**Pseudo Code and Method**

**Sravan Kumar Borra**

**Introduction**

Internet of things and Big data are the main fields we need to concentrate on for building a Smart city. All the main research is running on collecting data and analyzing it. The more we collect data and analyze using the big data techniques and machine learning techniques, the more cities will be smart. Smart cities are nothing but the cities that will run on internet. Smart city are typically based on connectivity where everything is connected to internet. Insights for these cities are based on big data. The data is collected from different sources such as traffic, library events, pollution etc. All this data is collected and analyzed to provide better services to the people living in the city.

In this article, we are trying to build a model which is suitable for smart cities. Predicting the weather conditions, analyzing the traffic, parking areas prediction, pollution readings etc. In this article we are specifically concentrating on weather conditions. We are going to predict maximum and minimum temperature, wind speed, humidity etc. We are going to use weather dataset provided by City pulse. We have the weather data for Aarhus city in Denmark.

**Data Set**

Weather data set of Aarhus has seven sub data sets. They are humidity, pressure, wind speed, wind direction, temperature etc. Each data set contains the readings from February 2014 to June 2014 and august 2014 to September 2014. All these data sets were provided in JSON format. We are using Zeppelin to perform our project and predict the weather conditions. We are going to build a model which will help us to predict the weather for Aarhus city in Denmark. We are going to use Artificial Neural Networks and Support vector machines to predict the weather for the Aarhus city.

**Method**

Artificial Neural Networks and Support vector Machines are the methods used in Data mining and Machine learning. We are going to use those techniques to predict the maximum and minimum temperatures for a particular day, for a particular hour a day. We are also going to predict the humidity scale for a particular day. We are going to predict the wind speed and direction of a particular day.

We have two sets of data. We use one of the data set to train the model and use another set to test the model and we can also test the model by predicting for upcoming days. Data Cleaning, implementation of methods on data set are going to be performed in Zeppelin platform. We are going to use Python language in Zeppelin to clean the data and run the methods on data set.

**Support Vector Machines:**

SVM works on statistical learning and it is also known as a kernel machine. It is a machine learning algorithm used for classification purpose and regression purpose. If we have n observations in a data set, Support vector machines can be explained as the simple co-ordinates which is related to each and every observation. Support vector machines segregates very two important parts of a plot. They are hyper plane and line.

There are five scenarios in understanding Support Vector Machines. In scenario one, or in step one we need to first classify the hyper plane. A plot can have two or more hyper planes. We need to classify those hyper planes and we have to select the best two planes which can classify our observation given in data.

In scenario 2, we might have lines and hyper planes. All the lines are segregating the planes equally and well. Now we have to select which hyper plane is better to segregate our observations. Now we have to select a plane by using the distance which is known as Margin. The distance have to be maximized between the hyper planes and nearest data point. The closest hyper planes can be excluded from the equation and we have to select the planes which is not so nearest to the observations when compared to the other two hyper planes. There is another reason to select the hyper plane which has margin higher than other two planes is robustness. Miss classification might be a problem if we select the hyper plane with low margin and this will affect our prediction.

In scenario 3, we need to understand that selecting a hyper plane based on its margin is not only the reason to select. A hyper plane should classify the classes equally. If a hyper plane having higher margin but not classifying the classes as efficiently as other hyper plane, then it is not right hyper plane to select. A hyper plane which classifies the classes correctly and have has better margin when compared to the other hyper planes is better for selection.

In scenario 4, we will learn about outliers. If the observations have outliers then it will be difficult to classify by using hyper plane. It is because we cannot find a plane which can classify the classes accurately. Support vector machines has a feature to eliminate and ignore the outliers. It will exclude the outliers out of the equation and divide the classes into hyper plane ignoring the outliers.

In scenario 5, we will see about the classes where we cannot divide those using linear hyper planes. Support vector machines will form an equation where the distances are squared. The new equation will be the sum of square of both distances. The trick of forming a new equation is known as Kernel trick. That is the reason why support vector machines is also known as Kernel method. It is an inbuilt function in support vector machines. These types of methods will make the lower dimensional space to a higher dimensional which will help us to classify the classes using linear classification or linear hyper planes.

