Esercizio 1 - Analisi dei furti di automobili a Chicago

Uberto Vittorio Favero[1,2] and Ludovico Loreti[1,3]

[1]UniversitÃă di Bologna
[2]Statistica Numerica

10 maggio 2017

1 Introduzione

In questa relazione esamineremo il file reso disponibile dal portale della città di Chicago dal 2001 inerente i furti di autoveicoli così da descrivere il campione di dati, analizzarlo formandone un modello.

Il datasource è liberamente scaricabile a: https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2

Ed è composto da:

- Data e ora di rilievo del reato
- Indirizzo
- Tipologia di reato
- Descrizione del tipo di veicolo
- Descrizione del luogo del furto (Strada, parcheggio privato, supermercato etc)
- Ward, ovvero il distretto di polizia di riferimento
- Community Area, ovvero il quartiere
- Anno
- Latitudine e longutudine

`kruskal.test` help page into a LaTeX document:

2 Preparazione dei dati

Il dati sono disponibili per il download in un file .csv di circa 200k record.

Innanzitutto valuto il file con la funzione scan per comprenderene la conformazione:

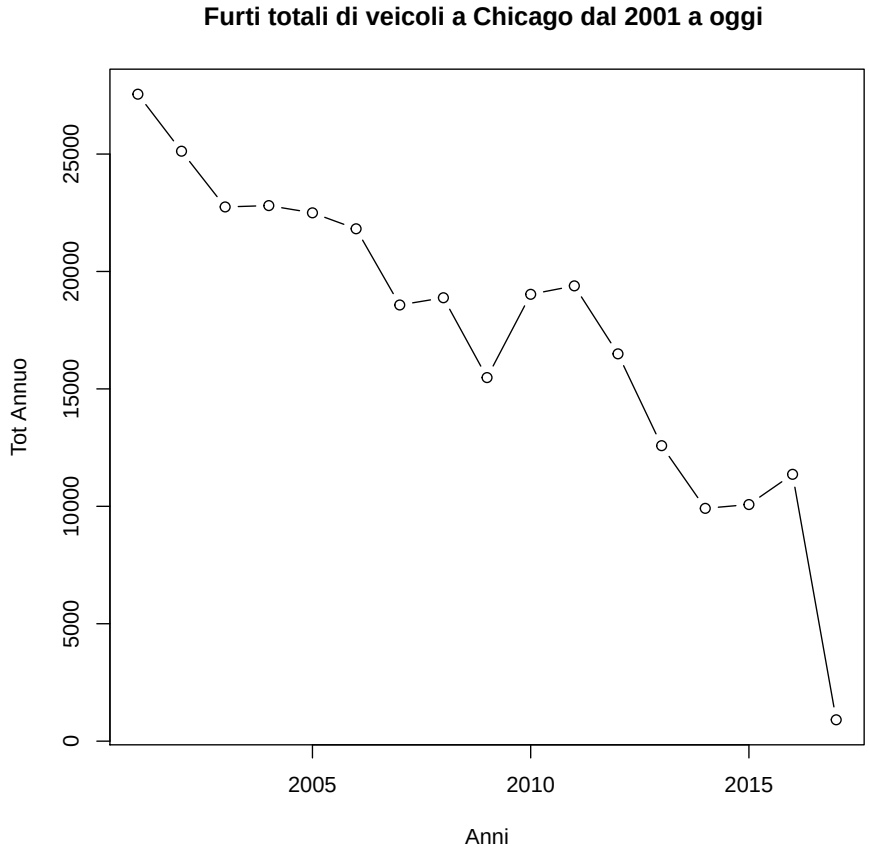
```
library(ggplot2)
library(stringr)
library(data.table)
library(ggmap)
library(dplyr)

## ------------------------------------------------
## data.table + dplyr code now lives in dtplyr.
## Please library(dtplyr)!
## ------------------------------------------------
##
## Attaching package:  'dplyr'
## The following objects are masked from 'package:data.table':
##
##    between, first, last
## The following objects are masked from 'package:stats':
##
##    filter, lag
## The following objects are masked from 'package:base':
##
##    intersect, setdiff, setequal, union

datiChicago <- read.csv('~/Desktop/UNIBO/Statistica /1_2017/dataset/motor_vehicle_theft.csv'
                        stringsAsFactors = FALSE)
str(datiChicago)

## 'data.frame': 295225 obs. of  9 variables:
##  $ Date                : chr  "03/20/2006 11:00:00 PM" "03/15/2006 03:00:00 PM" "03/22/20
##  $ Block               : chr  "064XX S WOLCOTT AVE" "092XX S LANGLEY AVE" "0000X E 118TH
##  $ Primary.Type        : chr  "MOTOR VEHICLE THEFT" "MOTOR VEHICLE THEFT" "MOTOR VEHICLE
##  $ Description         : chr  "AUTOMOBILE" "AUTOMOBILE" "THEFT/RECOVERY: AUTOMOBILE" "AUT
##  $ Location.Description: chr  "STREET" "STREET" "STREET" "STREET" ...
##  $ Ward                : int  15 9 9 6 2 43 47 15 4 15 ...
##  $ Community.Area      : int  67 44 53 69 28 8 3 67 38 67 ...
##  $ Year                : int  2006 2006 2006 2006 2006 2005 2006 2006 2006 2006 ...
##  $ Location            : chr  "(41.776711023, -87.671429342)" "(41.72632605, -87.6069981
```

Ora che ne ho verificato la conformazione posso passare alla valutazione di eventuali dati mancanti

