Fine Particular Matter Air Pollution in the United States

Collura Mattia, Corsetti Christian, Tonoli Rossana

Contents

List of Tables

List of Plots

List of Figures

Introduction

The Clean Air Act (CAA) is the principal statute addressing air quality and was first enacted in 1955, with major revisions in 1970, 1977, and 1990. This last amendment addressed acid rain, ozone depletion, and toxic air pollution, established a national permits program for stationary sources, and increased enforcement authority. The Act requires EPA to set health-based standards for ambient air quality, sets deadlines for the achievement of those standards by state and local governments, and requires EPA to set national emission standards for large or ubiquitous sources of air pollution, including motor vehicles, power plants, and other industrial sources.

The United States Environmental Protection Agency (EPA) is a government agency of USA, which aim regards environment protection and human health. The source of data used in this project is EPA website itself, which makes them freely available.

In our analysis we will focus on PM2.5 samples, which is a fine particle air pollution; it was started being measured in 1999. In particular, these PM2.5 samples are a sort of dust and we will refer to these data in micrograms per meter cubed.

Figure 1: *President Bush signing the Clean Air Act Amendments of 1990. Standing left to right are EPA Administrator William K. Reilly, Energy Secretary James Watkins, and Vice President Dan Quayle.*

Targets

The aim of this project is to analyse 1999 and 2012 data in order to confirm or not the CAA expectations. In more detail, we want to see if the mean levels of air pollution decreased from 1999 to 2012.

To achieve the final result, we will consider and evaluate different aspects and some specific situations, as you will see in the next sections.

Our project is structured in different steps, as it is suggested by the project schedule.

The programming language used to implement the scripts and the analysis is R.

Operating Process

Opening up and exploring data files

First of all, we took a look to the data from EPA. As concerns 1999 data we observed 117.421 detections, while for 2012 data 1.304.287; for both we had 28 attributes, but we focused on only few of these, such as State Code, County Code, Site ID and Sample Value.

During the data extraction we took into account some structural rules in the data file, in particular:

- the comments begin with "#" and the header line is a comment
- the separation character is the "|"
- a blank string represents a missing value

In order to make more readable and easier our task we changed column names from "v1", "v2", …, "v28" to the names contained in the header line of the data file.

```
# Returns a vector of name columns that you can use in a dataframe
getNames = function(percorso){
  res = readLines(percorso)
  vectNames = unlist(strsplit(res[1], "[|]"))
  vectNames[1] = substring(vectNames[1], 3)

  nameList = vectNames
  nameList = make.names(nameList)
  return(nameList)
}
```

Figure 2: *getNames function.*

By analysis of 1999 sample values we noticed the minimum value is 0.00 $\mu g/m^3$, the maximum is 157,10 $\mu g/m^3$, there are 13.217 missing values on 117.421 total observations, so in terms of the proportion on the entire data set 11,26% is missing. The mean value of PM2.5 in 1999 records results 13,74 $\mu g/m^3$, while the median is 11,50 $\mu g/m^3$ and the standard deviation is 9,41$\mu g/m^3$.

Analysing 2012 sample values, instead, we saw that the minimum value is -10,00 $\mu g/m^3$, the maximum is 909,00 $\mu g/m^3$, the missing values are 73.133 on 1.304.287 total observations. This time the proportion of the missing values on the entire data set is 5,61%. The values of mean, median and standard deviation are respectively 9,14 $\mu g/m^3$, 7,63 $\mu g/m^3$ and 8,56 $\mu g/m^3$.

	1999	2012
Records	117.421	1.304.287
Missing Value	13.217	73.133
Percentage Missing Value	11,26%	5,61%
Mean Value $(\mu g/m^3)$	13,74	9,14
Minimum Value $(\mu g/m^3)$	0,00	-10,00
Maximum Value $(\mu g/m^3)$	157,10	909,00
Median $(\mu g/m^3)$	11,50	7,63
Standard Deviation $(\mu g/m^3)$	9,41	8,56

Table 1: *A summary of 1999 and 2012 distributions.*

The results illustrated above are easily visible in *Table 1*; we can already assume that in 2012 there is an air quality improvement, thanks to the reduction of the mean value of the PM2.5, as well as the reduction of the median and the deviation standard values.

Are missing data a problem?

As you can observe from *Table 1*, the 1999 data are a tenth of 2012, while the 1999 missing values in percent are about twice, so the most recent observations are more accurate. If the 11,26% of missing values is evenly distributed between states, counties, sites, dates it will not affect the analysis. Otherwise, if missing values are centred on a specific attribute, we might find problems during analysis. For example, if a big amount of missing data would be related to a single state, we will never know the behaviour of air pollution behavior of this state. The same argument it could be done for other attributes.

In order to better understand if missing values in 1999 data frame are a problem, we wanted to count total missing values for every state, site and county. These attributes were evaluated at this point because we will use them in the next steps of the study.

We noticed that different states had different total amount of observations, missing or not (the same for counties and sites). So we built three matrixes, respectively for Site ID, State Code and County Code and saved them with the help of the code reported in *Figure 3* and *Figure 4*.

```
analyseAttr = function(pm, attr, attrName, pm0){
  lista = split(pm, attr)
  counter = countObs(lista)
  buildMatrix(counter, lista, attrName, pm0)
}

buildMatrix = function(counter, splittaggio, attrName, pm0){
  attrNum = which(colnames(pm0) == attrName)
  mat = matrix(nrow = length(counter), ncol = 4, byrow = TRUE)

  for(i in 1 : length(splittaggio)){ # for every distinct value
    mat[i, 1] = splittaggio[[i]][1, attrNum] # State.Code or Site.ID or County.Code
    mat[i, 2] = counter[i] # Total number of missing observations for value in column 1
    # we extract from pm0 values which corresponds to NA:
    totalObs = (sapply(pm0[attrNum], fun, mat[i, 1]))
    mat[i, 3] = length(totalObs[totalObs == TRUE]) # Total number of obs. for value in column 1
    mat[i, 4] = ceiling((mat[i, 2] / mat[i, 3]) * 100) # Observation percentage
  }
  colnames(mat) = c(attrName, "NA_obs", "TOT_obs", "Perc")
  return(mat)
}
```

Figure 3: *A part of the code used in order to build matrix.*

```
## 2) Are missing data a problem?

# We try to answer to this question, making a list of matrix in which
# we can see total missing observations relative to each State/County/Site:
pmOMISS = pm0[is.na(pm0$Sample.Value),] # Here we have pm0 where PM2.5 samples are missing
colnames(pmOMISS) = nameList
attrList = list(pmOMISS$Site.ID, pmOMISS$State.Code, pmOMISS$County.Code)
myList = list()
attrName = c("Site.ID", "State.Code","County.Code")

for(j in 1 : length(attrList)){
  mat = analyseAttr(pmOMISS, attrList[j], attrName[j], pm0)
  myList[[length(myList) + 1]] = mat
  write.table(myList[[j]], file = paste(attrName[j], "mat.txt", sep = "_"),
              sep = "\t", row.names = FALSE, col.names = FALSE)
  myList[[j]] = myList[[j]][myList[[j]][, "Perc"] >= 30, ]
  # We extract some interesting values and save them in myList
}
```

Figure 4: *The code where we use functions of Figure 3, building matrix for Site ID, State Code and County Code.*

In the first column of the matrix we saved the attribute value we wanted to analyse (for example State.Code), in the second we reported the total amount of missing values for the corresponding attribute in column 1 (a single state), in the third the total amount of samples for the corresponding attribute in column 1. In column 4 we calculated percentage by dividing total missing value for total observations. These results were saved in .txt files, and you can observe State Code output (from State.Code_mat.txt file) in *Figure 5*.

1	457	3203	15
4	197	2061	10
6	1326	9656	14
8	670	1799	38
9	562	1921	30
10	266	1392	20
11	1	668	1
12	32	4743	1
13	543	3265	17
15	94	945	10
18	444	4008	12
19	74	1443	6
20	171	1492	12
21	304	2845	11
23	131	1125	12
25	752	3237	24
27	29	841	4
28	70	1541	5
29	82	2928	3
30	174	958	19
31	164	1128	15
32	64	1007	7
33	47	684	7
34	389	2058	19
36	952	2600	37
37	771	5601	14
38	91	548	17
39	1458	7880	19
42	439	4289	11
44	359	1268	29
45	417	2991	14
46	132	848	16
47	1061	3539	30
48	16	2219	1
49	159	1814	9
50	81	659	13
51	2	2143	1
54	171	1924	9
56	36	502	8
78	29	54	54

Figure 5: *Output of file State.Code_mat.txt. In the first column we have the State Code, in the second the total number of NA (missing values), in the third the total number of observations and in the fourth the percentage.*

So, in the fourth column of the matrix in *Figure 5,* we reported, for every state, the percentage of its missing values in relation to its total amount of observations. We noticed that, while some states had a very little percentage of missing values, like 1% for State.Code = 1, others were more interesting. For example, for the state which code is 78 there are 54% of missing values in its observations. So, in *Figure* 6, we printed those state rows which percentage was above 30%, and in *Figure 7* and *Figure 8* you can find, respectively, the same for County Code and Site ID.

	State.Code	NA_obs	TOT_obs	Perc
[1,]	8	670	1799	38
[2,]	9	562	1921	30
[3,]	36	952	2600	37
[4,]	47	1061	3539	30
[5,]	78	29	54	54

Figure 6: *Cut from the matrix of Figure 5: only a few states are reported, those which missing value percentage is high.*

	County.Code	NA_obs	TOT_obs	Perc
[1,]	10	29	54	54
[2,]	41	256	529	49
[3,]	93	689	1290	54
[4,]	101	612	1725	36

Figure 7: *Cut from the matrix of related to County Code: only a few counties are reported, those which missing value percentage is high.*

	Site.ID	NA_obs	TOT_obs	Perc
[1,]	28	45	120	38
[2,]	42	132	398	34
[3,]	56	59	122	49
[4,]	76	31	61	51
[5,]	94	47	105	45
[6,]	136	228	331	69
[7,]	1015	38	122	32
[8,]	1017	238	425	57
[9,]	1020	352	364	97
[10,]	1201	50	121	42
[11,]	2010	53	113	47
[12,]	2124	63	121	53
[13,]	4301	3	9	34
[14,]	9003	24	57	43

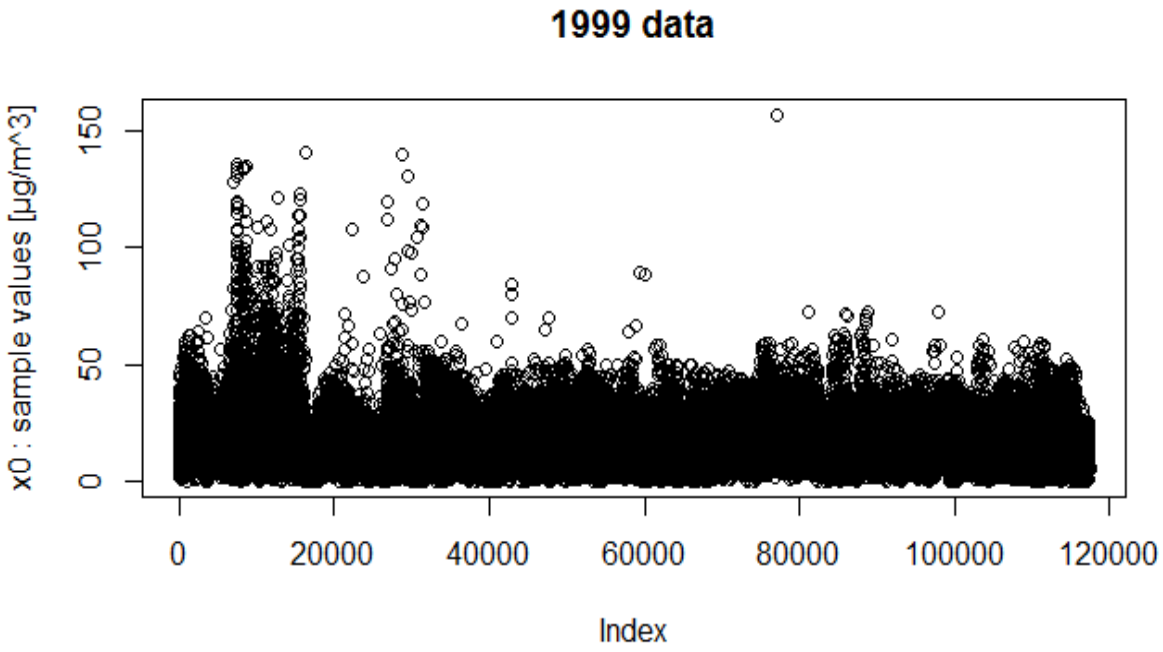
Figure 8: *Cut from the matrix of related to Site ID: only a few sites are reported, those which missing value percentage is high.*

In conclusion, we can tell that we should prefer the states, counties and sites that do not appear in *Figures 6, 7* and 8 during analysis, because these instances have a great amount of missing values. It is something that is good to know: notice, for example, that site which id is 1020 has 97% of missing values; we can't say a lot about this site.

