xi_i^*) \right)$$

which we minimize subject to:

$$\begin{aligned} (w^T \phi(x_i) + b) - y_i &\leq \varepsilon + \xi_i \\ y_i - (w^T \phi(x_i) + b) &\leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* &\geq 0, i = 1, \dots, N, \varepsilon \geq 0 \end{aligned}$$

There are number of kernels that can be used in Support Vector Machines models. These include linear, polynomial, radial basis function (RBF) and sigmoid.

[C] Kernel Functions

$$K(\mathbf{X}_i, \mathbf{X}_j) = \begin{cases} \mathbf{X}_i \cdot \mathbf{X}_j & \text{Linear} \\ (\gamma \mathbf{X}_i \cdot \mathbf{X}_j + C)^d & \text{Polynomial} \\ \exp(-\gamma |\mathbf{X}_i - \mathbf{X}_j|^2) & \text{RBF} \\ \tanh(\gamma \mathbf{X}_i \cdot \mathbf{X}_j + C) & \text{Sigmoid} \end{cases}$$

where $K(\mathbf{X}_i, \mathbf{X}_j) = \phi(\mathbf{X}_i) \cdot \phi(\mathbf{X}_j)$

that is, the kernel function, represents a dot product of input data points mapped into the higher dimensional feature space by transformation ϕ

Gamma is an adjustable parameter of certain kernel functions.

The RBF is by far the most popular choice of kernel types used in Support Vector Machines. This is mainly because of their localized and finite responses across the entire range of the real x-axis.

The SVM algorithm is implemented in practice using a kernel[12].

The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra, which is out of the scope of this introduction to SVM.

A powerful insight is that the linear SVM can be rephrased using the inner product of any two given observations, rather than the observations themselves. The inner product between two vectors is the sum of the multiplication of each pair of input values.

For example, the inner product of the vectors [2, 3] and [5, 6] is $2*5 + 3*6$ or 28.

The equation for making a prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows:

$$f(x) = B0 + \text{sum}(ai * (x, xi))$$

This is an equation that involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients B_0 and a_i (for each input) must be estimated from the training data by the learning algorithm.

Linear Kernel SVM

The dot-product is called the kernel and can be re-written as:

$$K(x, x_i) = \text{sum}(x * x_i)$$

The kernel defines the similarity or a distance measure between new data and the support vectors. The dot product is the similarity measure used for linear SVM or a linear kernel because the distance is a linear combination of the inputs.

Other kernels can be used that transform the input space into higher dimensions such as a Polynomial Kernel and a Radial Kernel. This is called the Kernel Trick.

It is desirable to use more complex kernels as it allows lines to separate the classes that are curved or even more complex. This in turn can lead to more accurate classifiers.

Polynomial Kernel SVM

Instead of the dot-product, we can use a polynomial kernel, for example:

$$K(x, x_i) = 1 + \text{sum}(x * x_i)^d$$

Where the degree of the polynomial must be specified by hand to the learning algorithm. When $d=1$ this is the same as the linear kernel. The polynomial kernel allows for curved lines in the input space.

Radial Kernel SVM

Finally, we can also have a more complex radial kernel. For example:

$$K(x,xi) = \exp(-\text{gamma} * \text{sum}((x - xi)^2))$$

Where gamma is a parameter that must be specified to the learning algorithm. A good default value for gamma is 0.1, where gamma is often $0 < \text{gamma} < 1$. The radial kernel is very local and can create complex regions within the feature space, like closed polygons in two-dimensional space.

In this report we have used Radial Kernel SVM

Implementation:

Using R we have trained the 60% of data while predicting so2 and plotted a graph State vs rms error , and again from the "State.csv" file and state specific data "MR.csv" we have plotted a graph Percentage of trained data Vs rms value.

in R,

for using svm function at first we have to include library(e1071)

```
s1<-svm(so2 ~ no2 + spm + rspm + pm2_5,data=st_train, type ="eps-  
regression",kernel="radial")
```

where st_train is the trained data set

Here s1 is the trained model, this model is further used to predict the data .

svm is used to train a support vector machine. It can be used to carry out general regression and classification (of nu and epsilon-type), as well as density-estimation. A formula interface is provided.

and for predicting ,we used the function,

```
s2<- (predict(m1,data.frame("no2"=st_test[,2], "spm"=st_test[,3],"rspm"=st_test[,4],  
"pm2_5"=st_test[,5]) ))
```

predict is a generic function for predictions from the results of various model fitting functions. The function invokes particular *methods* which depend on the class of the first argument[13].

3.4.5 REGRESSION TREE:

All regression techniques contain a single output (response) variable and one or more input (predictor) variables. The output variable is numerical. The general regression tree building methodology allows input variables to be a mixture of continuous and categorical variables. A decision tree is generated when each decision node in the tree contains a test on some input variable's value. The terminal nodes of the tree contain the predicted output variable values.

A Regression tree may be considered as a variant of decision trees, designed to approximate real-valued functions, instead of being used for classification methods.