3 Analisi quantitativa dei dati

Al fine di fare un'analisi quantitativa dei dati, decido di guardare in generale l'andamento dei furti in Chicago per anno. Quindi appoggiandomi al pacchetto `dplyr` raggruppo il file per anno e calcolo i totali:

```
totAnnuo <- group_by(datiChicago, Year, Description)

totsPerCateg <- summarize(totAnnuo,count=n())

str(totsPerCateg)

## Classes 'grouped_df', 'tbl_df', 'tbl' and 'data.frame': 174 obs. of  3 variables:
##  $ Year       : int  2001 2001 2001 2001 2001 2001 2001 2001 2001 2002 ...
##  $ Description: chr  "ATT: AUTOMOBILE" "ATT: TRUCK, BUS, MOTOR HOME" "ATTEMPT: CYCLE, SCO
##  $ count      : int  1439 56 3 20692 220 3228 11 274 1626 1092 ...
##  - attr(*, "vars")=List of 1
##   ..$ : symbol Year
##  - attr(*, "drop")= logi TRUE

#aggrego il dataframe precedente per anno sommandone le categorie -

totPerAnno <- aggregate(totsPerCateg$count, by = list(Year=totsPerCateg$Year), FUN = sum)

#rinomino colonna
names(totPerAnno)[names(totPerAnno) == 'x'] <- 'Totale.annuo'

# xlab - ylab = labels. type = b both (linee e punti)

plot(totPerAnno, ylab = "Tot Annuo", xlab = "Anni",
     main = "Furti totali di veicoli a Chicago dal 2001 a oggi ", type = 'b')
```




```
print(summary(totPerAnno))
```

```
##       Year       Totale.annuo
##  Min.   :2001   Min.   :  910
##  1st Qu.:2005   1st Qu.:12582
##  Median :2009   Median :18881
##  Mean   :2009   Mean   :17366
##  3rd Qu.:2013   3rd Qu.:22497
##  Max.   :2017   Max.   :27549
```

Analizzando i dati appena stampanti si nota come ci sia un generale calo dei furti. Il dato che fa sorridere è il picco del 2017. Picco comprensibile in quanto l'anno non è ancora terminato.

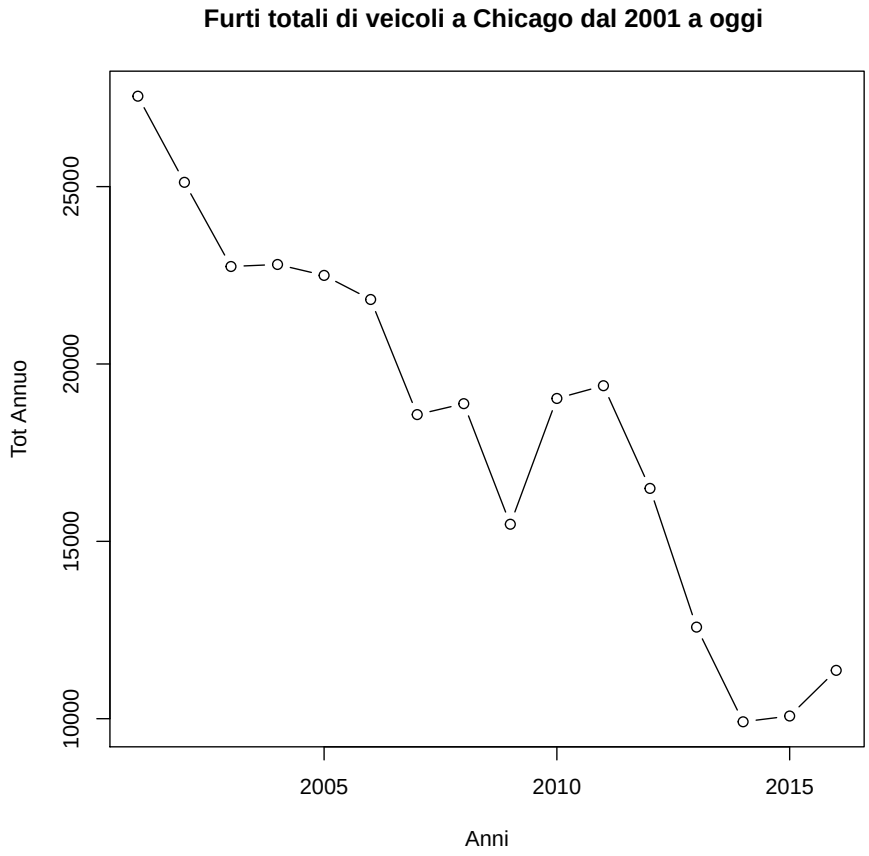
Sarebbe interessante nei paragrafi successivi stimare i furti di veicoli per quest'anno.

A questo punto il 2017 lo tratto come un outlier e per ora non lo prendo in considerazione, aggiornando i dataset rimuovendo i dati inerenti il 2017, e ricalcolo il `summary`, visualizzando anche meduana, massimo e minimo

```
totsPerCateg <- totsPerCateg[!(totsPerCateg$Year == '2017'),]
totPerAnno <- totPerAnno[!(totPerAnno$Year == '2017'),]

knitr::kable(totPerAnno)
```

Year	Totale.annuo
2001	27549
2002	25121
2003	22748
2004	22805
2005	22497
2006	21818
2007	18573
2008	18881
2009	15482
2010	19028
2011	19387
2012	16492
2013	12582
2014	9913
2015	10076
2016	11363

```
plot(totPerAnno, ylab = "Tot Annuo", xlab = "Anni",
     main = "Furti totali di veicoli a Chicago dal 2001 a oggi ", type = 'b')
```