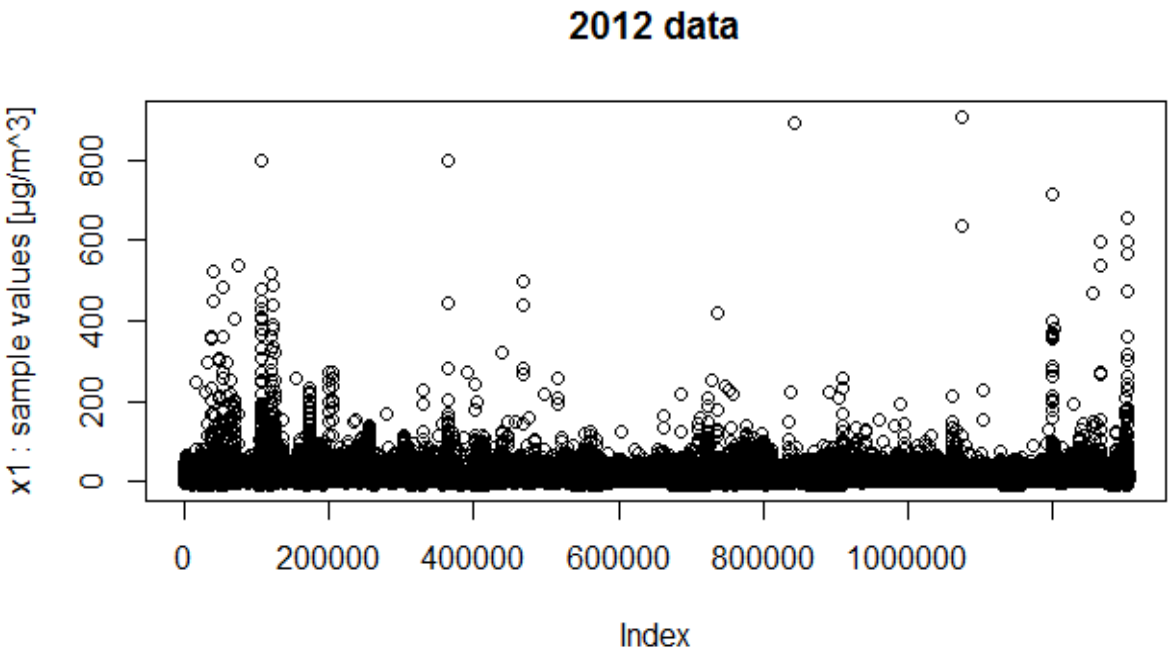
Making a quick comparison of the two different dataset looking at mean value, median and standard deviation in *Table 1*, it could be noticeable that the whole country has lower air pollution levels in 2012 than it did in 1999. For everyone of these measurements there is an improvement in the 2012 dataset.

Through a simple plot of the two dataset, it could be easier to make some

observations: *Plot 1* is a graphic representation of the PM2.5 sample values in 1999, while *Plot 2* is the same for 2012. From the plots it's quite evident that in 2012 dataset the range of outliers is bigger than in 1999 one: in fact, the mean value in *Plot 2* is 9,14 $\mu g/m^3$ but there are four samples which measure is around 900 $\mu g/m^3$, a lot between 200 $\mu g/m^3$ and 600 $\mu g/m^3$ and the maximum value is 909 $\mu g/m^3$. These measurements spread all over *Plot 2*, while in *Plot 1* the maximum value reached is 157,10 $\mu g/m^3$.

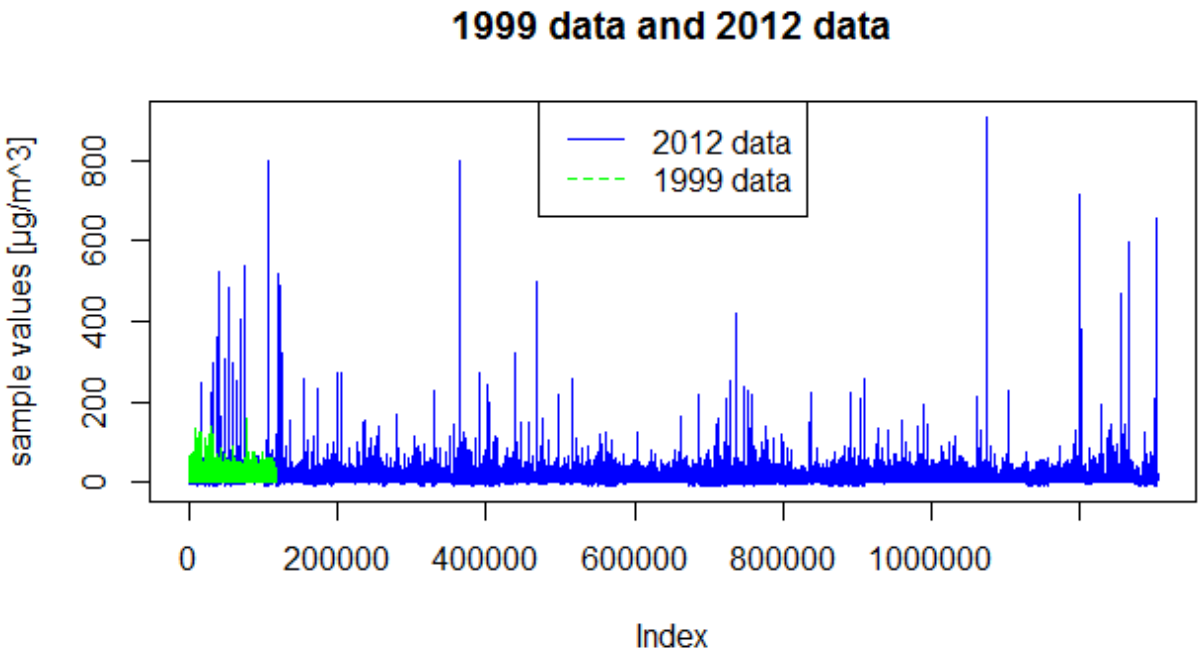


Plot 1: PM2.5 samples measured in 1999 in the United States



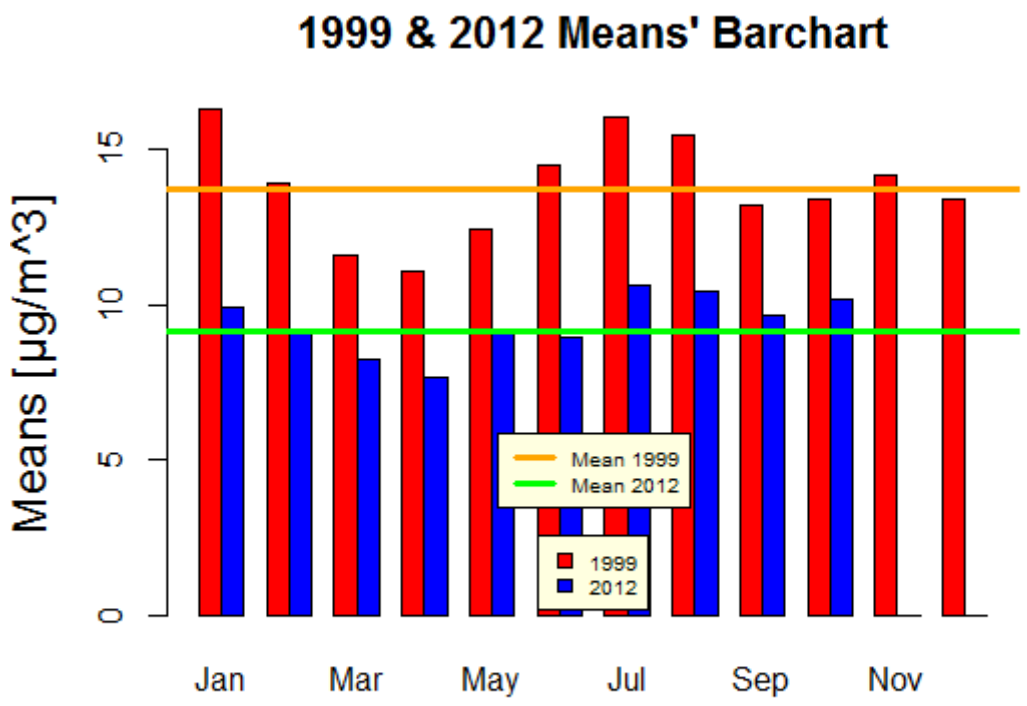
Plot 2: *PM2.5 samples measured in 2012 in the United States.*

Our purpose was to make a general quick comparison between the two dataset by observing the plots of the sample values, but from *Plot 1* and *Plot 2* it's not clear enough which dataset could be better to choose. In *Plot 3* the samples are represented in the same window and with a common range, but the only evidence is that the 2012 dataset is much bigger than the 1999 one and we still can't make a decision.



Plot 3: *PM2.5 samples measured in 2012 (blue) and in 1999 (green) in the United States.*

In *Plot 4*, we calculated mean values for every month and for both the years and compared them through a barchart: 2012 measurements are in blue, while those of 1999 are red. Finally, it's clear that we have good news for public health: except we don't have November and December samples in 2012, the computed data for every other month in 2012 dataset are lower than those of 1999.



Plot 4: *Mean values [µg/m^3] calculated for every month in 1999(red) and 2012(blue) dataset.*

Anyway, these general comparisons regard different states, county and sites, so it could be more interesting to analyse some aspects in detail, as it will be shown in the next steps.

In *Plot 1* and *Plot 2* we noticed that the 2012 dataset could have more extreme values and outliers than the 1999 one. In statistics, an outlier is a data point that significantly differs from the other data points in a sample. Often, outliers in a data set can alert errors in the measurements taken. Using RStudio we created a function that calculates minor and major outliers, which results can be observed in *Table 2*. In *Figure 9* and *Figure 10* are reported the code of the function used to compute quartiles and inner fences. Now we can confirm that in 2012 dataset there are more outliers than in 1999 as you can see in *Table 2*.

About outliers and extreme values, we can say that, if we do not consider them as measurements errors, they could be related to specific events. For example, in 2012 dataset the maximum value is 909 $\mu g/m^3$and it may be due to positioning the monitor in a very polluted area, like near a landfill or a

factory. Or maybe it could be caused by an environmental event like a fire.

A few natural sources of PM2.5 are:

- fires
- vulcanic eruptions
- rock erosions

While anthropogenic sources are:

- combustion engines
- home heating due to fuel or coal
- incineretors and power plants
- tobacco smoke

So it may be better to make deeper analysis in order to understand if these outliers concern, for example, a single monitor located in a polluted area. In this case the extreme values are important because they suggest a polluted area, in the other case, if these values are evenly distributed in different locations we can consider them as measurements errors.