Methodology

A regression tree is built through a process known as binary recursive partitioning, which is an iterative process that splits the data into partitions or branches, and then continues splitting each partition into smaller groups as the method moves up each branch.

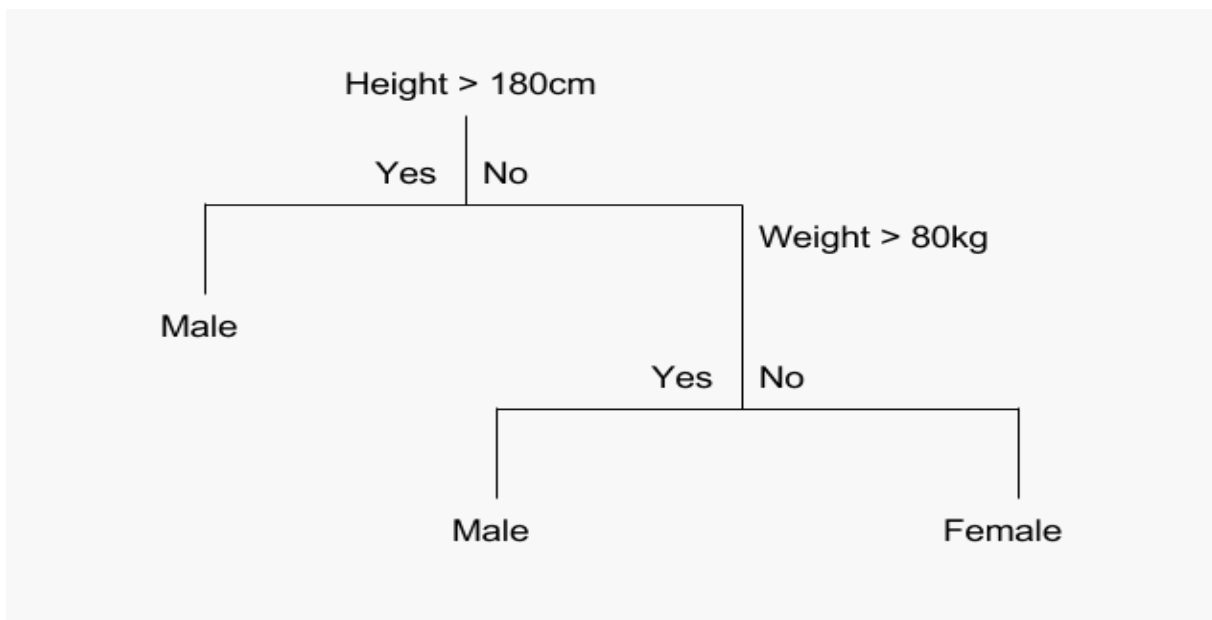
Initially, all records in the Training Set (pre-classified records that are used to determine the structure of the tree) are grouped into the same partition. The algorithm then begins allocating the data into the first two partitions or branches, using every possible binary split on every field. The algorithm selects the split that minimizes the sum of the squared deviations from the mean in the two separate partitions. This splitting rule is then applied to each of the new branches. This process continues until each node reaches a user-specified minimum node size and becomes a terminal node. (If the sum of squared deviations from the mean in a node is zero, then that node is considered a terminal node even if it has not reached the minimum size.)[14].

[A] CART Model Representation

The representation for the CART model is a binary tree. This is our binary tree from algorithms and data structures, nothing too fancy. Each root node represents a single input variable (x) and a split point on that variable (assuming the variable is numeric).

The leaf nodes of the tree contain an output variable (y) which is used to make a prediction.

Given a dataset with two inputs (x) of height in centimeters and weight in kilograms the output of sex as male or female, below is a crude example of a binary decision tree (completely fictitious for demonstration purposes only).



Example Decision Tree

The tree can be stored to file as a graph or a set of rules.

[B] Creating CART Model From Data

Creating a CART model involves selecting input variables and split points on those variables until a suitable tree is constructed.

The selection of which input variable to use and the specific split or cut-point is chosen using a greedy algorithm to minimize a cost function. Tree construction ends using a predefined

stopping criterion, such as a minimum number of training instances assigned to each leaf node of the tree.

[C] Greedy Splitting

Creating a binary decision tree is actually a process of dividing up the input space. A greedy approach is used to divide the space called recursive binary splitting.

This is a numerical procedure where all the values are lined up and different split points are tried and tested using a cost function. The split with the best cost (lowest cost because we minimize cost) is selected.

All input variables and all possible split points are evaluated and chosen in a greedy manner (e.g. the very best split point is chosen each time).

For regression predictive modeling problems the cost function that is minimized to choose split points is the sum squared error across all training samples that fall within the rectangle:

$$\text{sum}(y - \text{prediction})^2$$

Where y is the output for the training sample and prediction is the predicted output for the rectangle.

