Air Quality Analysis

Sudhir Panda ,Chitaranjan Pradhan, Bikram Pradhan, Sahil Pradhan

24th September 2017

**ABSTRACT**

The dataset contains the responses of a gas multi-sensor device deployed on the field in an Italian city. Hourly responses averages are recorded along with gas concentrations references from a certified analyzer**.**

DATASET CHARACTERISTICS:- Multivariate Time Series

ATTRIBUTE CHARACTERISTICS:- REAL

ASSOCIATED TASK:- REGRESSION

NUMBER OF INSTANCES:-9358

NUMBER OF ATTRIBUTES:-15

MISSING VALUES:-YES

**1.Introduction**

* 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 were 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) and were provided by a co-located reference certified analyzer. Evidences of cross-sensitivities as well as both concept and sensor drifts are present as described in De Vito et al., Sens. And Act. B, Vol. 129,2,2008 (citation required) eventually affecting sensors concentration estimation capabilities. Missing values are tagged with -200 value.   
  This dataset can be used exclusively for research purposes. Commercial purposes are fully excluded.

**2.Data Sets**

* Date (DD/MM/YYYY)
* Time (HH.MM.SS)
* True hourly averaged concentration CO in mg/m^3 (reference analyzer)
* PT08.S1 (tin oxide) hourly averaged sensor response (nominally CO targeted)
* True hourly averaged overall Non Metanic HydroCarbons concentration in microg/m^3 (reference analyzer)
* True hourly averaged Benzene concentration in microg/m^3 (reference analyzer)
* PT08.S2 (titania) hourly averaged sensor response (nominally NMHC targeted)
* True hourly averaged NOx concentration in ppb (reference analyzer)
* PT08.S3 (tungsten oxide) hourly averaged sensor response (nominally NOx targeted)
* True hourly averaged NO2 concentration in microg/m^3 (reference analyzer)
* PT08.S4 (tungsten oxide) hourly averaged sensor response (nominally NO2 targeted) 11 PT08.S5 (indium oxide) hourly averaged sensor response (nominally O3 targeted)

**3. Variable Identification**

TYPES OF VARIABLE:-Prediction Variable:- CO(GT), NMMHC(GT), NO(GT),

NOx(GT), C6H6 (GT)

Target Value:-Temp, RH(Relative Humidity),

AH(Absolute Humidity

VARIABLE CATEGORIES:- Continuous:- ALL OF THE ATTRIBUTE

**4.Loading Data**

library("neuralnet")

library("ggplot2")

#Reading CSV FILES .

feed<-read.csv("feed.csv")

names(feed)

[1] "Date" "Time" "CO.GT." "PT08.S1.CO."

"NMHC.GT." "C6H6.GT."

[7] "PT08.S2.NMHC." "NOx.GT." "PT08.S3.NOx."

"NO2.GT." "PT08.S4.NO2." "PT08.S5.O3."

[13] "T" "RH" "AH"

# Checking Number of Columns and Rows

dim(feed1)

[1] 9357 15

#Compactly display the internal **str**ucture of an Data Sets

str(feed1)

'data.frame': 9357 obs. of 15 variables:

$ Date : Factor w/ 391 levels "2004-03-10","2004-03-11",..: 1

1 1 1 1 1 2 2 2 2 ...

$ Time : Factor w/ 24 levels "0.00.00","1.00.00",..: 11 12 14 1

5 16 17 1 2 13 18 ...

$ CO.GT. : num 2.6 2 2.2 2.2 1.6 1.2 1.2 1 0.9 0.6 ...

$ PT08.S1.CO. : num 1360 1292 1402 1376 1272 ...

$ NMHC.GT. : num 150 112 88 80 51 38 31 31 24 19 ...

$ C6H6.GT. : num 11.9 9.4 9 9.2 6.5 4.7 3.6 3.3 2.3 1.7 ...

$ PT08.S2.NMHC.: num 1046 955 939 948 836 ...

$ NOx.GT. : num 166 103 131 172 131 ...

$ PT08.S3.NOx. : num 1056 1174 1140 1092 1205 ...

$ NO2.GT. : num 113 92 114 122 116 ...

$ PT08.S4.NO2. : num 1692 1559 1555 1584 1490 ...

$ PT08.S5.O3. : num 1268 972 1074 1203 1110 ...