	First Quartile $\mu g/m^3$	Third Quartile $\mu g/m^3$	Percentage Minor outliners	Percentage Maximum outliners
1999 sample values	7,2	17,9	3,07%	0,54%
2012 sample values	4	12	3,62%	0,74%

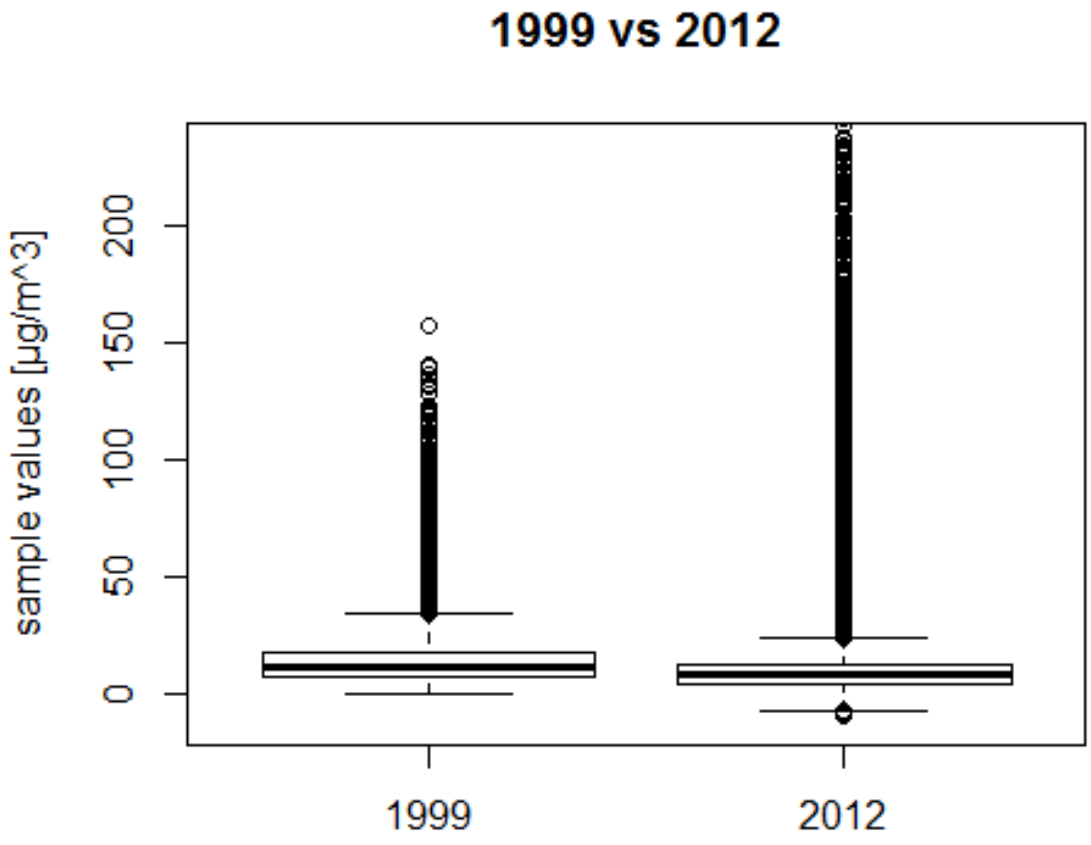
Table 2: With the help of a simple function, we calculated first and third quartile of our distributions, then the inner fences both for major and minor outliers. We obtained 47.160 minor outliers and 9.739 major outliers for 2012 dataset and we divided these *numbers for the total of samples we have in pm1 (1.304.287) in order to calculate these percentages. For 1999 dataset we had 3.605 minor outliers and 634 major outliers above 117.421 total samples contained in pm0.*

```
# Calculate inner fences for a sample vector (both major and minor)
calcInnerFences = function(vector, pm, factor){
  q1 = quantile(vector, na.rm = T)[2]
  q3 = quantile(vector, na.rm = T)[4]
  interquantileRange = (q3 - q1) * factor # factor can be 1.5 or 3
  highBound = interquantileRange + q3
  lowBound = q1 - interquantileRange
  pmOut = subset(pm, Sample.Value < lowBound | Sample.Value > highBound)
  return(pmOut)
}
```

Figure 9: *A function created to calculate quartiles, inner fences and outliers. If the factor is 1.5, we obtain minor outliers, while if it's 3 we obtain major outliers.*

```r
# Calculating inner fences for x1 with the help of a function:
pmOut1_min = calcInnerFences(x1, pm1, 1.5)
dim(pmOut1_min) # there are 47160 observations, the 3.62% of 2012 dataset (pm1)
# A point that falls outside the data set's inner fences is classified as a minor outlier,
# while one that falls outside the outer fences is classified as a major outlier.
pmOut1_maj = calcInnerFences(x1, pm1, 3)
dim(pmOut1_maj) # there are 9739 observations which are major outliers (0.74%)

# Calculating inner fences for x0 with the help of a function:
pmOut0_min = calcInnerFences(x0, pm0, 1.5)
dim(pmOut0_min) # there are 3605 observations, the 3.07% of 1999 dataset
pmOut0_maj = calcInnerFences(x0, pm0, 3)
dim(pmOut0_maj) # there are 634 observations which are major outliers (0.54%)
```

***Figure 10**: The code where there is a calling to calcInnerFences function: the return is a dataset made only of outliers.*

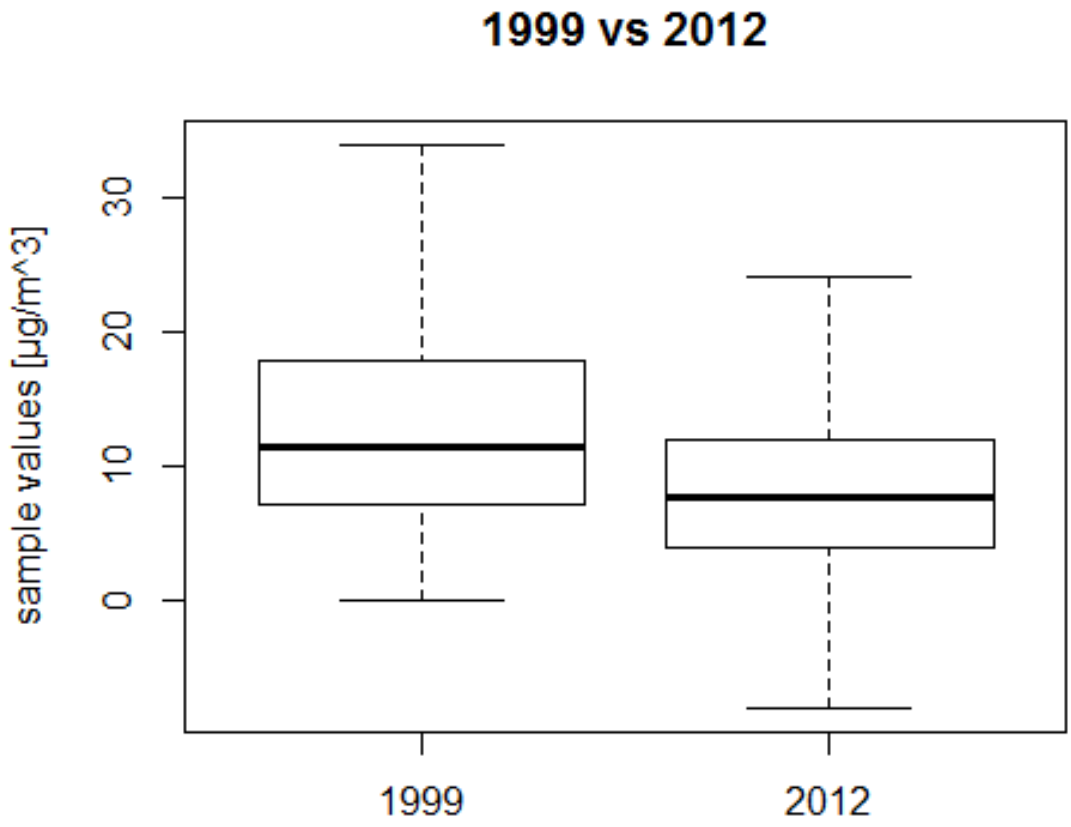


***Plot 5:** 1999 (left) and 2012 (right) dataset boxplot with outliers*

```r
# Calculating inner fences for x1 with the help of a function:
pmOut1_min = calcInnerFences(x1, pm1, 1.5)
dim(pmOut1_min) # there are 47160 observations, the 3.62% of 2012 dataset (pm1)
# A point that falls outside the data set's inner fences is classified as a minor out
# while one that falls outside the outer fences is classified as a major outlier.
pmOut1_maj = calcInnerFences(x1, pm1, 3)
dim(pmOut1_maj) # there are 9739 observations which are major outliers (0.74%)

# Calculating inner fences for x0 with the help of a function:
pmOut0_min = calcInnerFences(x0, pm0, 1.5)
dim(pmOut0_min) # there are 3605 observations, the 3.07% of 1999 dataset
pmOut0_maj = calcInnerFences(x0, pm0, 3)
dim(pmOut0_maj) # there are 634 observations which are major outliers (0.54%)
```

Figure 10: *The code where there is a calling to calcInnerFences function: the return is a dataset made only of outliers.*

A good way to analyse a statistic distribution and its outliers is to make its boxplot. That is what we did in *Plot 5*, where we have the 1999 data boxplot and the 2012 data boxplot, while in *Plot 6* we have the same boxplot but without outliers.



Plot 6: *1999 (left) and 2012 (right) dataset boxplot without outliers.*

By observing *Plot 5* and *Plot 6*, it's clear that these data have right-skew, which entails that the mean is on the right of the peak value and we have asymmetrical distributions. So besides median and mean value, we have to consider also the mode in order to have three metrics to rely on; we reported them in *Table 3*.

	Mean Value $\mu g/m^3$	Median $\mu g/m^3$	Mode $\mu g/m^3$
1999 dataset	13,74	11,50	7
2012 dataset	9,14	7,63	6

Table 3: *In any symmetrical distribution the mean, median and mode are equal, while in asymmetrical distributions things are different and it's better having three different metrics in order to analyse our data, which are reported in this table.*

In *Figure 11* is reported the code used in order to calculate the mode value of both distributions.

```
# The mode is the value that appears most often in a set of data and that's what this
# function returns
Mode <- function(x) {
  xna = x[!is.na(x)] # cleans vector from NA
  ux <- unique(xna)
  ux[which.max(tabulate(match(x, ux)))]
}
```

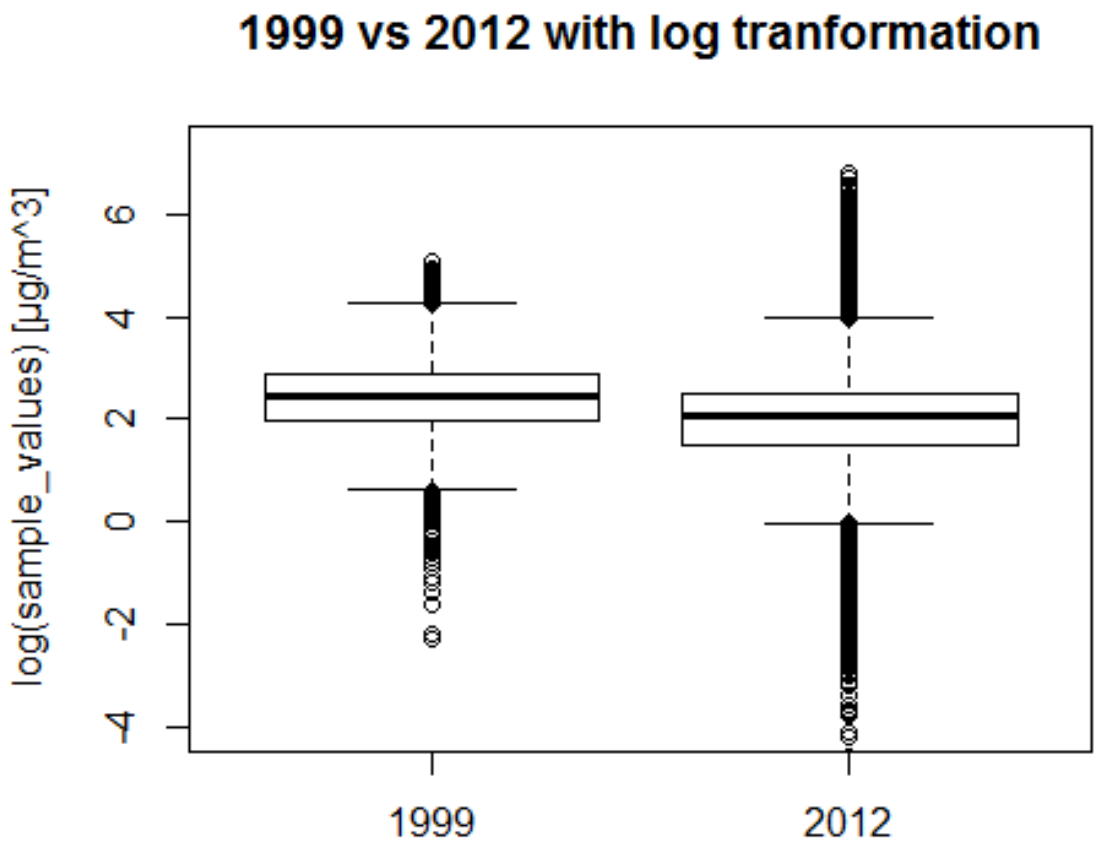
Figure 11: A function which returns a mode value from an array which represents a statistic distribution.

```
boxplot(log(x0, base = exp(1)), horizontal = T, main = title1999, xlab = logsample)
boxplot(log(x1, base = exp(1)), horizontal = T, main = title2012, xlab = logsample)

boxplot(log(x0, base = exp(1)), outline = F, horizontal = T, main = title1999,
        xlab = logsample)
boxplot(log(x1, base = exp(1)), outline = F, horizontal = T, main = title2012,
        xlab = logsample)
```