**Pseudo Code:**

In this article we selected weather data of Aarhus to predict the weather conditions. We have different sets of data. We have data sets for wind speed, humidity, dew point, temperature etc. We are predicting each parameter separately and we are going to predict each parameter using its own kind of data set. The pseudo code for predicting those parameters have been given below.

Before going to pseudo code, we need to discuss about the language we are going to use and the platform we are going to use. We are using Python language and we are using that language in Zeppelin platform where we also can use R, SQL, and data visualization. We can use any of those for performing statistical operations and visualizations.

We need to import some libraries such as JSON, Datetime, SVM, sklearn, matplotlib.pyplot etc. We are going to see all these libraries and many other libraries as we progress in to the code. We need to learn about these libraries and their functioning.

.JSON: import.json command will help us to import json parameters into our platform. We are using JSON library because our data is in JSON format. We need to import our data in JSON format and we have to clean the data into two columns and several rows. Our data has date time of particular day and there is a value assigned to that particular date and time. So we need to clean the data as a list which have date time as keys and the value assigned to those keys. All this will be done by importing JSON library.

We are also importing datetime library in python as our data entries are in date and time format. By importing datetime library we can split the key into two keys where we can have date and time split by a character or space. All this entries after cleaning the data is assigned in a list where we will get the data as a date and a particular time with values assigned to those keys.

**Pseudo code for Dew Point:**

**#Data Cleaning**

Importing json library

Importing datetime library

Opening the file by specifying the file path

Assign the lines by reading all the lines in data

Creating a list

for each in lines:

data = json.loads(each)

li=[]

for a,b in data.items():

Splitting the items in data where we have “T”

Converting the list into a string

dt=datetime.datetime.strptime(x,"%Y-%m-%d" "%H:%M:%S")

li.append([dt,b])

bigLi.append(li)

**#Performing Support Vector Machines**

Import library

We have predictor as K and target as L for train data and M is the test predictor for test

Creating the object for SVM classification

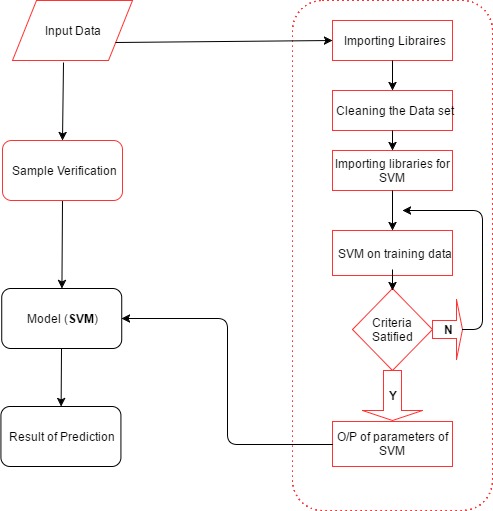
Dewptm\_model = svm.svc(kernel=’linear’, c=1, gamma=1)

Dewptm\_model.fit/(k,l)

Dewptm\_model.score(k,l)

Predicted output for testing the data

**Flow chart**

****

**Pseudo code for Humidity**

**#Data Cleaning**

Importing json library

Importing datetime library

Opening the file by specifying the file path

Assign the lines by reading all the lines in data

Creating a list

for each in lines:

data = json.loads(each)

li=[]

for a,b in data.items():

Splitting the items in data where we have “T”

Converting the list into a string

dt=datetime.datetime.strptime(x,"%Y-%m-%d" "%H:%M:%S")

li.append([dt,b])

bigLi.append(li)

**#Performing Support Vector Machines**

Import library

We have predictor as A and target as B for train data and J is the test predictor for test

Creating the object for SVM classification

hum\_model = svm.svc(kernel=’linear’, c=1, gamma=1)

hum\_model.fit/(a,b)

hum\_model.score(a,b)

Predicted output for testing the data

**Pseudo code for Temperature**

**#Data Cleaning**

Importing json library

Importing datetime library

Opening the file by specifying the file path

Assign the lines by reading all the lines in data

Creating a list

for each in lines:

data = json.loads(each)

li=[]

for a,b in data.items():

Splitting the items in data where we have “T”

Converting the list into a string

dt=datetime.datetime.strptime(x,"%Y-%m-%d" "%H:%M:%S")

li.append([dt,b])

bigLi.append(li)

**#Performing Support Vector Machines**

Import library

We have predictor as i and target as j for train data and k is the test predictor for test