$ T : num 13.6 13.3 11.9 11 11.2 11.2 11.3 10.7 10.7 10

$ RH : num 48.9 47.7 54 60 59.6 59.2 56.8 60 59.7 60.2 .

$ AH : num 0.758 0.726 0.75 0.787 0.789 ...

**5. Missing Value treatment**

# Handle missing values effectively is a required step to reduce bias

and to produce powerful models

cat("CO before","\n")

nrow(feed1[feed1$CO.GT.<0, ])

cat("after","\n")

m<-mean(feed1$CO.GT.[feed1$CO.GT.#0])

m

for(i in 1:9357)

{

c<-feed1$CO.GT.[i]

##cat(c,"\n")

if(c<0)

feed1[i,3]=m

}

nrow(feed1[feed1$CO.GT.<0, ])

m=0

Conclusion:- we got the conclusion that the value of nrow

before CO values is 0.And then after finding the mean value it gives the outpt 2.15275

cat("CO before","\n")

CO before

# nrow(feed1[feed1$CO.GT.<0, ])

[1] 0

# cat("after","\n")

after

# m<-mean(feed1$CO.GT.[feed1$CO.GT.#0])

# m

[1] 2.15275

# for(i in 1:9357)

+ {

+ c<-feed1$CO.GT.[i]

+ ##cat(c,"\n")

+ if(c<0)

+ feed1[i,3]=m

+ }

# nrow(feed1[feed1$CO.GT.<0, ])

[1] 0

**6. Outlier’s**

# outlier is defined as a observation which stands far away from the

most of other observations. Often a outlieris present due to the

measurement error.

#outlier's

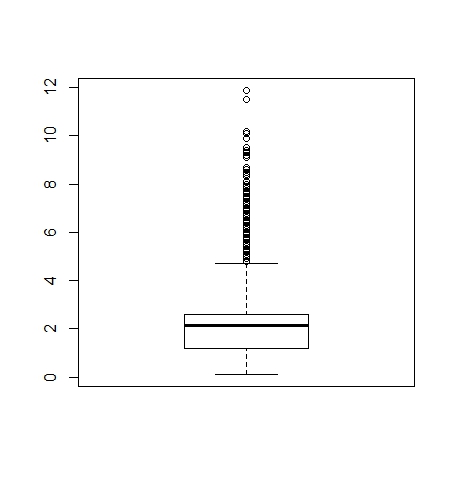
summary(feed1$CO.GT.)

boxplot(feed1$ CO.GT.)

v1<-boxplot.stats(feed1$CO.GT.)

#str(v1)

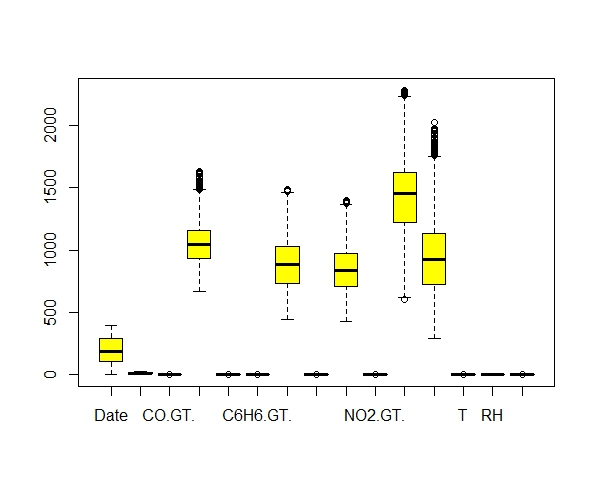
l1<-length(v1$out)



# It store all the data of q in the feed2 and by the help of the boxplot we

Found all the outlier in given data sets.

feed2=q

boxplot(feed2,col="yellow")

**7. Normalization**

# Data normalization is to reduce and even eliminate data redundancy,

an important consideration for application developers because it is

incredibly difficult to stores objects in a relational database that

maintains the same information in several places.

#normalize

feed3=feed2

max\_CO=max(feed2$CO.GT.)

min\_CO=min(feed2$CO.GT.)

feed2$CO.GT.[1:10]

for(i in 1:7712){

feed2$CO.GT.[i]=(feed2$CO.GT.[i]-min\_CO)/(max\_CO-min\_CO)

}

feed2$CO.GT.[1:10]

Output :-

In this above computation we got the output maximum input and

minimum input value frome the data set.