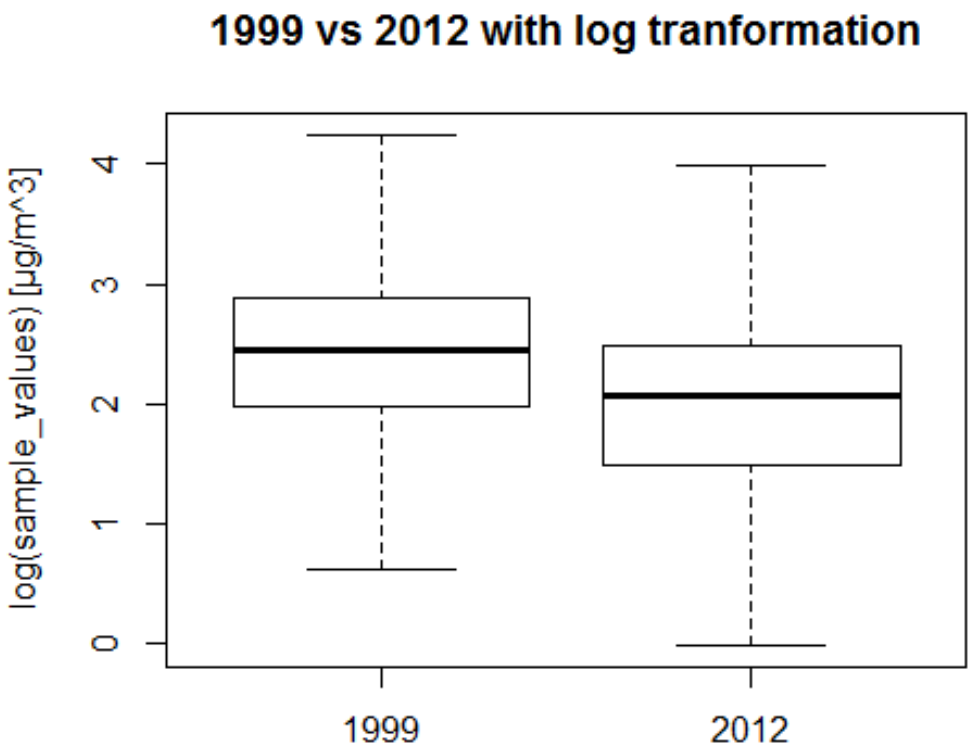
Figure 12: *The code where we transformed the 1999 and 2012 distributions using logarithm.*

We have applied a logarithmic transformation to the distributions, in order to even out the boxplot of *Plot 5* and we obtained a better result in *Plot 7* and *Plot 8*. In these last plots, we can see that here is easier to compare the two datasets than what we saw in *Plot 3* because the boxplot is a method to

represent a statistical distribution. The inner line of the box represents the median of the distribution; the extreme lines of the box express the first and third quartile. In this way, in every quartile we have 25% of population overshadowing the quantity of the data. In this way we can compare the two graphs with greater ease and we can observe that the 1999 boxplot is a bit shifted to the right, consequently every quartile, median included. For this reason, we can say that we record a slight increase of the PM2.5 mass.



Plot 7: *A logarithmic transformation of 1999 (left) and 2012 (right) dataset with outliers.*



Plot 8: *A logarithmic transformation of 1999 (left) and 2012 (right) dataset without outliers.*

In the summary of 2012 we noticed that the minimum value was -10 $\mu g/m^3$and this it's quite a peculiar thing, because PM2.5 measure concernes particle mass. That's why in the next step we will analyse this aspect.

Why are there negative values?

Theoretically, we can't have negative values because to measure PM2.5 you have to install a filter in the environment (there are a lot of filter: a porous membrane air filter, a fibrous air filter, a transparent air filter, and many others… each with specific characteristics), and the particles in the air are blocked by it. The particles have a positive mass, and it's impossible to have a negative value in the measurement. You can see an example in *Figure 13*.

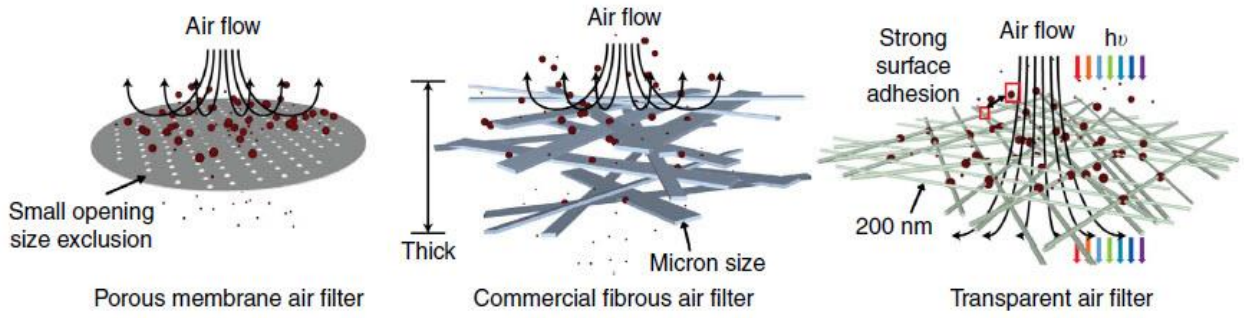


Figure 13: *Three different air filter examples.*

So if there's a negative value for the PM 2.5 variable that's a little wacky, that's a quite anomalous.

With the command that you can see in *Figure 14* we found how many negative values there are:

```
pm1Neg = pm1[pm1$Sample.Value < 0 & !is.na(pm1$Sample.Value), ]
dim(pm1Neg) # There are 26474 in the 2012 data set over 1304287 totally observations; so the proporti
( dim(pm1Neg)[1] / dim(pm1)[1] ) * 100 # about 2.02%
```

Figure 14: *A part of the code, that we used in order to obtain new datasets only negative values*

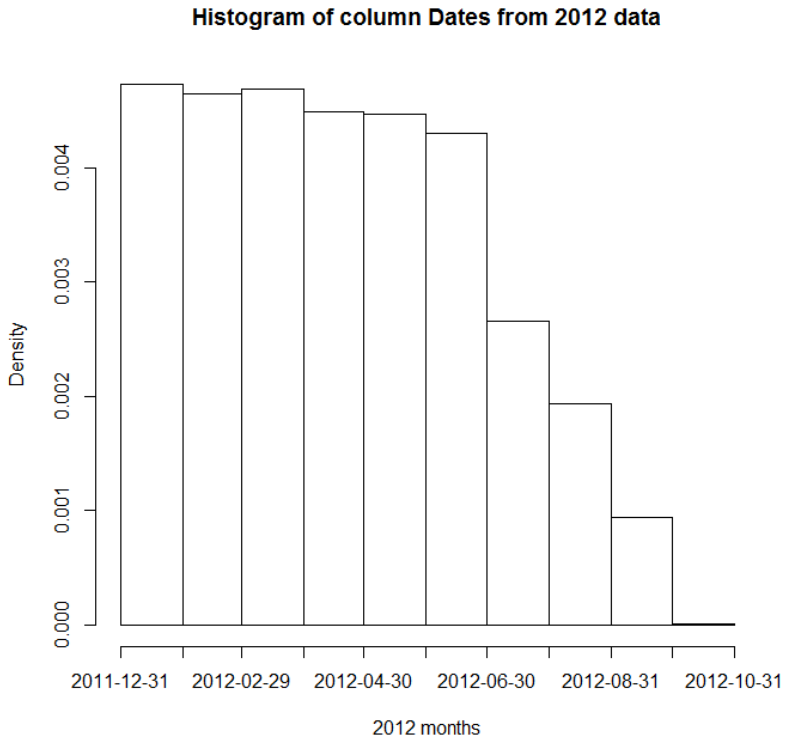
The result we obtained is in the next table (*Table 4*):

	1999	2012
Records	117.421	1.304.287
Missing Value	13.217	73.133
Percentage of Missing Value	11,26%	5,61%
Negative Value	0	26.474
Percentage of Negative Value	0,0%	2,02%

Table 4: *In this table are reported a few operations we made on missing and negative values.*

In 1999 there aren't negative values, so we analysed only the 2012 file.

It may be important to observe if there are negative values at specific times of the year, or in certain places or times. So we plotted a histogram of the dates by month to see where the collection occurs. The graph we obtained is in *Plot 9*:



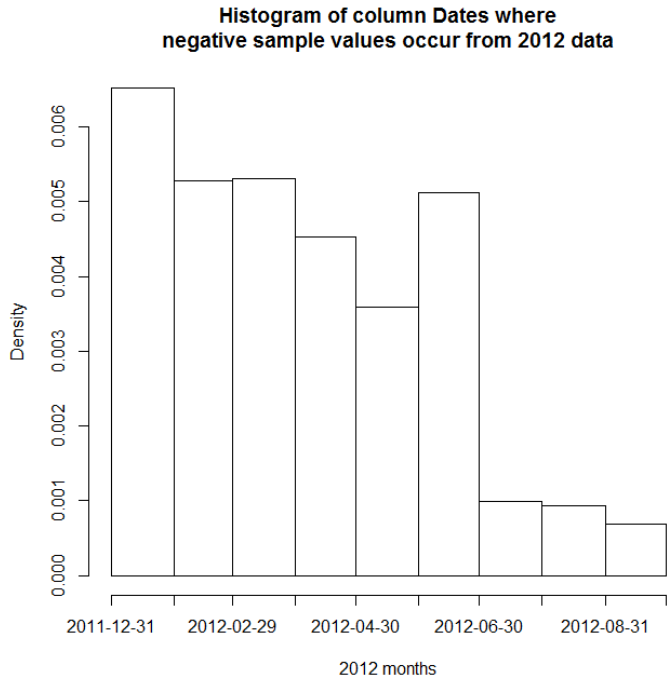
Plot 9: *An Histogram of 2012 dataset where you can see when the measurements have been take through the months.*

```r
# Corrects format dates
correctDate = function(vector){
  date = as.Date(as.character(vector), "%Y%m%d")
  return(date)
}
```

Figure 15: *The code used in order to correct the date format.*

We noticed that a lot of observations has been taken in the period between 2011-12-31 / 2012-06-30 (January/ June), less between 2012-06-30 / 2012-09-30 (July / September), and almost none from 2012-09-30 / 2012-12-31 (October to December).

And in *Plot 10* there is the histogram of the dates where the negative values occur:



Plot 10: *An Histogram of 2012 dataset where you can see the amount of negative values of PM2.5 reported through the months.*

We observed that during winter and spring, between January and June, the most of negative sample values occur; while during summer, between July and September, the frequency of negative sample values measured is lower; finally, from October to December there aren't.

We noticed, therefore, that PM2.5 is very low in winter while high in the summer. Also, when the pollution values are high, they are easier to measure; while when they are low are more difficult to measure.