Ora, per studiare i dati in maniera più consona, calcolo le somme mensili per ogni tipologia di furto. Ma per fare questo per evitare errori sui formati di data, converto la colonna `Date` secondo lo standard POSIXct - mentre ora sono salvati in POSIXlt. Successivamente creo una tabella con le righe contenenti sulla prima colonna una sequenza di numeri dal 2001 al 2017 ordinati in maniera crescente divisi separati da un intervallo di 1/12 e calcolo per ogni mese i totali per ciascuno delle tipologie di furto indicate. Per terminare salvo il csv così da poterlo riusare successivamente

```
Sys.setenv(TZ='America/Sao_Paulo')

datiChicago$Date <- as.POSIXct(datiChicago$Date, tz="GMT", format="%m/%d/%Y %I:%M:%S %p")
#creo una tabella vuota dove la prima colonna sono gli anni dal 2006 a og-
gi in forma numerica 2006, 2006.08333-...
monthlyByCateg <- data.frame(Months =seq(as.Date("2001-01-01"),
                                         as.Date("2017-01-01"), by="months" ))
```

```r
#monthlyByCateg <- data.frame(Months = seq(from = 2001, to = 2017, by= 1/12 ))
descrizioni <- as.character(unique(unlist(datiChicago$Description)))
#genero la matrice con dati vuoti
monthlyByCateg[,descrizioni] <-NA

anno = 2001
mese = 1
rowsTot = nrow(monthlyByCateg)
categTot <- length(descrizioni)
#descrizionhe vale come indice colonna per la matrice monthlyByCa-
teg

for(riga in 1: rowsTot){
    #so fare due nested for, ma mi serviva così. ci mette 3 minuti buo-
ni a calcolare tutto
    monthlyByCateg[riga, 2]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[1] & month(Dat
    monthlyByCateg[riga, 3]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[2] & month(Dat
    monthlyByCateg[riga, 4]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[3] & month(Dat
    monthlyByCateg[riga, 5]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[4] & month(Dat
    monthlyByCateg[riga, 6]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[5] & month(Dat
    monthlyByCateg[riga, 7]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[6] & month(Dat
    monthlyByCateg[riga, 8]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[7] & month(Dat
    monthlyByCateg[riga, 9]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[8] & month(Dat
    monthlyByCateg[riga, 10]  <- summarize(subset(datiChicago, Year == anno
                                           & Description == descrizioni[9] & month(Da
    monthlyByCateg[riga, 11]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[10] & month(D
    monthlyByCateg[riga, 12]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[11] & month(D
    monthlyByCateg[riga, 13]  <- summarize(subset(datiChicago, Year == anno
                                          & Description == descrizioni[12] & month(D

    mese = mese+1
    if(riga %% 12 == 0){
      anno = anno +1
      mese = 1
    }
```

```
}

write.csv(monthlyByCateg, file = "monthlyByCateg.csv")
#rinomino colonna
names(monthlyByCateg)[names(monthlyByCateg) == 'Months'] <- 'time'

knitr::kable(summary(monthlyByCateg))
```

	time	AUTOMOBILE	THEFT/RECOVERY: AUTOMOBILE	TRUCK, BUS, MC
	Min. :2001-01-01	Min. : 550	Min. : 31.0	Min. : 8.0
	1st Qu.:2005-01-01	1st Qu.: 897	1st Qu.: 65.0	1st Qu.: 37.0
	Median :2009-01-01	Median :1246	Median : 86.0	Median :139.0
	Mean :2008-12-30	Mean :1191	Mean :111.2	Mean :125.6
	3rd Qu.:2013-01-01	3rd Qu.:1430	3rd Qu.:153.0	3rd Qu.:191.0
	Max. :2017-01-01	Max. :2296	Max. :372.0	Max. :281.0

A questo punto posso creare una funzione che mi gestisce le varie colonne e mi stampa un summary

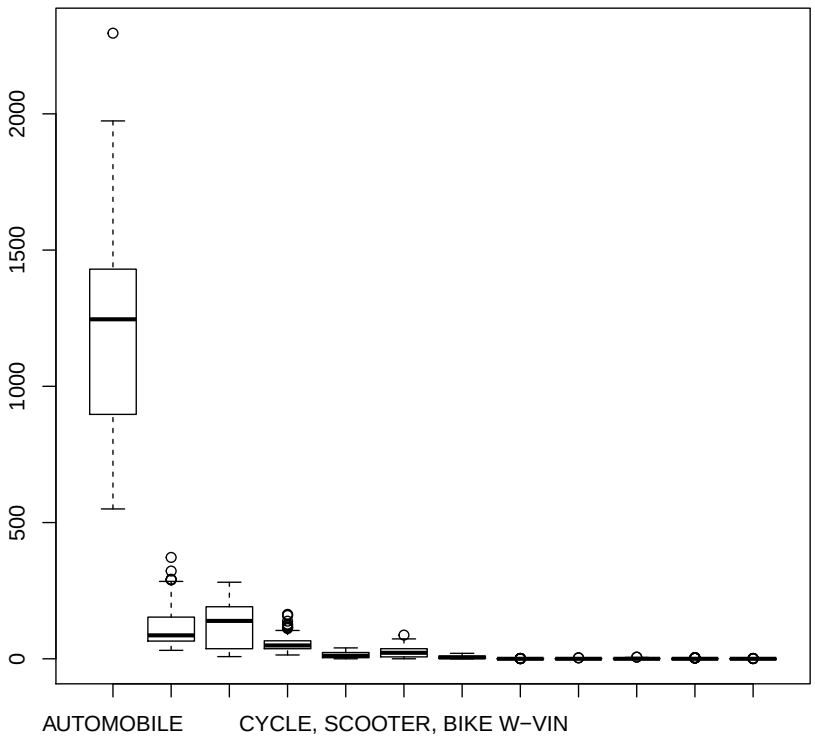
```
analisiDescr <- function(file, colStart, colEnd){
  dataset <- file[,colStart:colEnd]
  dim <- length(dataset)
  #2 significa colonne, #1 righe
  media <- apply(dataset, 2 , mean, na.rm = TRUE)
  varianza <-apply(dataset, 2,var, na.rm = TRUE)
  devStd <- sqrt(varianza)
  minimo <- apply(dataset, 2, min, na.rm = TRUE)
  massimo <- apply(dataset, 2, max, na.rm = TRUE)
  Desc <- data.frame(media, varianza, devStd, minimo, massimo)
  colnames <- c("media", "varianza", "devStd", "minimo", "massimo")
  return(Desc)
}

dati <- analisiDescr(monthlyByCateg, 2, 13)
dati
```

```
##                                                media     varianza
## AUTOMOBILE                                 1.190508e+03 1.018440e+05
## THEFT/RECOVERY: AUTOMOBILE                 1.111969e+02 4.053024e+03
## TRUCK, BUS, MOTOR HOME                     1.256425e+02 6.000668e+03
## ATT: AUTOMOBILE                            5.560104e+01 7.423244e+02
## THEFT/RECOVERY: TRUCK,BUS,MHOME            1.428497e+01 1.309444e+02
## CYCLE, SCOOTER, BIKE W-VIN                 2.438860e+01 3.747701e+02
## ATT: TRUCK, BUS, MOTOR HOME                5.927461e+00 2.306763e+01
## THEFT/RECOVERY: CYCLE, SCOOTER, BIKE NO VIN 5.181347e-02 4.938472e-02
## ATTEMPT: CYCLE, SCOOTER, BIKE W-VIN        4.611399e-01 5.102008e-01
## THEFT/RECOVERY: CYCLE, SCOOTER, BIKE W-VIN  9.792746e-01 1.770402e+00
```

```
## CYCLE, SCOOTER, BIKE NO VIN                 5.906736e-01 9.930376e-01
## ATTEMPT: CYCLE, SCOOTER, BIKE NO VIN        3.108808e-02 3.027850e-02
##                                                   devStd minimo massimo
## AUTOMOBILE                                   319.1300042    550    2296
## THEFT/RECOVERY: AUTOMOBILE                    63.6633610     31     372
## TRUCK, BUS, MOTOR HOME                        77.4639813      8     281
## ATT: AUTOMOBILE                               27.2456304     14     163
## THEFT/RECOVERY: TRUCK,BUS,MHOME               11.4430944      0      40
## CYCLE, SCOOTER, BIKE W-VIN                    19.3589793      0      87
## ATT: TRUCK, BUS, MOTOR HOME                    4.8028770      0      20
## THEFT/RECOVERY: CYCLE, SCOOTER, BIKE NO VIN    0.2222267      0       1
## ATTEMPT: CYCLE, SCOOTER, BIKE W-VIN            0.7142834      0       4
## THEFT/RECOVERY: CYCLE, SCOOTER, BIKE W-VIN     1.3305644      0       6
## CYCLE, SCOOTER, BIKE NO VIN                    0.9965127      0       4
## ATTEMPT: CYCLE, SCOOTER, BIKE NO VIN           0.1740072      0       1
```