[1] 0.12077295 0.09178744 0.10144928 0.10144928 0.07246377

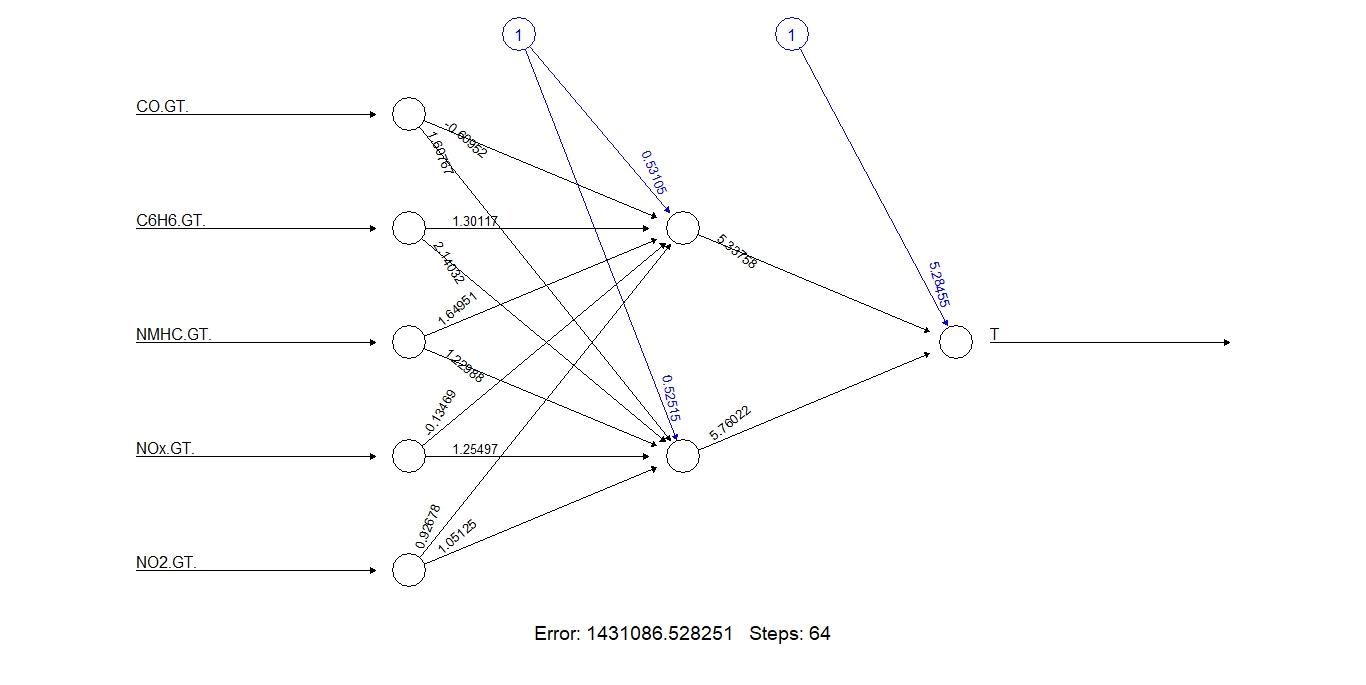
0.05314010 0.05314010 0.04347826 0.03864734

[10] 0.02415459

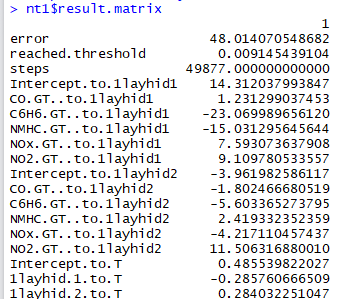
**8.Neuralnet**

#neuralnet is used to train neural networks using backpropagation,

re silient backpropagation

* nt1 <- neuralnet(T~CO.GT.+C6H6.GT.+NMHC.GT.+NOx.GT.+NO2.GT., data = feed2[1:5398,1:15], hidden=2,rep=2,learningrate =0.1,algorithm = "rprop+",err.fct = "ce", linear.output=TRUE)
* ****plot(nt1)