In conclusion we can say that some of the negative values are just a measurement error when there are very low values, but we removed them with this part of code, in order to understand changes:

```
# The positive value without the missing values are:
pm1Pos = pm1[pm1$Sample.Value >= 0 & !is.na(pm1$Sample.Value), ]
```

Figure 16: *A part of the code, that we used in order to obtain new datasets without negative and missing values (only positive values).*

we obtain these values:

	1999	2012
Records	117.421	1.304.287
Missing Value	13.217	73.133
Positive Value	104.204	1.204.680
Percentage Correct Value	88,74%	92,37%
Mean Value with positive value $(\mu g/m^3)$	13,74	9,38
Median with positive value $(\mu g/m^3)$	11,50	7,90
Standard Deviation with positive value $(\mu g/m^3)$	9,41	8,48

Table 5: *In this table are reported a few operations we made on missing and negative values.*

We can observe that the 1999 data remained the same because in 1999 there aren't negative values. In 2012, removing the negative values, we saw a little change, in fact the mean changed from 9,14 $\mu g/m^3$ in 9,38 $\mu g/m^3$, the median from 7,63 $\mu g/m^3$ in 7,90 $\mu g/m^3$ and consequently the standard deviation from 8,56 $\mu g/m^3$ in 8,48 $\mu g/m^3$.

The consequences of this removal may change some graphics, although not significantly, because of the little difference of the previous values (mean, median and standard deviation) listed above. It might be useful to analyze the difference with and without negative values at the state-level, as we will see in step 5 (Exploring change at the state level).

Exploring change at one monitor

After studying the general air pollution levels trend between 1999 and 2012, we focused on one location and one of its monitor. Particularly, it was requested to consider New York state (State Code = 36). At this point, we had to find a monitor that is in both periods and that has a lot of observations.

```
countySite0 = paste(pm0NY$County.Code, pm0NY$Site.ID)
countySite1 = paste(pm1NY$County.Code, pm1NY$Site.ID)
pm0NY <- cbind(pm0NY, countySite0)
pm1NY <- cbind(pm1NY, countySite1)
intersection <- intersect(countySite0, countySite1)


library('plyr') # library for "count"
freq0 <- count(pm0NY, 'pm0NY$countySite0')
names(freq0)[1] <- 'countySite'
freq1 <- count(pm1NY, 'pm1NY$countySite1')
names(freq1)[1] <- 'countySite'


index0 <- c()
index1 <- c()
for (i in 1:length(intersection)) {
  ind0 <- match(intersection[i], freq0$countySite)
  index0 <- append(index0, ind0)

  ind1 <- match(intersection[i], freq1$countySite)
  index1 <- append(index1, ind1)
}

freq00 <- freq0[index0,]
freq11 <- freq1[index1,]
```

Figure 17: *Using "plyr" library we found County ID and Monitor ID combinations in both 1999 and 2012 dataset (relative to New York state). We obtained the total observations for these combinations.*

Tab 0

countySite	freq
1 5	122
1 12	61
5 80	61
13 11	61
29 5	61
31 3	183
63 2008	122
67 1015	122
85 55	7
101 3	152

Tab 1

countySite	freq
1 5	64
1 12	31
5 80	31
13 11	31
29 5	33
31 3	15
63 2008	30
67 1015	31
85 55	31
101 3	31

Figure 18: *these two tabs show the frequency of the County.Code-Site.ID combinations in 1999 dataset (Tab 0) and in 2012 dataset (Tab1).*

Thanks to this code, for the New York State Code, we found 10 County Code-Site ID combinations in 1999 data that match in 2012's. Then, we extracted from the data frames the rows (index) which contained the couples of Site ID - County Code founded in both 1999 and 2012, as reported in *Figure 17*. After this, we counted the amount of observations for every combination of Site ID - County Code in 1999 dataset and in 2012 one: *Figure 18* reports this.

As suggested by the text, chose County_Code 63 and Site_ID 2008 and we obtained two new data frames.

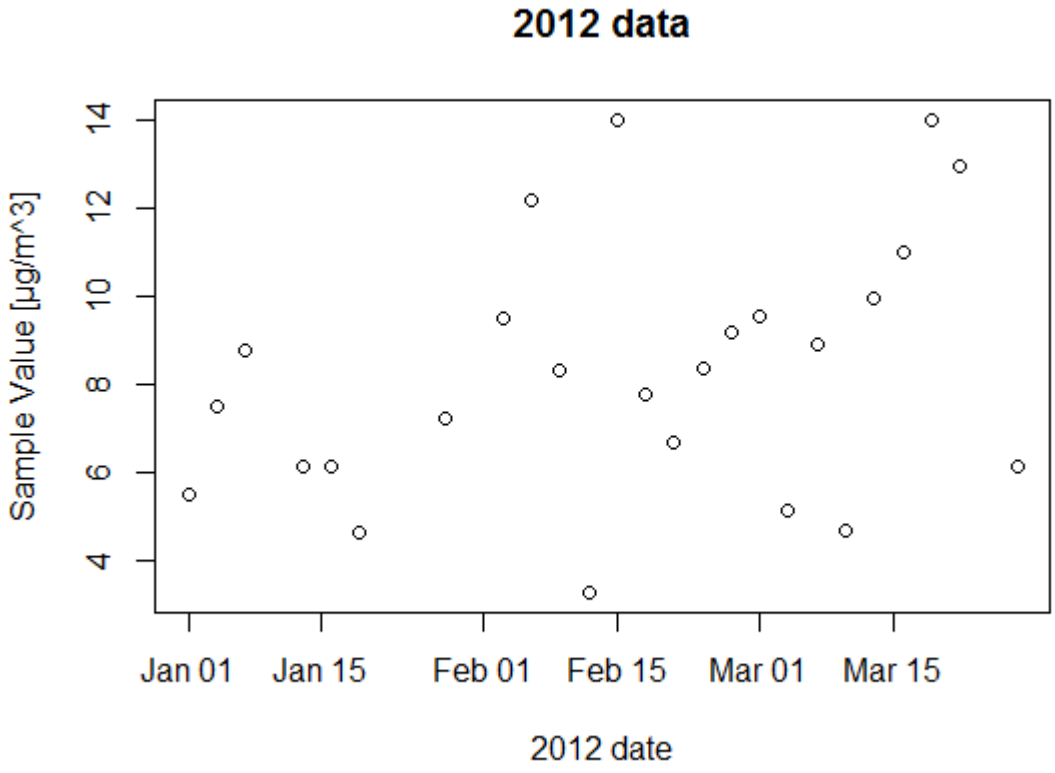
```
# Call a new data-frame pm1sub, as a subset of pm1 with Country code 63 and site ID 2008
# (always related to New York state).
pm1sub = subset(pm1NY, County_Code == 63 & Site_ID == 2008)

# Then do the same thing for the 1999 data, and call it pm0sub.
pm0sub = subset(pm0NY, County_Code == 63 & Site_ID == 2008)
```

Figure 19: *Split in separate data frames by each monitor.*

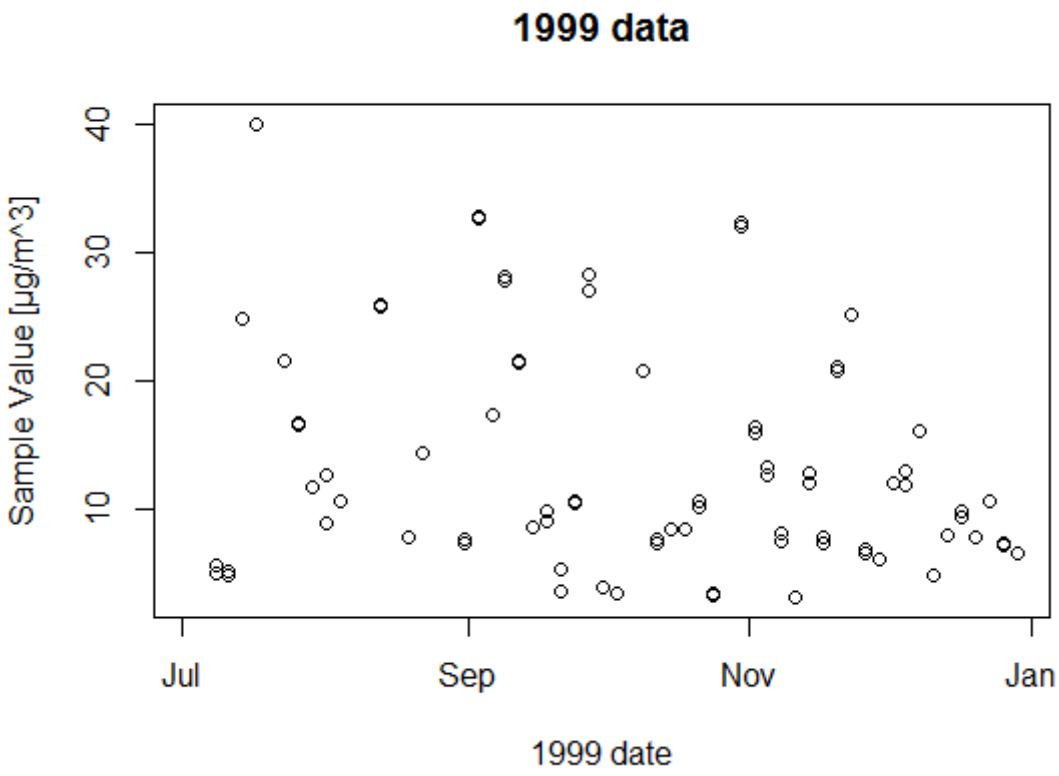
These new two data frames have, respectively, 122 and 30 rows, that means that in 2012 have been collected less observations than in 1999.

Thanks to the *Plot 13* below, it's easy to notice that the sample values in the 2012 dataset are distributed in non-uniform manner for all the space, somewhere between 4 and 14 micrograms per meter cubed.

For 1999 the data are only actually recorded starting in July, through the end of the year. In the plot below, you can see that the values are placed in a range between approximately 5 micrograms per meter cubed and 40 (*Plot 12)*

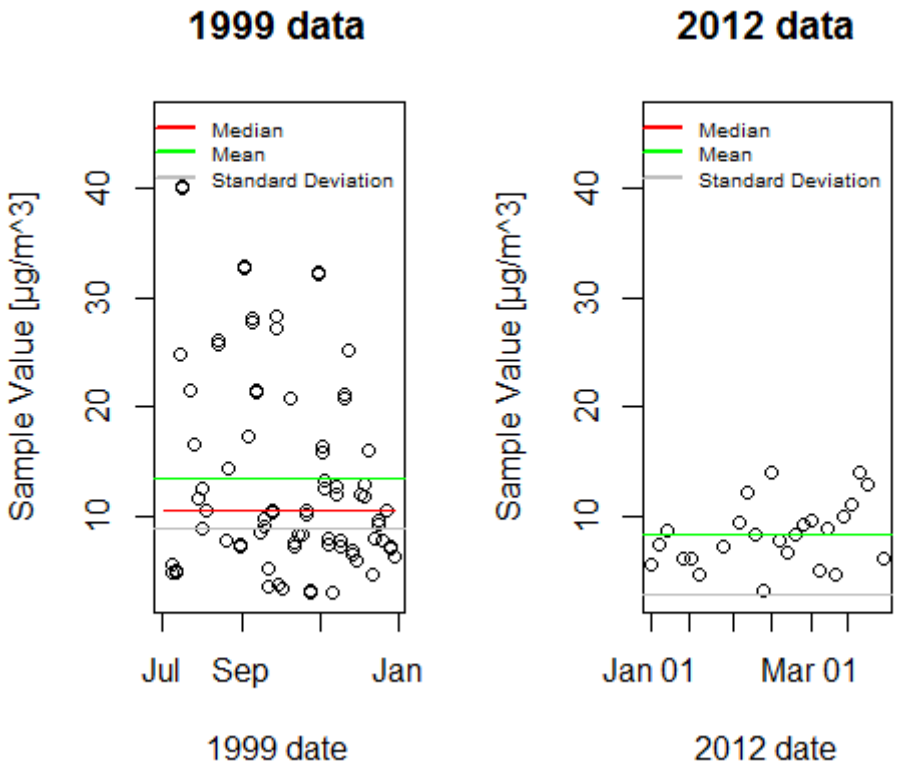


Plot 11: *Data in 2012 which are distributed in non-uniform manner for all the space.*



Plot 12: *Data in 1999 which started from July and are distributed in non-uniform manner for all the space.*

The better to see the trend of PM2.5 values, we put both 1999 and 2012 on the same panel, putting them on the same range.