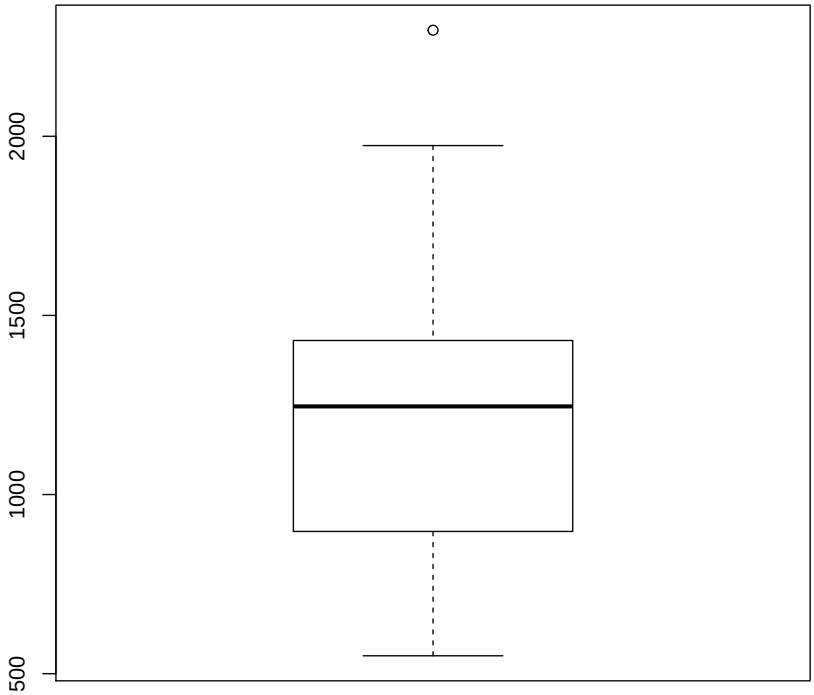
Dall'analisi dei dati notiamo che i dati sui furti di automobili (come quelli di mezzi pesanti) vriano notevolmente. Studio quindi la presenza di outliers. Piccola osservazione: se vi rubano un mezzo a chicago non sperate di recuperarlo, sciocchi.

```
furti <- boxplot(x= as.list(monthlyByCateg[,2:13]))
```

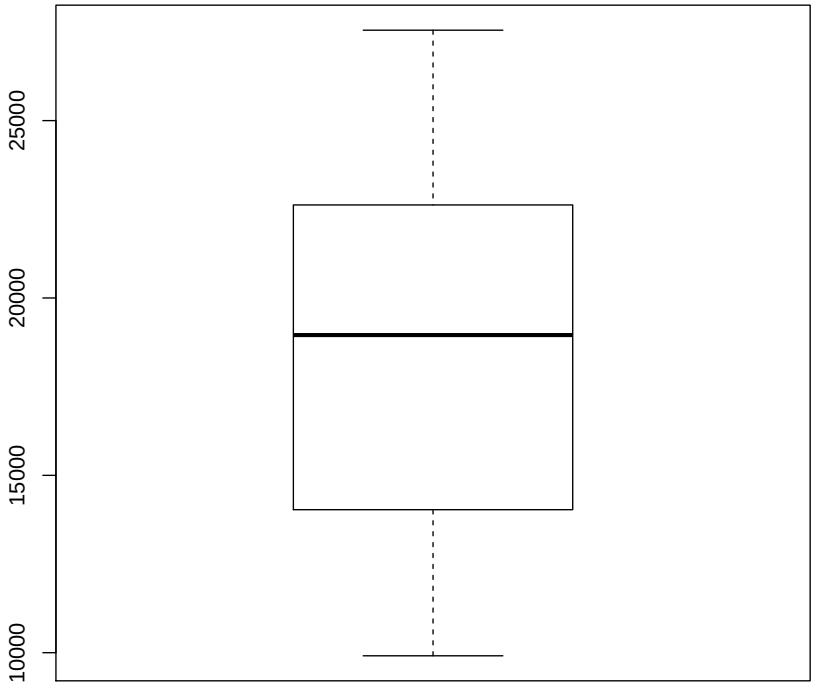


Come si evince dal grafico ci sono degli outliers potenziali sopratutto nei grafici 1, 2, 4 e 6 che, mi accingo a rimuovere tramite identify, che fornisce la possiiblità di intervenire direttamente col mouse sul grafico.

```
furtiAuto <- boxplot(x= monthlyByCateg$AUTOMOBILE)

darimuovere <- identify(rep(3, length(monthlyByCateg[,2]))+ monthlyByCateg[2])
```