For classification the Gini index function is used which provides an indication of how “pure” the leaf nodes are (how mixed the training data assigned to each node is).

$$G = \text{sum}(pk * (1 - pk))$$

Where G is the Gini index over all classes, pk are the proportion of training instances with class k in the rectangle of interest. A node that has all classes of the same type (perfect class purity) will have $G=0$, where as a G that has a 50-50 split of classes for a binary classification problem (worst purity) will have a $G=0.5$.

For a binary classification problem, this can be re-written as:

$$G = 2 * p1 * p2$$

or

$$G = 1 - (p1^2 + p2^2)$$

The Gini index calculation for each node is weighted by the total number of instances in the parent node. The Gini score for a chosen split point in a binary classification problem is therefore calculated as follows:

$$G = ((1 - (g1_1^2 + g1_2^2)) * (ng1/n)) + ((1 - (g2_1^2 + g2_2^2)) * (ng2/n))$$

Where G is the Gini index for the split point, g1_1 is the proportion of instances in group 1 for class 1, g1_2 for class 2, g2_1 for group 2 and class 1, g2_2 group 2 class 2, ng1 and ng2 are the total number of instances in group 1 and 2 and n are the total number of instances we are trying to group from the parent node.

[D] Stopping Criterion

The recursive binary splitting procedure described above needs to know when to stop splitting as it works its way down the tree with the training data.

The most common stopping procedure is to use a minimum count on the number of training instances assigned to each leaf node. If the count is less than some minimum then the split is not accepted and the node is taken as a final leaf node.

The count of training members is tuned to the dataset, e.g. 5 or 10. It defines how specific to the training data the tree will be. Too specific (e.g. a count of 1) and the tree will overfit the training data and likely have poor performance on the test set.

[E] Pruning The Tree

The stopping criterion is important as it strongly influences the performance of our tree. We can use pruning after learning our tree to further lift performance.

The complexity of a decision tree is defined as the number of splits in the tree. Simpler trees are preferred. They are easy to understand, and they are less likely to overfit the data.

The fastest and simplest pruning method is to work through each leaf node in the tree and evaluate the effect of removing it using a hold-out test set. Leaf nodes are removed only if it results in a drop in the overall cost function on the entire test set. We stop removing nodes when no further improvements can be made.

More sophisticated pruning methods can be used such as cost complexity pruning (also called weakest link pruning) where a learning parameter (α) is used to weigh whether nodes can be removed based on the size of the sub-tree[15].

Implementation:

Using R we have trained the 60% of data while predicting so2 and plotted a graph State vs rms error , and again from the "State.csv" file and state specific data "MR.csv" we have plotted a graph Percentage of trained data Vs rms value.

in R,

for using svm function at first we have to include library(rpart)

```
r1<-rpart(so2 ~ no2 + spm + rspm + pm2_5,data=st_train,method = "anova")
```

where st_train is the trained data set

Here r1 is the trained model, this model is further used to predict the data .rpart model is the inbuilt regression tree model in R. For predicting ,we used the function,

```
p1<(predict(m1,data.frame("no2"=st_test[,2],"spm"=st_test[,3],"rspm"=st_test[,4],"pm2_5"=st_test[,5])))
```

predict is a generic function for predictions from the results of various model fitting functions. The function invokes particular *methods* which depend on the class of the first argument.

EXPERIMENTAL DATA

4.1. CALCULATION ON AIR POLLUTION DATA 37 STATES OF INDIA:

In this section the experimental graphs have been provided. These graphs are plotted using three different machine learning algorithms.

4.1.1 Percentage of trained data vs rms error(while predicting SO₂)

LINEAR REGRESSION:

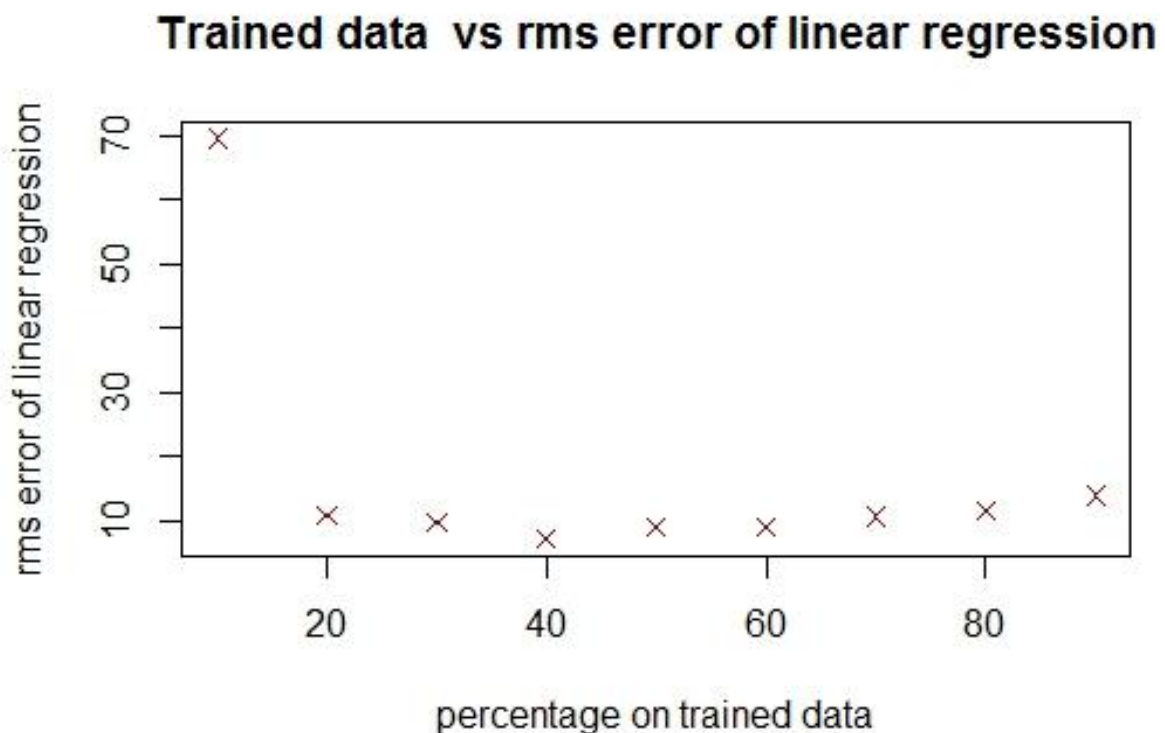


FIG 4.1

SUPPORT VECTOR MACHINE(SVM):

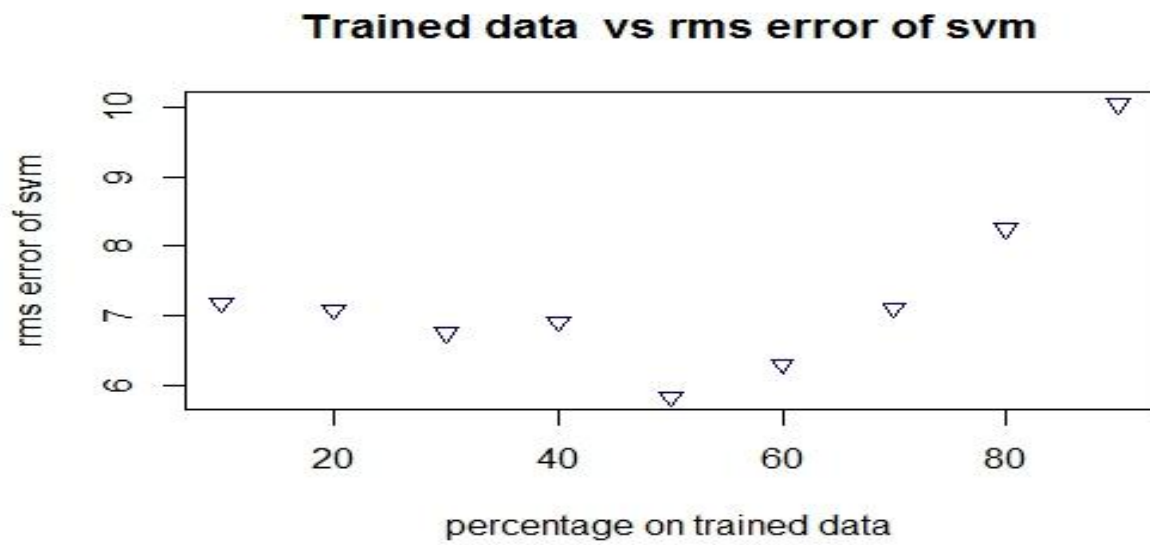


FIG 4.2

REGRESSION TREE:



FIG 4.3

4.1.2 THE GRAPH OF STATE VS RMS ERROR GRAPH (FOR SO2 PREDICTION)

LINEAR REGRESSION:

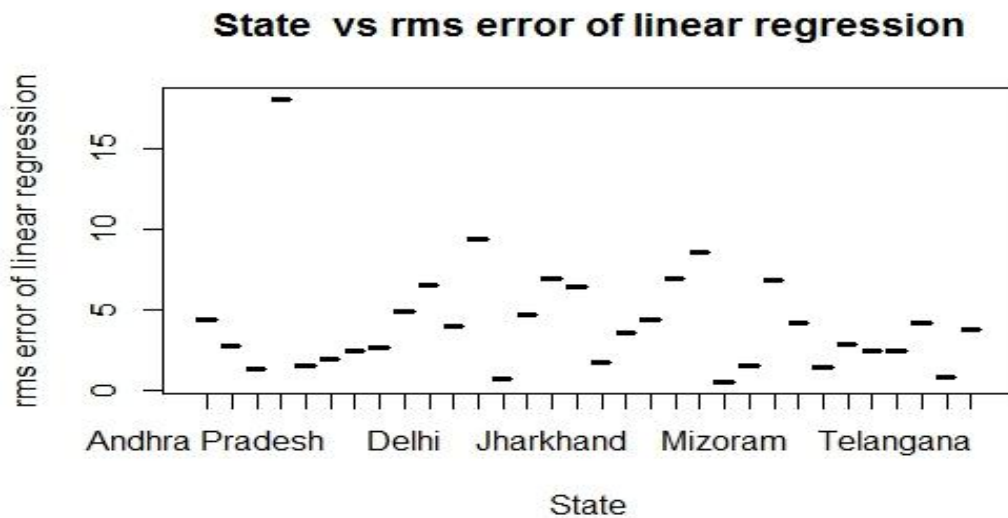


FIG 4.4

SUPPORT VECTOR MACHINE(SVM):