Plot 13: *1999 Data and 2012 data put on the same panel, it's better to see the difference. The line red represents the median, the line green the mean and the grey line represents the standard deviation.*

As you can see from *Plot 13*, the median decreased in 2012 at this monitor and it is interesting to note that the 1999 values are shed in the plot, while the 2012 ones are centred around the mean and the median that are overlapped. This means that in 1999 there was a greater dispersion of the samples than in 2012.

On average, in 2012 there has been an improvement and, compared to 1999, there were no high values, while there was a "stabilization" of the same values, as we can easily see from the standard deviation decrease.

Exploring change at state level

In this part of code, we analysed the air pollution at country level, divided state by state. First of all, we removed the NAN value (missing value) and we obtained some results that you can see in *Table 6*:

	1999	2012
Records	117.421	1.304.287
Missing Value	13.217	73.133
Correct Value	104.204	1.231.154
Percentage of Correct Value	88,74%	94,39%

Table 6: *In this table are reported a few operations we made on missing values in both 1999 and 2012 dataset.*

After we wrote the code that you can observe in *Figure 20* (the same for 1999 and 2012) to obtain a matrix with 3 columns (State Code, Sum of Sample Value of the same state, counter):

```
pm0NA = pm0[!is.na(pm0$Sample.Value),] # remove the NA values
stateCodeCol = pm0NA$State.Code # we take the State Code Column
firstVal = stateCodeCol[1]
valuePMCol = pm0NA$Sample.Value # and the Sample Value column
cont = 1
list = 1

# We create a matrix with 3 columns: StateCode, Tot PM, and cont
# We do this to analyze better the results
a0 = matrix(nrow = length(stateCodeCol),
            ncol = 3,              # number of columns
            byrow = TRUE)          # fill matrix by rows

# We initialize the first row of the matrix
a0[1, 1] = firstVal
a0[1, 2] = valuePMCol[1]
a0[1, 3] = cont

# Now we can fill the matrix
for (i in 2 : length(stateCodeCol)){

  if(stateCodeCol[i] == stateCodeCol[i - 1]){
    a0[list, 2] = a0[list, 2] + valuePMCol[i]
    a0[list, 3] = a0[list, 3] + 1
  }
  else{
    list = list + 1
    cont = 1
    a0[list, 1] = stateCodeCol[i]
    a0[list, 2] = valuePMCol[i]
    a0[list, 3] = cont
  }
}
b0 = head(a0, list)
```

Figure 20: *The code we used to make some manipulation on both the datasets in order to obtain a matrix without missing values with three columns: State Code, Sum of sample values for this state, Counter of how many samples we have in this state.*

We did the same thing for the 2012 file. This process was helpful to us to analyse the dates more effectively.

We divided the second column (the sum of PM2.5 of each state) for the third column (the counter) and we obtained the mean value of PM2.5 for each state. We did this for the two years. After we united the two matrix and obtained what you can see in *Figure 21*:

	stateCode	mean.x	mean.y				
1	1	19.956391	10.126190	28	31	9.167770	9.2074
2	2	6.665929	4.750389	29	32	9.235101	7.0099
3	4	10.795547	8.609956	30	33	11.836578	6.2036
4	5	15.676067	10.563636	31	34	13.866028	8.3505
5	6	17.655412	9.277373	32	35	6.511285	8.0897
6	8	7.533304	4.117144	33	36	12.453519	8.6382
7	9	13.276085	7.561940	34	37	15.760104	9.5682
8	10	14.492895	11.236059	35	38	7.988184	6.6544
9	11	15.786507	11.991697	36	39	17.578823	11.7719
10	12	11.137139	8.239690	37	40	10.657617	10.8498
11	13	19.943240	11.321364	38	41	9.148453	7.3736
12	15	4.861821	8.749336	39	42	14.370468	10.8229
13	16	9.321034	8.913206	40	44	11.300660	6.6052
14	17	16.724550	10.761918	41	45	14.685859	9.2251
15	18	15.639703	11.486988	42	46	9.586732	6.0998
16	19	11.523740	9.519467	43	47	17.129217	11.7158
17	20	12.391900	8.144006	44	48	12.157830	11.4562
18	21	16.039906	11.210751	45	49	9.356495	6.3746
19	22	14.293732	11.520043	46	50	9.711592	7.4071
20	23	9.519416	7.208252	47	51	14.332742	8.7080
21	24	15.985083	10.428976	48	53	9.966754	6.3646
22	25	12.314527	8.636699	49	54	16.769652	9.8212
23	26	13.819438	8.758203	50	55	12.682977	7.9145
24	27	10.090739	7.533809	51	56	7.207296	4.0055
25	28	16.297349	11.019940	52	72	9.095173	6.0480
26	29	13.555692	10.215110	53	78	10.460000	
27	30	9.199872	7.297635				

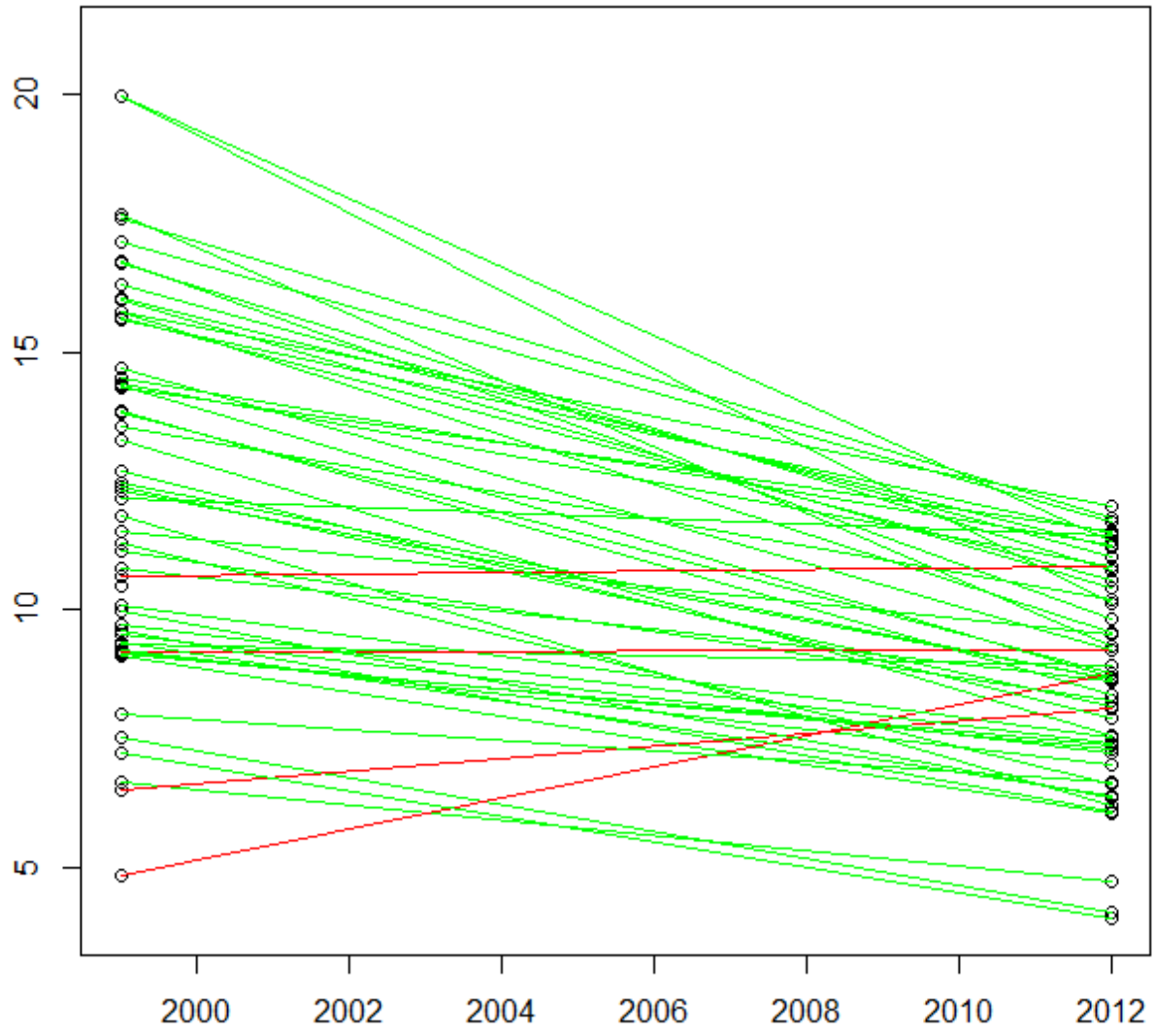
Figure 21: *We obtained a matrix with three columns: the first reports the id of very state (State Code), the second reports the mean value for every State Code found in 1999 dataset, the third reports the mean value for every State Code found in 2012 dataset.*

In the first column you can see the State Code, in the second the mean of the Sample Value in 1999 and in the third column the mean of the Sample Value in 2012. The state with the State Code 78 in the 2012 hasn't revelations.

To display in a more understandable way the data we obtained, we decided to plot them with the graph of *Plot 14*. On the vertical axis are placed the mean value of PM2.5 of each state in 1999 (at the left), and each of them is connected with the corresponding mean value of the same state in 2012. On the horizontal axis are placed the years. We decided to colour the lines of the graph with green the values that decreased from 1999 to 2012 and with red the values that increased from 1999 to 2012. After we regulated the axis because some lines went out from the graph. What we obtained was this:

```
# Plot the first line of points relative at the 1999 data
plot(line1999, cc[, 2], xlim = c(1999, 2012), ylim = c(4, 21))
# Plot on the same graph the second line of points relative at the 2012 data
points(line2012, cc[, 3])
```

Figure 22: *With this code we corrected the axis in Plot 14.*

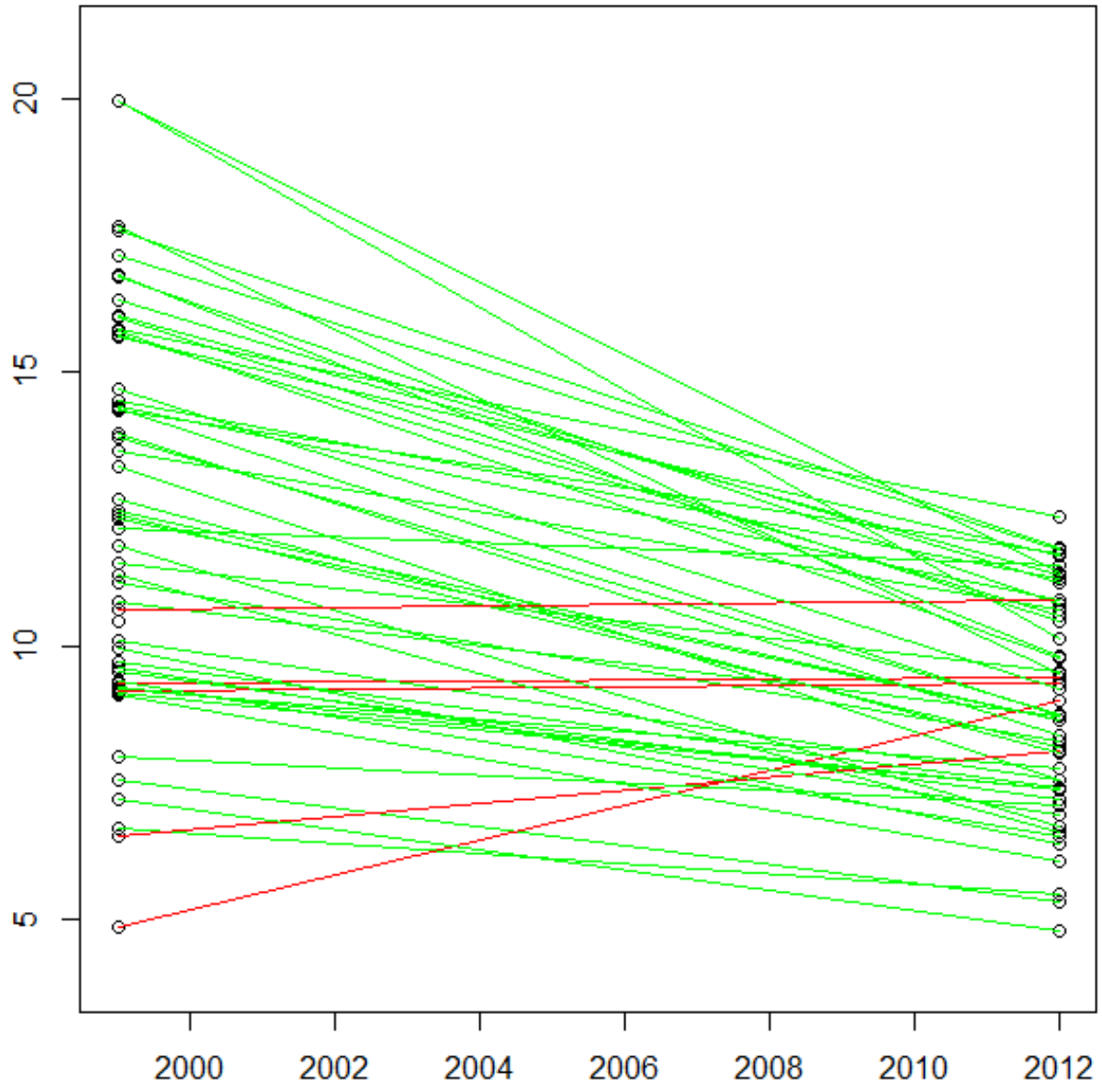


Plot 14: *Vertical axis expresses mean values for every state, while horizontal axis expresses different years: of course we only have data from 1999 (to the left) and 2012 (to the right). In green we pointed out those states which mean value decreased in time, while in red those state which mean value increased.*

Only four lines were red, it means that only four states have increased their air pollution from 1999 to 2012. Obviously, all other states have decreased their level of air pollution. We were curious to see which were the states that have increased their level of air pollution, and obtained that they were the state

with this State Code: 15, 31, 35, 40 that matched with Hawaii, Nebraska, New Mexico and Oklahoma. States that didn't pollute much were Alaska and Colorado.