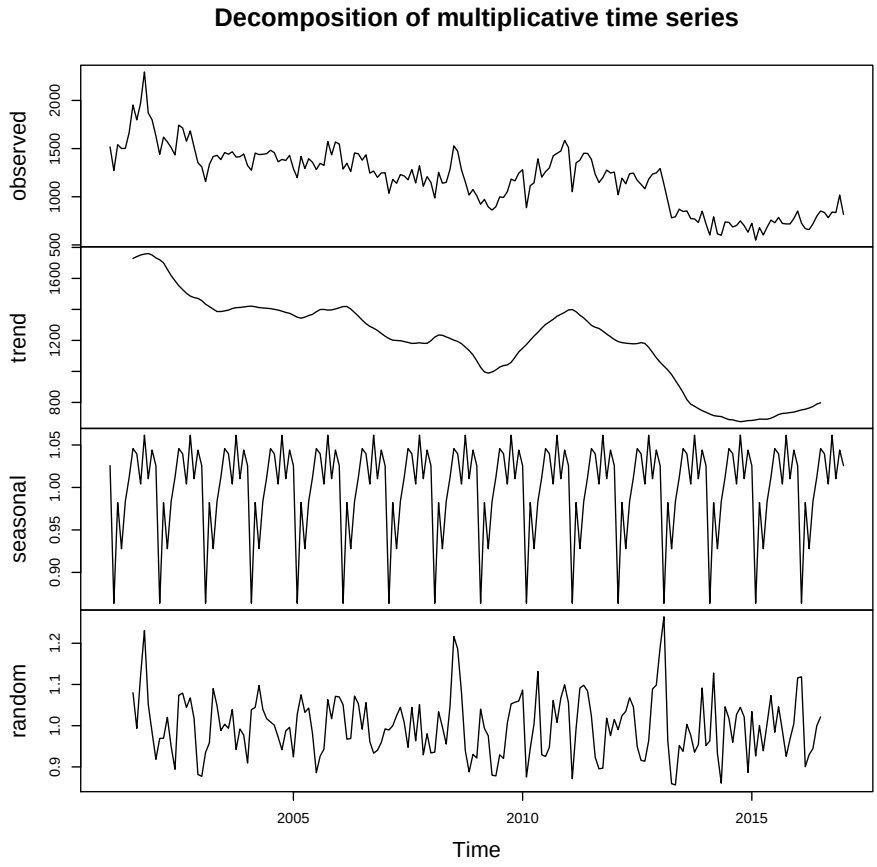

```
boxplot( totPerAnno$Totale.annuo)
```



Sarò sincero... la funzione "identify" mi ha sempre risposto `warning: no point within 0.25 inches` nonostante cliccassi correttamente sul punto outlier segnalato, al che ho proceduto alla rimozione manuale dell'outlier, corrispondente alla misurazione su Ottobre 2001.

Ora studiamo i file come time-series. Il formato TimeSerie permette tutta una serie di analisi sul trend e sulle previsioni grazie alle funzionalità built-in di R. Pertanto converto il dataset tramite la funzione in una serie temporale che ha una ciclicità di 12 mesi. Successivamente applico la funzione decompose che mi permette di studiare la serie temporale stessa.

```
timeSerie <- ts(select(monthlyByCateg, AUTOMOBILE), frequency = 12, start = c(2001,1) )
timeSerieDec <- decompose(timeSerie, type = "multiplicative")
plot(timeSerieDec)
```



Questo grafico, tramite l'analisi di una serie temporale (`timeserie`) mostra:

- Andamento dell'osservazione
- Il trend, che abbastanza assimilabile a un processo lineare
- Stagionalità. Come vediamo c'è una certa ripetitività nel fenomeno e quindi sì, il fenomeno è stagionale (o meglio, nel grafico è annuale)
- Random, ovvero la stima dei dati che non seguono l'andamento (componenti irregolari)

Da una timeserie e il relativo andamento risulta abbastanza ageovole lavorare sulle stime per la creazione di un modello Utilizzando una previsione di Holt,

```
serieFurti <- HoltWinters(timeSerie, beta = FALSE, gamma = FALSE)
serieFurti$fitted
```