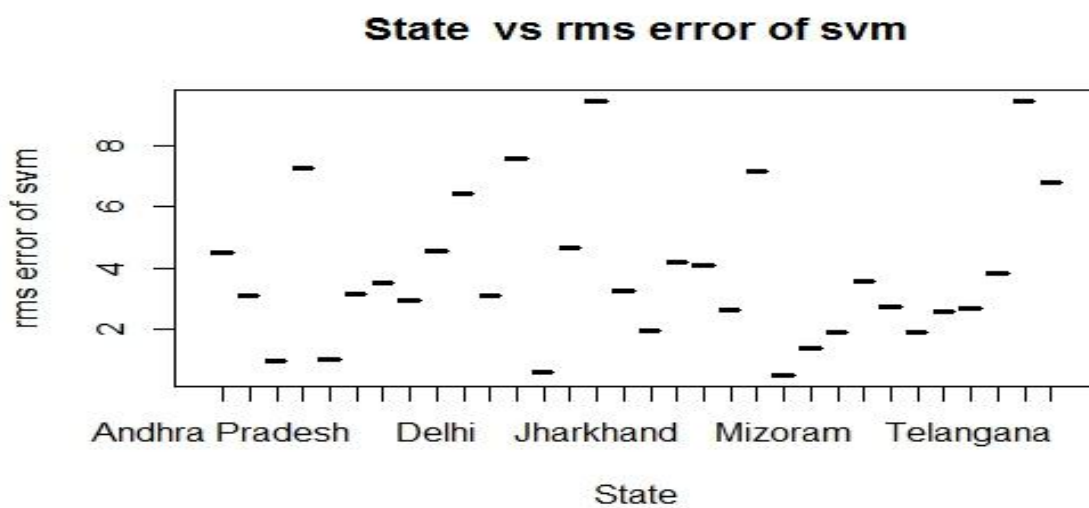


FIG 4.5

REGRESSION TREE:

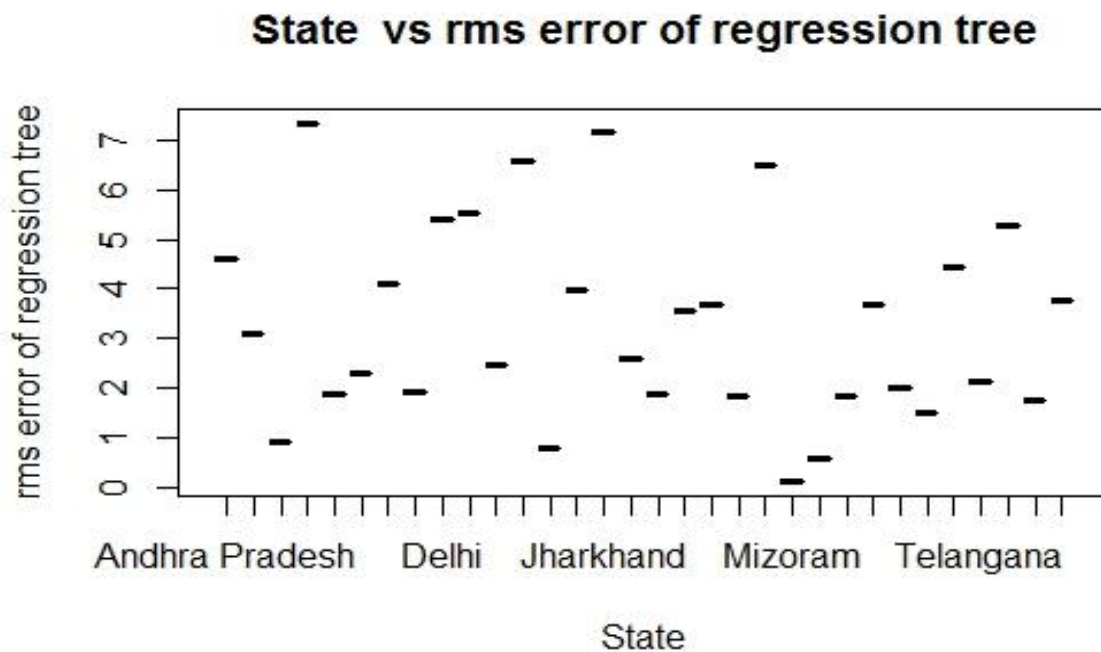


FIG 4.6

4.1.3 ANALYSIS

The above graphs are drawn through using plot function.

plot is generic function for plotting of R objects.