If we removed the negative values (as said in step 3 – Why are there negative values?) we obtained a very similar graph:



Plot 15: *Vertical axis expresses mean values for every state, while horizontal axis expresses different years: of course we only have data from 1999 (to the left) and 2012 (to the right). In green we pointed out those states which mean value decreased in time, while in red those state which mean value increased. The difference with the previously graph is that in this case there aren't negative values.*

There is only one state which is added at the list of states that increased their air pollution, its State Code is 16, which represents is Idaho.

Neural Network

In this section, we list the main steps we followed to train a neural network. The basic idea was to try to build a neural network, starting from a training set, capable of predicting the level of air pollution (the "Sample.Value").

To do this, we considered the data set without rows with "NA" under the "Sample.Value" column and selected 7 of the 28 attributes of our data set: "County.Code", "Site.ID", "POC", "Sample.Duration", "Method", "hour" and "DayNumber". Obviously, we selected the attributes that seemed more significant, also in relation to values that they assumed: in fact, we discarded the attributes without values or with few values.

Then, for the "Date" attribute we decide to convert it into "dayNumber" that trivially indicates the number of the days in the year, so it is between 1 and 366 (2012 is a leap year).

```
# function to transform a date in a day between 1 and 366
# @param dt: date in yyyymmdd format
fromDateToNumber <- function(dt){
  withoutYear <- dt %% 10000
  day <- withoutYear %% 100
  month <- floor(withoutYear / 100)
  numberOfDay <- 0
  for (i in 1:(month-1)) {
    numberOfDay <- numberOfDay + dayInMonths[i]
  }
  numberOfDay <- numberOfDay + day
  if(month == 1){
    return (day)
  }
  return(numberOfDay)
}
```

Figure 23: *the code of the function that convert a "yyyymmdd" date in a day between 1 and 366.*

For the "Start.Time" attribute, instead, we had to convert in integer values.

```
myTime <- factor(pm1days$Start.Time)
hour = format(as.POSIXct(myTime,format="%H:%M"),"%H")
hour <- as.integer(hour)
pm1days <- cbind(pm1days, hour)
```

Figure 24: *the code to convert the "Start.Time" values into integer values.*

Now, we could create the training and the test set, starting from the data set. We had many possibilities, but we decided to select the third week of every month as test set and the remaining as training set.

```
lista = split(pm1days, pm1days$monthCol1)
df <- lista[[1]]

dfTest <- sqldf(strwrap(sprintf("select * from df where days >= %d and days <= %d",
                               15, 21)))
dfTrain <- sqldf(strwrap(sprintf("select * from df where days >= %d and days <= %d",
                                 0, 14)))
dfTrain <- rbind(dfTrain,sqldf(strwrap(sprintf("select * from df where days >= %d
                                               and days <= %d", 22, 31))))

for(i in 2 : 9){
  dfTest = rbind(dfTest, selectTrainOrTest(lista[[i]], 15, 21, days))
  dfTrain = rbind(dfTrain, selectTrainOrTest(lista[[i]], 0, 14, days))
  dfTrain = rbind(dfTrain, selectTrainOrTest(lista[[i]], 22, 31, days))
}
```

Figure 25: *the code for the creation of the training set and the test set, according to our implementation choices.*

```
# function for selecting a subset of the dataframe on the basis of intervals of days
# param @day0, @dayf: indicate initial and final days (included) in the interval
selectTrainOrTest = function(df, day0, dayf,  nameCol){
  dfTrainOrTest <- c()
  res = sqldf(strwrap(sprintf("select * from df where '$nameCol' >= %d
                              and '$nameCol' <= %d", day0, dayf)))

  if(dim(res)[1] != 0){ # if the query result is not empty
    dfTrainOrTest <- res
  }
  return(dfTrainOrTest)
}
```

Figure 26: *the code of the function used to extract the training set and the test set from the data set.*

For building the neural network we chose to use the "neuralnet" R library, so we had to normalize the values and create our new train and test data frame.

```
dfTrainNorm <- apply(dfTrain[c(4,5,7,8,10,11,13,32,33)], 2, norm.fun)
dfTestNorm <- apply(dfTest[c(4,5,7,8,10,11,13,32,33)], 2, norm.fun)

dfTrainNorm <- data.frame(dfTrainNorm)
dfTestNorm <- data.frame(dfTestNorm)
```

Figure 27: *the code used to normalize the data useful to create the neural network and to create the data frames.*

```
# function for values normalization
norm.fun <- function(x){
  (x - min(x))/ (max(x)-min(x))
  }
```

Figure 28: *the code of the function for values normalization.*

Now we are able to train a neural network, fixing some parameters:

- "hidden": vector of integers specifying the number of hidden neurons (vertices) in each layer.
- "linear.output": specifies whether we want to do regression (TRUE) or classification (FALSE);
- "err.fct": is a differentiable function that is used for the calculation of the error. In our case, "sse" stands for "sum of squared error".

Tests and Results

Once we realized that train a neural network using the whole training set, we decided to choose a State and focused on it. We selected the State of Ohio (State Code = 39), because it has a low number of negative values (2.06%), low number of missing values (2%) and a large enough number of instances (30416).

We made several attempts with different values of the "hidden" parameter. In *Figure 29* we see the code for the neural network with only 2 hidden layer, for the other changes only the value of the "hidden" parameter. The same goes for the code in *Figure 30* for the prediction on the network built previously.

```
myNn <- neuralnet(Sample.Value ~ County.Code + Site.ID + POC
                  + Sample.Duration + Method  + hour + dayNumber,
                  data = dfTrainNorm, hidden = 2, linear.output = TRUE,
                  err.fct = "sse")
```

Figure 29: *an example of the code used to create our neural networks, using the "neuralnet" library.*

```
myPredict2 <- compute(myNn2, dfTestNorm[,c(1:5,7,8)])$net.result
myRmse2 <- sqrt(mean(abs(dfTestNorm[,7] - myPredict2)))
```

Figure 30: *an example of the code used to predict the "Sample.Value" values and calculate the root mean squared error (rmse), using the "neuralnet" library.*

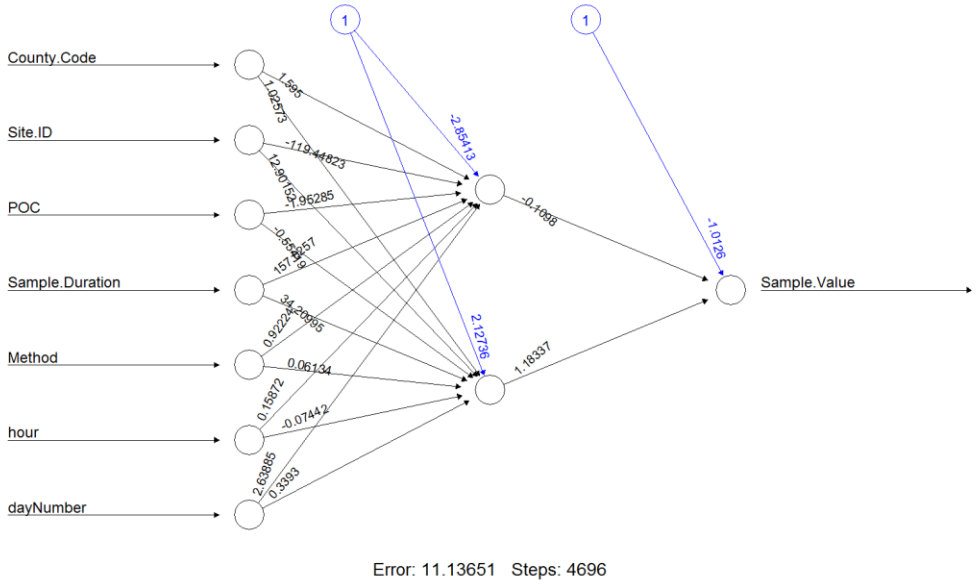


Figure 31: *neural network with "hidden = 2" train by Ohio data. The results are: "Error: 11.13651", "Reached threshold: 0.009287413", "Steps: 4696".*
The prediction generates a rmse = 0.1450606.

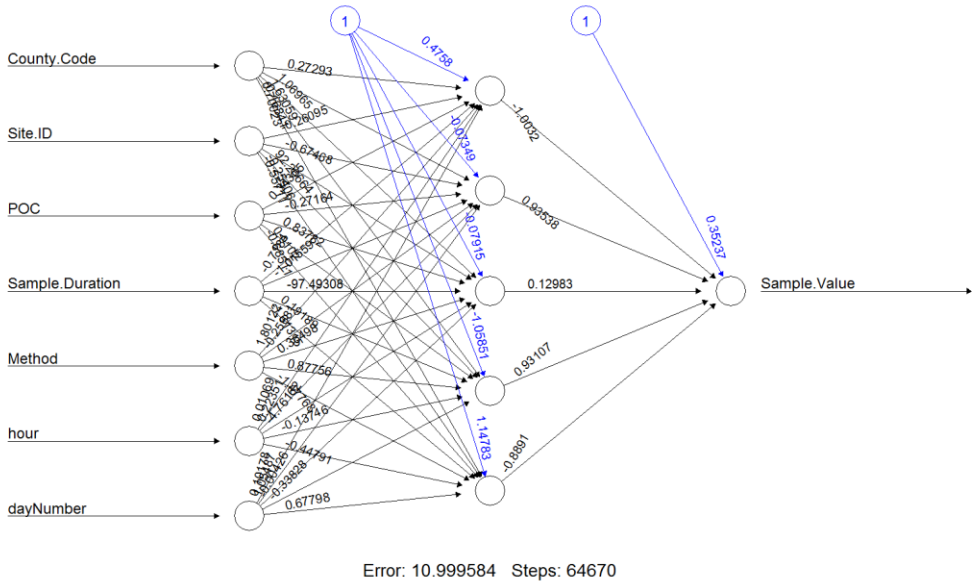


Figure 32: *neural network with "hidden = 5" train by Ohio data. The results are: "Error: 10.999584", "Reached threshold: 0.009941659", "Steps: 64670".*
The prediction generates a rmse = 0.1400914.