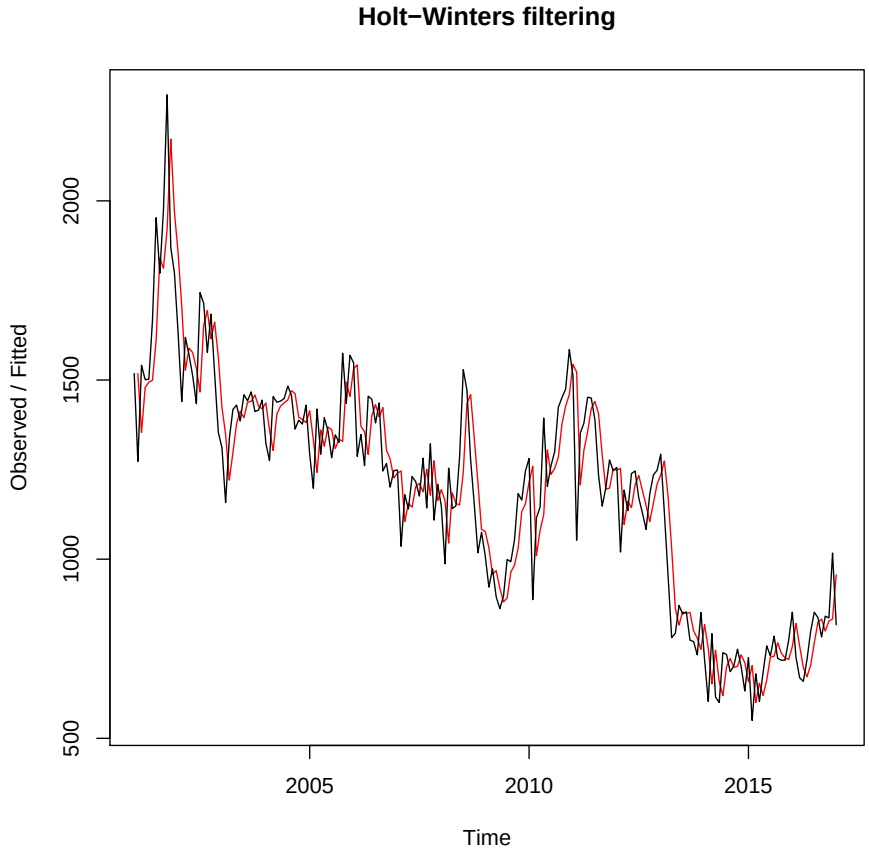
```
##               xhat     level
## Feb 2001 1518.0000 1518.0000
## Mar 2001 1353.6806 1353.6806
## Apr 2001 1479.3141 1479.3141
## May 2001 1493.8587 1493.8587
## Jun 2001 1499.3190 1499.3190
## Jul 2001 1611.7812 1611.7812
## Aug 2001 1840.6337 1840.6337
## Sep 2001 1812.0397 1812.0397
## Oct 2001 1920.6651 1920.6651
## Nov 2001 2172.3990 2172.3990
## Dec 2001 1969.5826 1969.5826
## Jan 2002 1855.1743 1855.1743
## Feb 2002 1705.4932 1705.4932
## Mar 2002 1527.4292 1527.4292
## Apr 2002 1588.8449 1588.8449
## May 2002 1576.2058 1576.2058
## Jun 2002 1532.4728 1532.4728
## Jul 2002 1466.4279 1466.4279
## Aug 2002 1652.5932 1652.5932
## Sep 2002 1693.7782 1693.7782
## Oct 2002 1615.4561 1615.4561
## Nov 2002 1661.4279 1661.4279
## Dec 2002 1564.5613 1564.5613
## Jan 2003 1423.3396 1423.3396
## Feb 2003 1347.9944 1347.9944
## Mar 2003 1220.5668 1220.5668
## Apr 2003 1297.3162 1297.3162
## May 2003 1378.2578 1378.2578
## Jun 2003 1412.9608 1412.9608
## Jul 2003 1394.8784 1394.8784
## Aug 2003 1437.8842 1437.8842
## Sep 2003 1440.6446 1440.6446
## Oct 2003 1458.3210 1458.3210
## Nov 2003 1427.2539 1427.2539
## Dec 2003 1419.7060 1419.7060
## Jan 2004 1435.9998 1435.9998
## Feb 2004 1360.8825 1360.8825
## Mar 2004 1303.2818 1303.2818
## Apr 2004 1404.3672 1404.3672
## May 2004 1426.9244 1426.9244
## Jun 2004 1436.3648 1436.3648
## Jul 2004 1444.1684 1444.1684
## Aug 2004 1470.2124 1470.2124
## Sep 2004 1462.0217 1462.0217
```

```
## Oct 2004 1395.6087 1395.6087
## Nov 2004 1390.5056 1390.5056
## Dec 2004 1381.4475 1381.4475
## Jan 2005 1414.0113 1414.0113
## Feb 2005 1332.1793 1332.1793
## Mar 2005 1242.1864 1242.1864
## Apr 2005 1360.7738 1360.7738
## May 2005 1315.3185 1315.3185
## Jun 2005 1368.7602 1368.7602
## Jul 2005 1361.5434 1361.5434
## Aug 2005 1308.8650 1308.8650
## Sep 2005 1334.4418 1334.4418
## Oct 2005 1328.7800 1328.7800
## Nov 2005 1493.9176 1493.9176
## Dec 2005 1454.4021 1454.4021
## Jan 2006 1531.2619 1531.2619
## Feb 2006 1542.4880 1542.4880
## Mar 2006 1371.1344 1371.1344
## Apr 2006 1355.6183 1355.6183
## May 2006 1292.1586 1292.1586
## Jun 2006 1401.3749 1401.3749
## Jul 2006 1431.3046 1431.3046
## Aug 2006 1396.8950 1396.8950
## Sep 2006 1423.1224 1423.1224
## Oct 2006 1304.3279 1304.3279
## Nov 2006 1279.2924 1279.2924
## Dec 2006 1226.7824 1226.7824
## Jan 2007 1240.3422 1240.3422
## Feb 2007 1246.1489 1246.1489
## Mar 2007 1105.2038 1105.2038
## Apr 2007 1155.3690 1155.3690
## May 2007 1145.0611 1145.0611
## Jun 2007 1202.6996 1202.6996
## Jul 2007 1211.6201 1211.6201
## Aug 2007 1187.7300 1187.7300
## Sep 2007 1250.9561 1250.9561
## Oct 2007 1178.5509 1178.5509
## Nov 2007 1274.7610 1274.7610
## Dec 2007 1163.5865 1163.5865
## Jan 2008 1194.0449 1194.0449
## Feb 2008 1163.1630 1163.1630
## Mar 2008 1045.0120 1045.0120
## Apr 2008 1185.1785 1185.1785
## May 2008 1155.5483 1155.5483
## Jun 2008 1151.1564 1151.1564
```

```
## Jul 2008 1241.5949 1241.5949
## Aug 2008 1434.3550 1434.3550
## Sep 2008 1459.6032 1459.6032
## Oct 2008 1338.4742 1338.4742
## Nov 2008 1214.7489 1214.7489
## Dec 2008 1082.7911 1082.7911
## Jan 2009 1077.5657 1077.5657
## Feb 2009 1032.2500 1032.2500
## Mar 2009  958.3062  958.3062
## Apr 2009  968.1612  968.1612
## May 2009  919.0926  919.0926
## Jun 2009  880.8011  880.8011
## Jul 2009  891.6656  891.6656
## Aug 2009  963.6539  963.6539
## Sep 2009  983.3361  983.3361
## Oct 2009 1030.0591 1030.0591
## Nov 2009 1132.6352 1132.6352
## Dec 2009 1154.3420 1154.3420
## Jan 2010 1215.8162 1215.8162
## Feb 2010 1259.5344 1259.5344
## Mar 2010 1009.6788 1009.6788
## Apr 2010 1080.3168 1080.3168
## May 2010 1123.6993 1123.6993
## Jun 2010 1304.9877 1304.9877
## Jul 2010 1236.5854 1236.5854
## Aug 2010 1252.9601 1252.9601
## Sep 2010 1284.5093 1284.5093
## Oct 2010 1378.0645 1378.0645
## Nov 2010 1426.9817 1426.9817
## Dec 2010 1459.1872 1459.1872
## Jan 2011 1543.5688 1543.5688
## Feb 2011 1522.3959 1522.3959
## Mar 2011 1207.5761 1207.5761
## Apr 2011 1304.4400 1304.4400
## May 2011 1355.1174 1355.1174
## Jun 2011 1420.0957 1420.0957
## Jul 2011 1440.1523 1440.1523
## Aug 2011 1405.8449 1405.8449
## Sep 2011 1290.5900 1290.5900
## Oct 2011 1194.9561 1194.9561
## Nov 2011 1198.3390 1198.3390
## Dec 2011 1251.0963 1251.0963
## Jan 2012 1248.3489 1248.3489
## Feb 2012 1253.4804 1253.4804
## Mar 2012 1096.8871 1096.8871
```

```
## Apr 2012 1161.3492 1161.3492
## May 2012 1143.6770 1143.6770
## Jun 2012 1207.6093 1207.6093
## Jul 2012 1233.3576 1233.3576
## Aug 2012 1191.5349 1191.5349
## Sep 2012 1149.5933 1149.5933
## Oct 2012 1104.9297 1104.9297
## Nov 2012 1156.6201 1156.6201
## Dec 2012 1209.1888 1209.1888
## Jan 2013 1235.2192 1235.2192
## Feb 2013 1273.9723 1273.9723
## Mar 2013 1176.7406 1176.7406
## Apr 2013 1028.6918 1028.6918
## May 2013  862.5670  862.5670
## Jun 2013  815.9090  815.9090
## Jul 2013  852.8581  852.8581
## Aug 2013  848.9291  848.9291
## Sep 2013  851.6594  851.6594
## Oct 2013  799.5739  799.5739
## Nov 2013  779.7389  779.7389
## Dec 2013  748.3915  748.3915
## Jan 2014  817.2102  817.2102
## Feb 2014  751.3414  751.3414
## Mar 2014  651.8501  651.8501
## Apr 2014  745.8474  745.8474
## May 2014  658.0892  658.0892
## Jun 2014  619.1293  619.1293
## Jul 2014  699.5255  699.5255
## Aug 2014  722.6473  722.6473
## Sep 2014  698.0683  698.0683
## Oct 2014  700.7052  700.7052
## Nov 2014  733.0961  733.0961
## Dec 2014  710.8988  710.8988
## Jan 2015  657.9821  657.9821
## Feb 2015  702.9304  702.9304
## Mar 2015  600.3613  600.3613
## Apr 2015  653.7743  653.7743
## May 2015  619.7204  619.7204
## Jun 2015  662.8322  662.8322
## Jul 2015  726.6604  726.6604
## Aug 2015  728.9003  728.9003
## Sep 2015  766.5259  766.5259
## Oct 2015  737.3334  737.3334
## Nov 2015  724.3667  724.3667
## Dec 2015  720.0966  720.0966
```