Creating the object for SVM classification

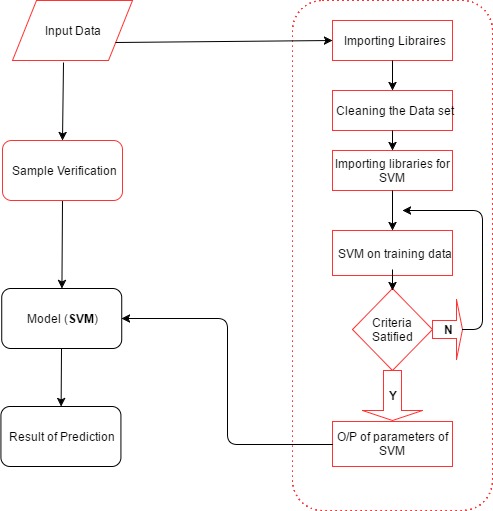
temp\_model = svm.svc(kernel=’linear’, c=1, gamma=1)

temp\_model.fit/(i,j)

temp\_model.score(i,j)

Predicted output for testing the data

**Flow chart**

****

**Pseudo code for Wind speed**

**#Data Cleaning**

Importing json library

Importing datetime library

Opening the file by specifying the file path

Assign the lines by reading all the lines in data

Creating a list

for each in lines:

data = json.loads(each)

li=[]

for a,b in data.items():

Splitting the items in data where we have “T”

Converting the list into a string

dt=datetime.datetime.strptime(x,"%Y-%m-%d" "%H:%M:%S")

li.append([dt,b])

bigLi.append(li)

**#Performing Support Vector Machines**

Import library

We have predictor as K and target as L for train data and M is the test predictor for test

Creating the object for SVM classification

winspd\_model = svm.svc(kernel=’linear’, c=1, gamma=1)

winspd\_model.fit/(k,l)

winspd\_model.score(k,l)

Predicted output for testing the data

**Pseudo code for Pressure**

**#Data Cleaning**

Importing json library

Importing datetime library

Opening the file by specifying the file path

Assign the lines by reading all the lines in data

Creating a list

for each in lines:

data = json.loads(each)

li=[]

for a,b in data.items():

Splitting the items in data where we have “T”

Converting the list into a string

dt=datetime.datetime.strptime(x,"%Y-%m-%d" "%H:%M:%S")

li.append([dt,b])

bigLi.append(li)

**#Performing Support Vector Machines**

Import library

We have predictor as E and target as F for train data and G is the test predictor for test

Creating the object for SVM classification

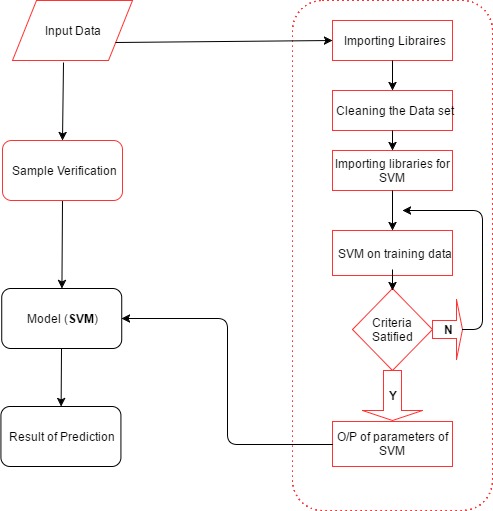
pres\_model = svm.svc(kernel=’linear’, c=1, gamma=1)

pres\_model.fit/(e,f)

pres\_model.score(e,f)

Predicted output for testing the data

**Flow chart**

****

**Artificial Neural Networks:**

We all know about neural network in our brain. There are many programs and functions in our body are being controlled by that neural network in our brain. There are many debates running on this topic whether computers can handle neural network concept in programming and building algorithms. From the past few years ANN has been the most unique and efficient method of statistical learning which is used to predict the data accurately. ANN’s uses the time series provided by the data set and analyze the data set by creating different layers in the program.

In our article we are trying to implement Artificial Neural networks method along with Support Vector Machines. We are going to predict the temperatures and humidity etc. as we predicted using SVM. We are going to compare both results obtained from ANN and SVM. We are going to opt the method which will give us the accurate results for our prediction.