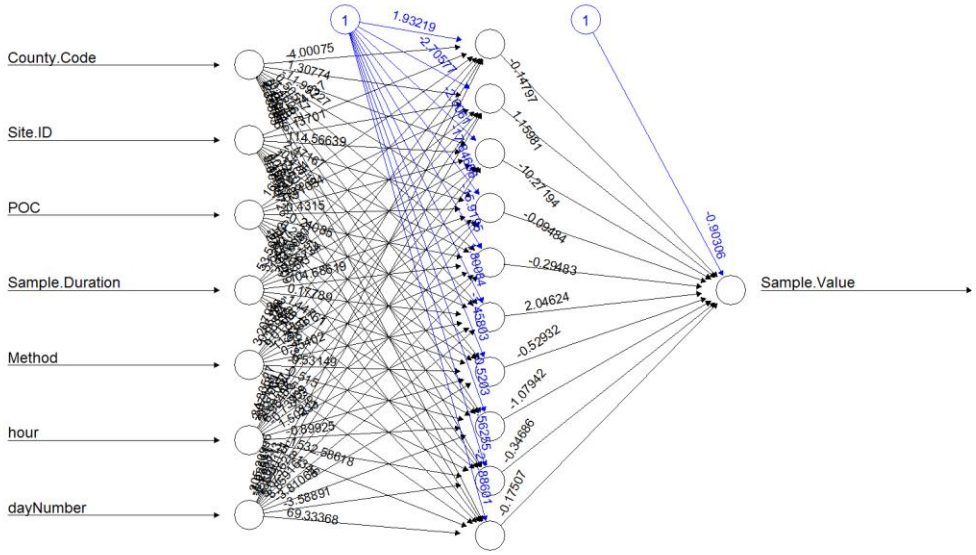


Figure 33: *neural network with "hidden = 10" train by Ohio data. The results are: "Error: 11.58595", "Reached threshold: 0.008706", "Steps: 62831".*
The prediction generates a rmse = 0.3458091.

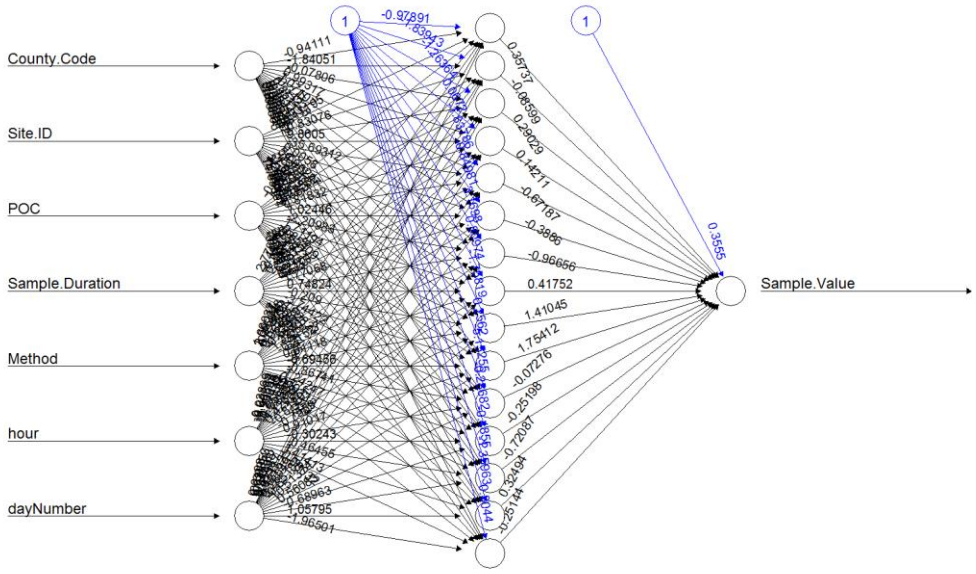


Figure 34*: neural network with "hidden = 15" train by Ohio data. The results are: "Error: 10.72262", "Reached threshold: 0.009586797", "Steps: 17393".*
The prediction generates a rmse = 0.1524939.

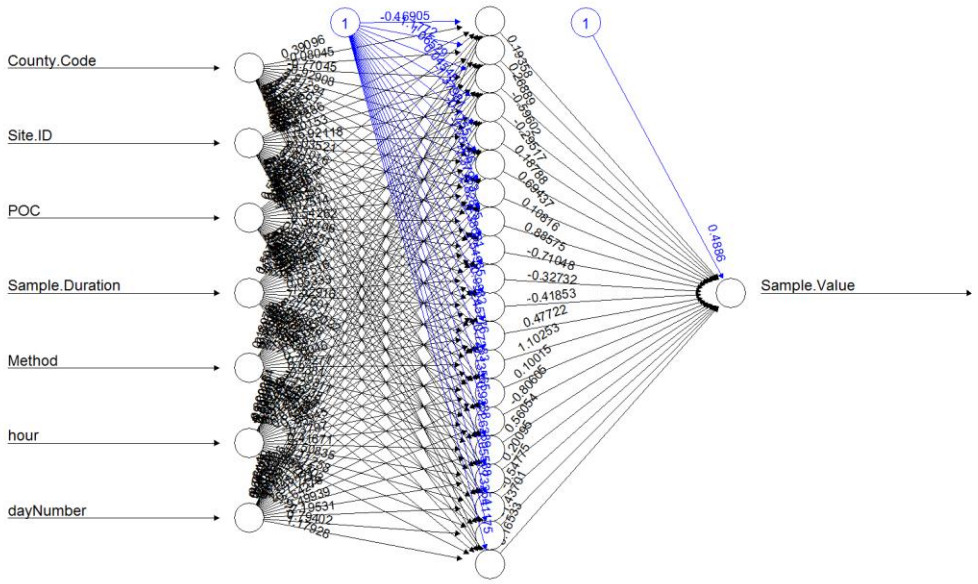


Figure 35: *neural network with "hidden = 20" train by Ohio data. The results are: "Error: 10.36132", "Reached threshold: 0.008240049", "Steps: 91378".*
The prediction generates a rmse = 0.1918753.

Future Developments

We tried to build a complete network, using almost two-thirds of the whole dataset as a training set, but requires too many time. The following are the results obtained, although they're coarse.

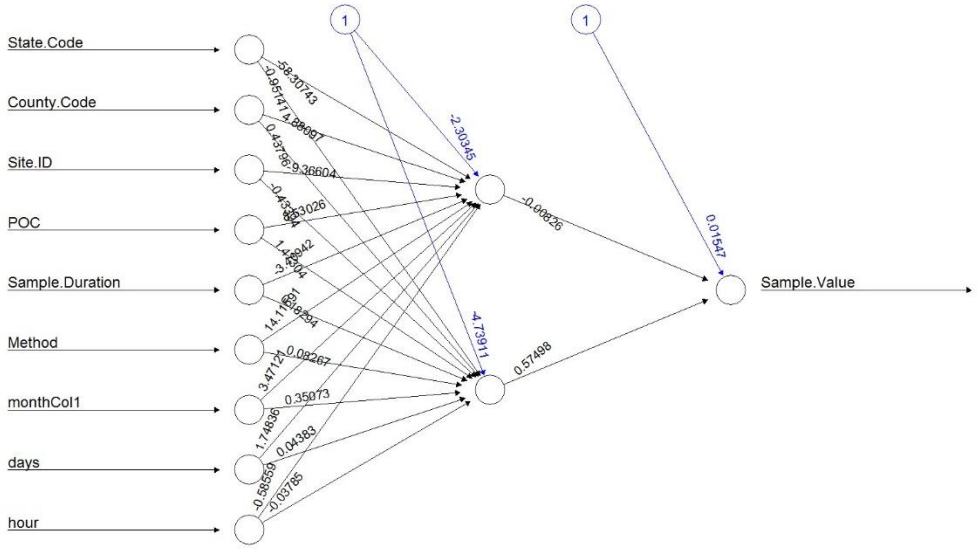


Figure 36: *neural network with "hidden = 2" train by whole data. The results are: "Error: 41.56744", "Reached threshold: 0.008536994", "Steps: 8646".*

The prediction generates a rmse = 0.08052485.

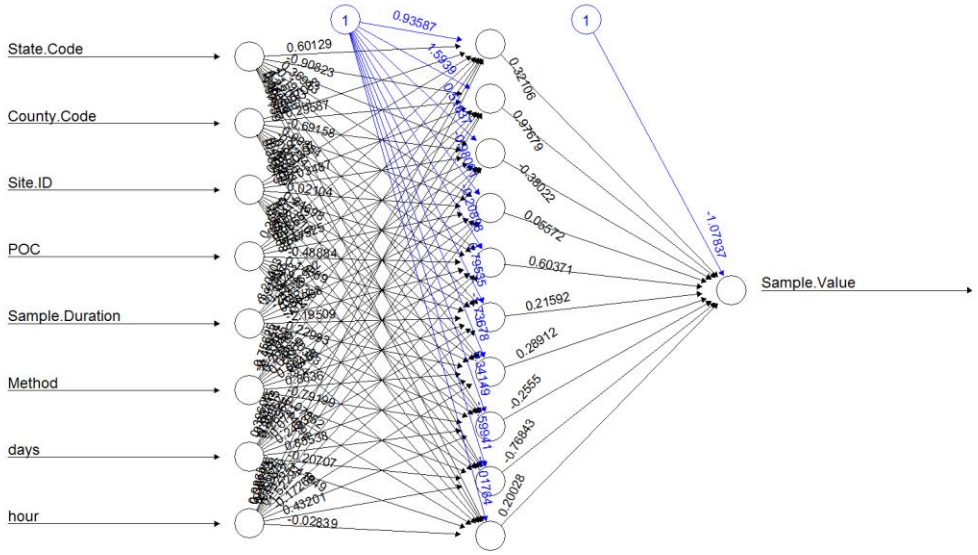


Figure 37: *neural network with "hidden = 10" and "threshold = 0.5" train by whole data. The results are: "Error: 42.01402", "Reached threshold: 0.4464969", "Steps: 5612".*
The prediction generates a rmse = 0.08011494.

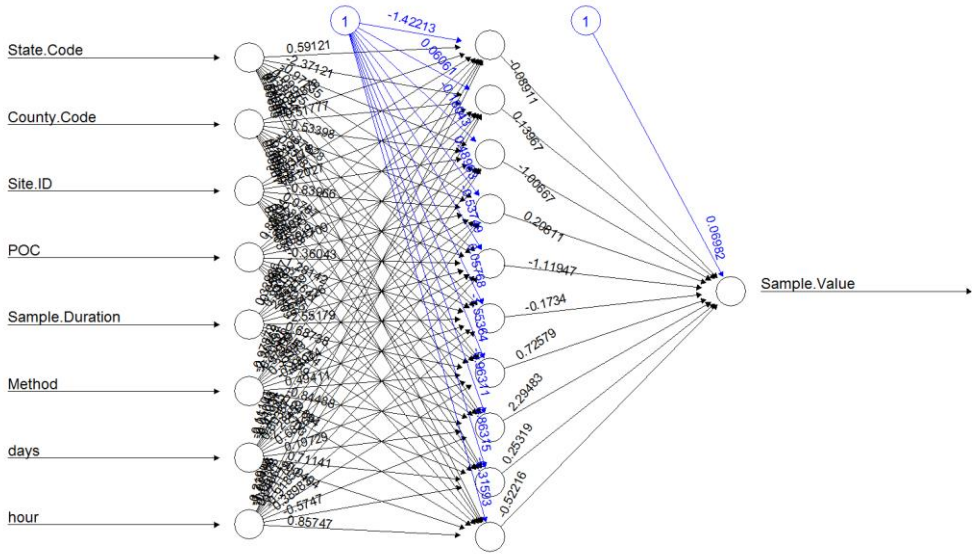


Figure 38: *neural network with "hidden = 10" and "threshold = 0.25" train by whole data. The results are: "Error: 41.66167", "Reached threshold: 0.2478608", "Steps: 26891".*
The prediction generates a rmse = 0.07985659.

Although the study was not sufficiently detailed, in the complete neural networks you immediately notice how the rmse is decreased, because the prediction became more precise.

For the future, it would be interesting try to obtain a more complex neural network, using more attributes and the whole dataset. Obviously, this work would require further data manipulations and more resources.

Conclusions

In the summary, we wanted to understand if particular matter level decreased between 1999 and 2012: we found that there was a general improvement in the country for what concerns the reduction of pollution level, as expected from the Clean Air Act.

Our conclusions derive not only from the study of the general case, but also from the study of a more specific one: we divided the data set by single state and single monitor.

For the future it could be interesting to make a deeper analysis to understand if the extreme values are measurement errors or samples due to specific situations. In this case it could be curious to classify different locations based on their pollution levels.