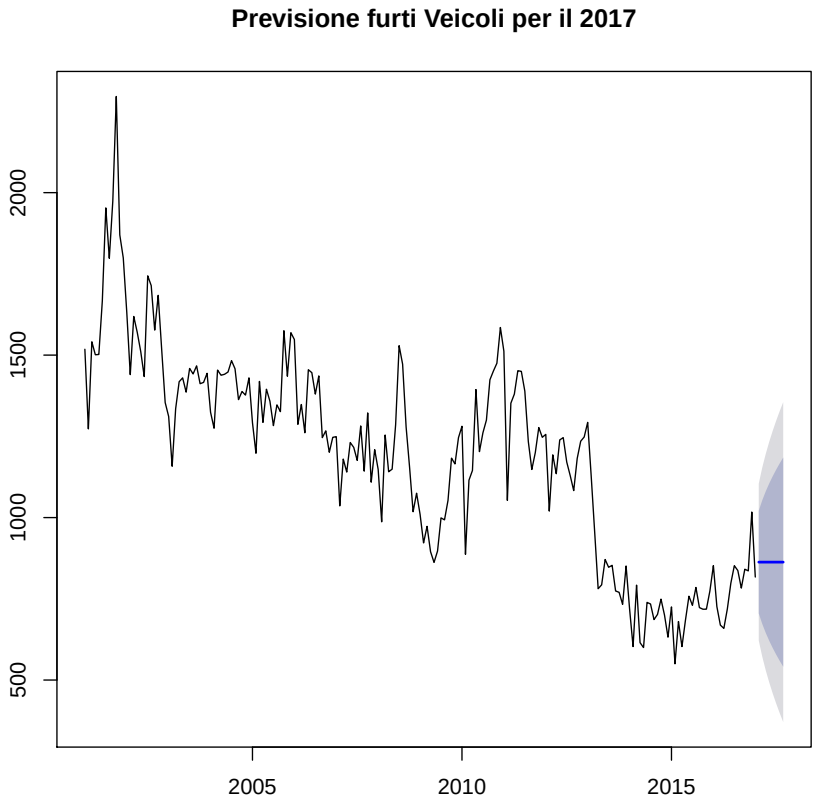
```
## Jan 2016  756.2492  756.2492
## Feb 2016  820.4684  820.4684
## Mar 2016  757.1093  757.1093
## Apr 2016  698.0151  698.0151
## May 2016  671.8480  671.8480
## Jun 2016  703.4724  703.4724
## Jul 2016  766.8713  766.8713
## Aug 2016  823.9664  823.9664
## Sep 2016  832.7079  832.7079
## Oct 2016  799.3692  799.3692
## Nov 2016  827.2906  827.2906
## Dec 2016  833.1319  833.1319
## Jan 2017  956.4507  956.4507

plot(serieFurti)
```



```r
require(forecast)
```

```
## Loading required package:  forecast
```

```r
serieFurti2 <- forecast::forecast.HoltWinters(serieFurti, h = 8)
plot(serieFurti2, main = "Previsione furti Veicoli per il 2017")
```




```r
acf(serieFurti2$residuals, lag.max = 20)
```

```
## Error in na.fail.default(as.ts(x)):  missing values in object
```