The first three graph(i.e. FIG 4.1, FIG 4.2 ,FIG 4.3) is made from the "State.csv" file data, where the next three graphs(i.e. FIG 4.4, FIG 4.4 ,FIG 4.5) are made on percentage change of trained data vs rms error of prediction of SO₂.

From here also we can conclude that Regression tree gives better result while considering the other as the max rms error of regression tree here is 8.5 and it gives best result when we take 60% data as trained data.

The next three graphs represent the rms error of predicting SO₂(considering 60% data as trained data) vs STATE in three different method. Among those three method REGRESSION TREE gives more accurate result as its highest rms value is approx 7.

4.2 CALCULATION FOR THE STATE MAHARASHTRA:

Among all the states Maharashtra contains the maximum record. So We have predicted So2 and No2 from this record. Mainly the data is collected from the the filename "MR.csv", which we have already created previously and then several graphs are plotted in three different machine learning method.

4.2.1 So2 prediction:

LINEAR REGRESSION:

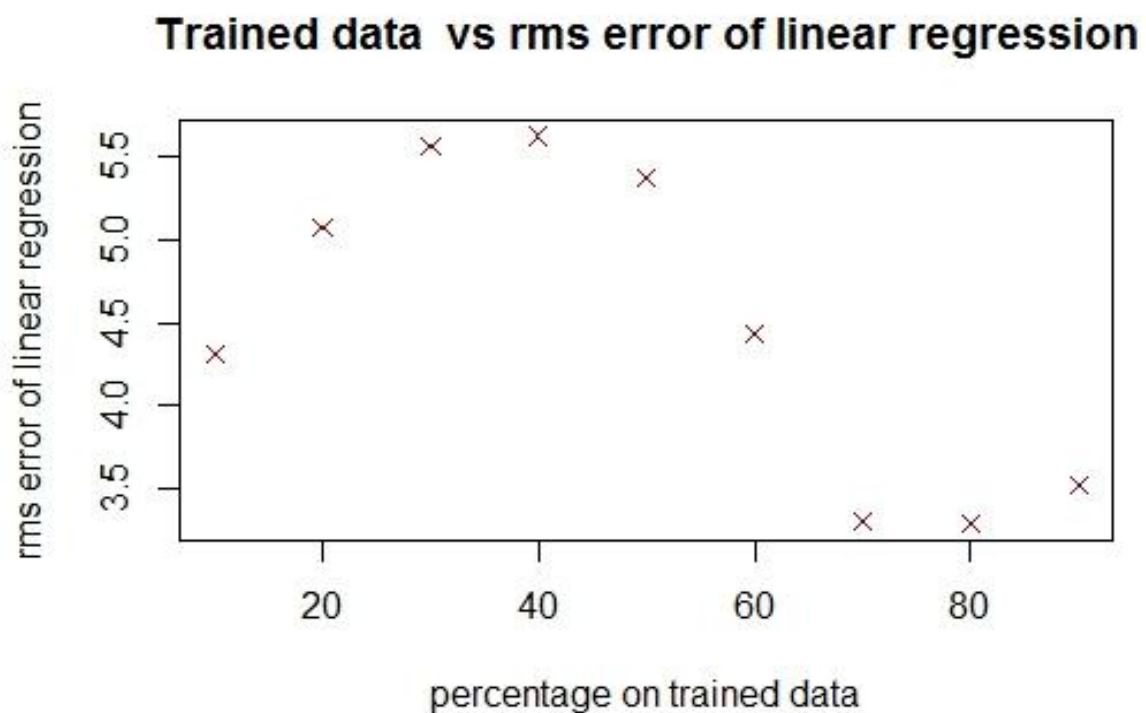


FIG 4.7

Support Vector Machine:

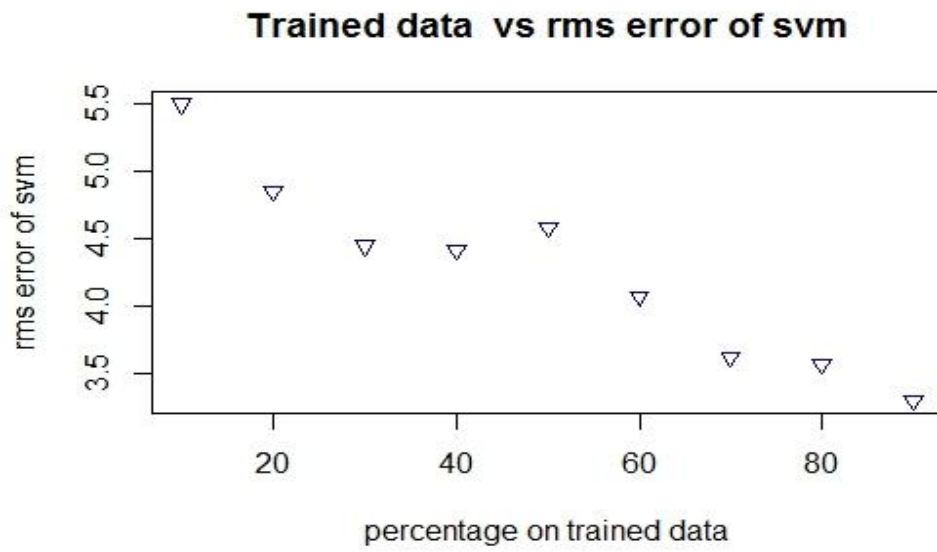


FIG 4.8

REGRESSION TREE

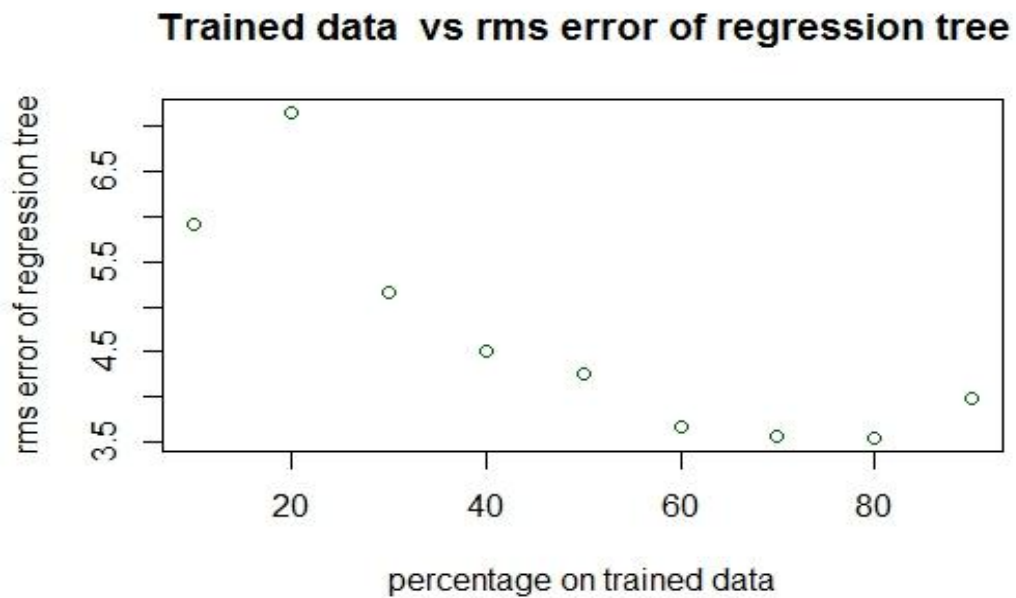


FIG 4.9

4.2.2 No2 prediction:

LINEAR REGRESSION:

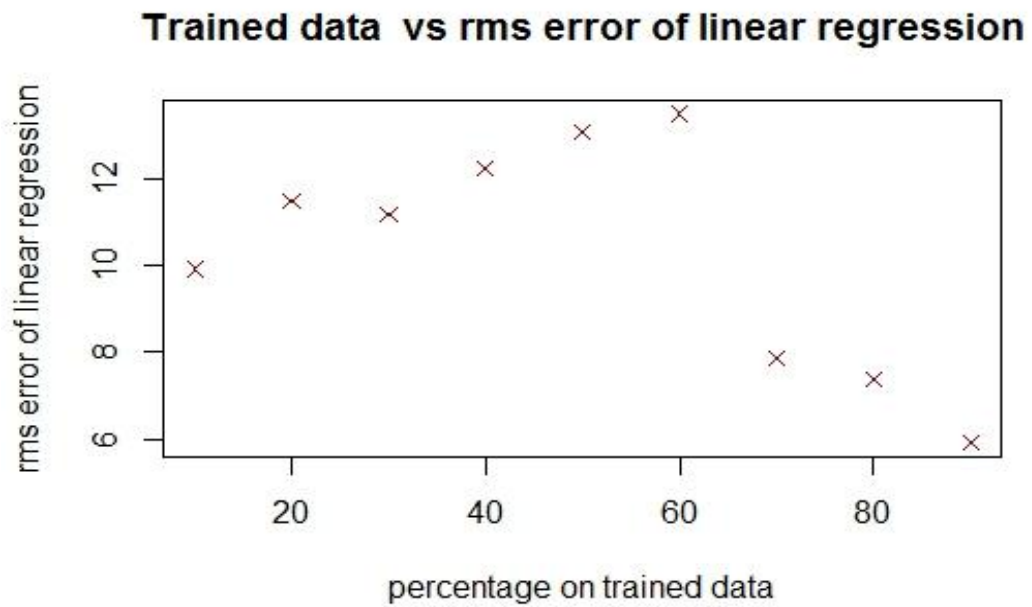


FIG 4.10

Support Vector Machine(SVM):