**#Here We have Taken only 70% of the data set for Training the Neuralnet Model and Rest 30% for testing**

****

**9.Testing Value**

#In this Testing value we finding the actual and predicted value from

a data sets of air quality.

r1<-feed2$CO.GT.[5399:7712]

r3<-feed2$NMHC.GT.[5399:7712]

r2<-feed2$C6H6.GT.[5399:7712]

r4<-feed2$NOx.GT.[5399:7712]

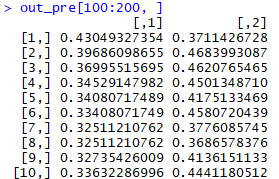
r5<-feed2$NO2.GT.[5399:7712]

nt1.output<-compute(nt1,covariate =cbind(r1,r2,r3,r4,r5))

nt1.output$net.result[1:10]

out\_pre=cbind(feed2$T[5399:7712],nt1.output$net.result)

out\_pre[100:200, ]

****

**10.FINDING THE RMSE VALUE:-**

#finding the rmse value of given data sets

#error

err<-c(1:2314)

for(i in 1:2314){

err[i]=n1.output$net.result[i]-feed2$T[i+5398]

}

err

length(err)

#RMSE

err2<-err^2

length(err2)

rmse<-sqrt(mean(err2))

rmse

mean(n1.output$net.result)

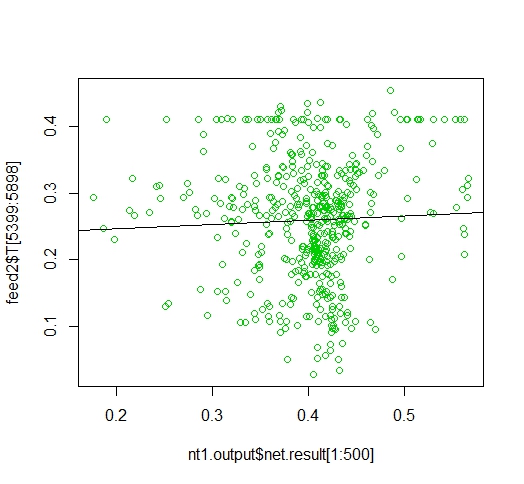
mean(feed2$T[5399.7712])

**#after normalizing the data set we found the rmse value from the data set**

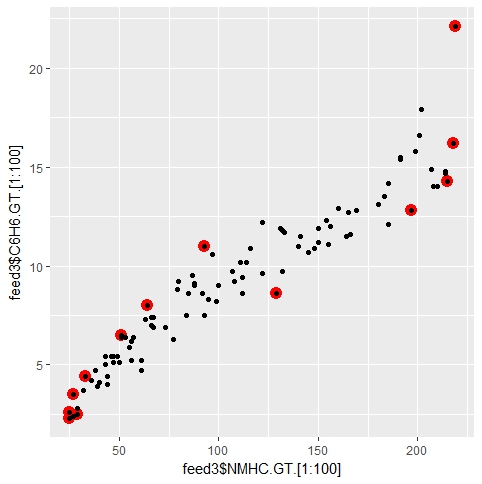
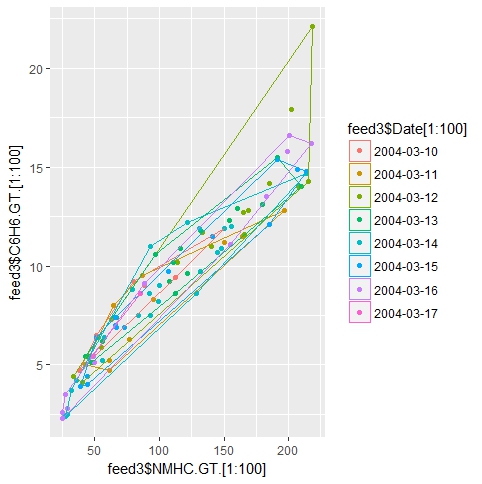




**11.Actual vs Predicted value:-**

* plot(nt1.output$net.result[1:500],feed2$T[5399:5898],col=3)
* x=nt1.output$net.result[1:500]
* y=feed2$T[5399:5898]
* abline(lm(y~x))

**12.Graph:-**



CONCLUSION

From the above analysis we predicted the value of temperature by training the neuralnet model against 5 input parameters and found out the RMSE error i.e 0.222.