La stima di `Holt, (1957)` ci permette di prevedere l'andamento del fenomeno preso in esame utilizzando 3 equazioni:

Una equazione per la previsione:

$$\hat{y}_{t+h|t} = l_t + hb_t$$

Una equazione di smoothing per il livello:

$$l_t = \alpha y_t + (1 - \alpha)(l_{t-1} + b_{t-1})$$

Una equazione di smoothing per il trend:

$$b_t = \beta^*(l_t - l_{t-1}) + (1 + \beta^*)b_{t-1}$$

4 Analisi delle distribuzioni

Analizzo la correlazione

```
corr
```

```
## Error in eval(expr, envir, enclos):  oggetto "corr" non trovato
```

La mia analisi ora vuole spostarsi sui giorni (e le relative ore) al fine di valutare tramite un grafico heat, quali siano le ore più rischiose.

```
par(mfrow = c(2,2))

y <- c("Fn(Auto)", "Fn(RD)", "Fn(ROE)")
x <- c("Auto", "Ritrovamenti", "ROE")

plot(ecdf(monthlyByCateg[,2]), main = "Auto", xlab = x[1], ylab = y[1])
plot(ecdf(monthlyByCateg[,3]), main = "Attempted auto", xlab = x[1], ylab = y[1])
plot(ecdf(monthlyByCateg[,5]), main = "Recovered Auto", xlab = x[1], ylab = y[1])


prova <- filter(tots, Description ==   "ATT: AUTOMOBILE", Year == '2001')
```

```
## Error in filter_(.data, .dots = lazyeval::lazy_dots(...)):  oggetto
"tots" non trovato
```

```
#tipologie di descrizione
n_distinct(tots$Description)
```

```
## Error in n_distinct_multi(list(...), na.rm):  oggetto "tots" non
trovato
```

```
tots.transposed <- t(tots[,3])
```

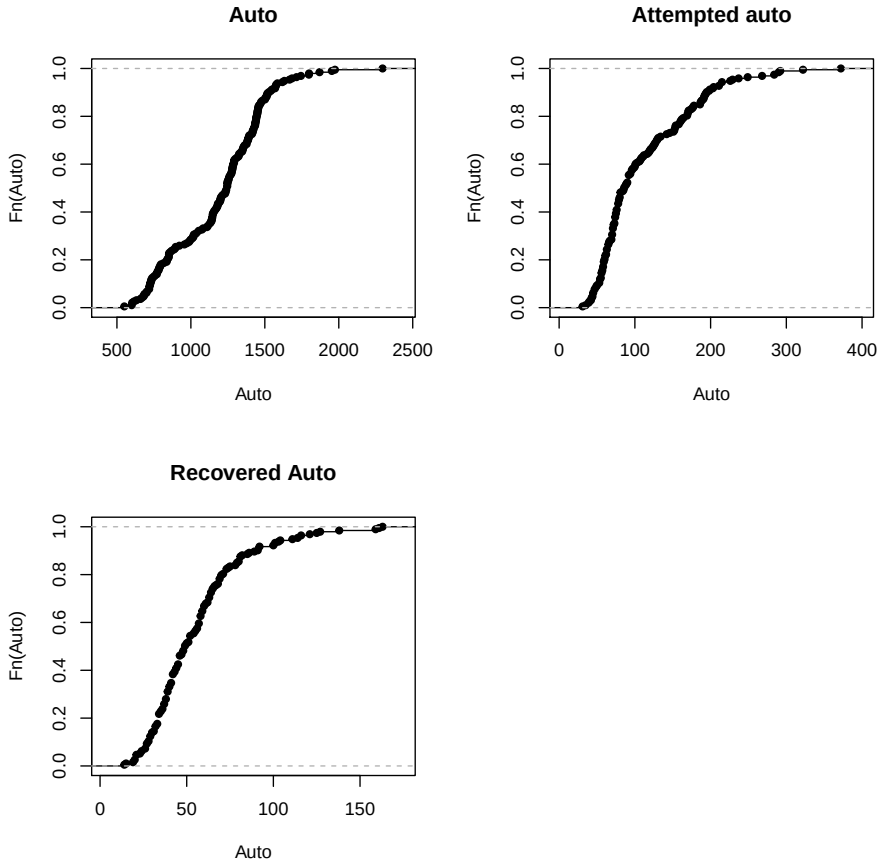
```
## Error in t(tots[, 3]):  oggetto "tots" non trovato
```

```
boxplot(filter(tots,tots$Year == 2005))
```

```
## Error in filter_(.data, .dots = lazyeval::lazy_dots(...)):  oggetto
"tots" non trovato
```

```
qplot(data = tots, x = Year, y = count, )
```

```
## Error in ggplot(data, aesthetics, environment = env):  oggetto "tots"
non trovato
```



Noto immediatamente che il 2017 ha solo 910 furti. Chiaramente essendo il 2017 iniziato da poco, questo valore è da trattare adeguatamente per non sfalsare la nostra analisi.

Ora risulterebbe più interessante accoppiare il totale di ogni anno con le tipologie di furto (colonna `Description`)