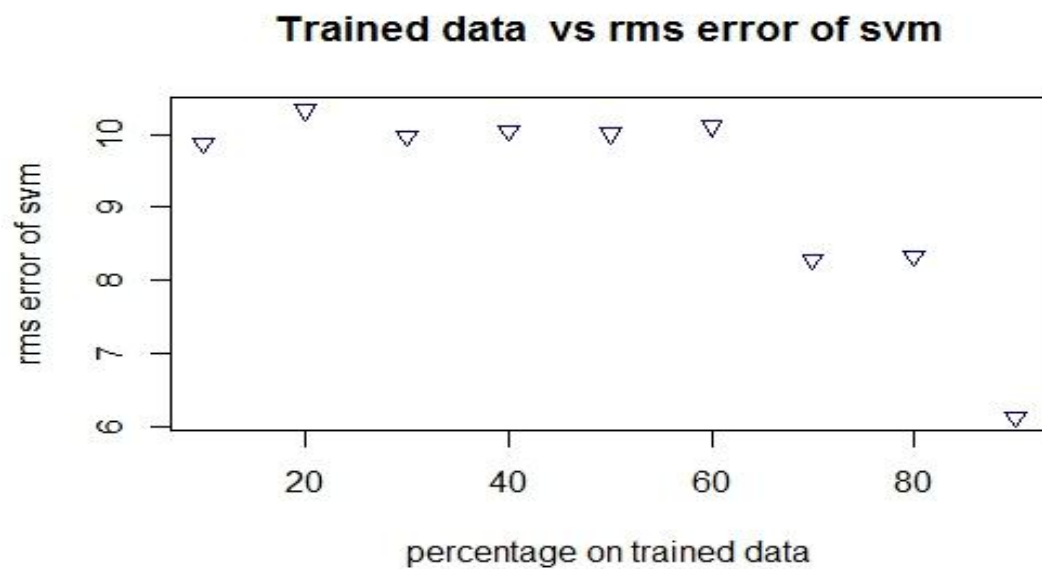


FIG 4.11

Regression Tree:



FIG 4.12

4.2.3 ANALYSIS

The above three graphs(i.e FIG 4.7,FIG 4.8,FIG 4.9) predicts SO_2 and the next three graph(i.e FIG 4.10,FIG 4.11,FIG 4.12) predict NO_2 in three different Machine learning algorithm. From the above graphs it can be said that the algorithm gives better results when 60%-70% data is trained though in this case the three different methods are giving more or less same result on a given trained percentages.

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

At the end we have found that Regression tree works better than linear regression and Support Vector Machine(SVM),when the experiment is done on the whole data base. But when we applied the same method on state specific data those three algorithm gives more or less same result. But we can conclude that it gives better result when 60%-70% data is trained. From the above observation we can also conclude that machine learning algorithm always works better on a large data.

5.2 FUTURE WORK

Using the dataset of Indian air quality , another approaches of machine learning like KNN regression, Naive Bayes Classifier can be made while predicting So₂ and No₂.Studies can be done on the basis of location (i.e whether residential or industrial) using Classification. Thus one can be aware of the air pollution of India.

REFERENCES

1. Zhu, Dixian, Changjie Cai, Tianbao Yang, and Xun Zhou. "A Machine Learning Approach for Air Quality Prediction: Model Regularization and Optimization." *Big Data and Cognitive Computing* 2, no. 1 (2018): 5.
2. Kleine Deters, Jan, Rasa Zalakeviciute, Mario Gonzalez, and Yves Rybarczyk. "Modeling PM2.5 urban pollution using machine learning and selected meteorological parameters." *Journal of Electrical and Computer Engineering* 2017 (2017).
- 3 Anuradha, C., and T. Velmurugan. "A comparative analysis on the evaluation of classification algorithms in the prediction of students performance." *Indian Journal of Science and Technology* 8, no. 15 (2015).
4. Alesheykh, Rohollah. "Comparative Analysis of Machine Learning Algorithms with Optimization Purposes." *Biquarterly Control and Optimization in applied Mathematics* 1, no. 2 (2016): 63-75.
5. Anantvir Singh Romana "A Comparative Study of Different Machine Learning Algorithms for Disease Prediction" Research Article July 2017
6. https://www.sas.com/en_in/insights/analytics/machine-learning.html
7. **Website from where the dataset is collected-**
<https://www.kaggle.com/shrutibhargava94/india-air-quality-data>
8. <http://statweb.stanford.edu/~susan/courses/s60/split/node60.html>
9. <http://www.stat.yale.edu/Courses/1997-98/101/linreg.htm>
10. <http://onlinestatbook.com/2/regression/intro.html>

11. <https://www.statisticssolutions.com/what-is-multiple-linear-regression>
12. <http://www.statsoft.com/Textbook/Support-Vector-Machines#Regression%20SVM>
13. <https://machinelearningmastery.com/support-vector-machines-for-machine-learning/>
14. <https://www.solver.com/regression-trees>
15. <https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/>
16. <http://www.statsoft.com/Textbook/Classification-and-Regression-Trees>