Multiple Linear Regression

Here, we will use the MTS package to compute forecasts of the log gas consumption, basing ourselves on : - the precedent values of the log gas consumption - the selected temperature metrics, that is to say min_temp_val and heat_days_val - the computed dummy variables extreme_heat and extreme_cold

Load data

As usual, we load the datasets.

```
library(readr)
library(forecast)
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: timeDate

## This is forecast 7.3
```

```
gas <- na.omit(read_csv("data/gas.csv", col_types = cols(date = col_date(format = "%Y-%m-%d"))))
temp <- read_csv("data/temp.csv", col_types = cols(date = col_date(format = "%Y-%m-%d")))
google <- read_csv("data/google.csv", col_types = cols(date = col_date(format = "%Y-%m-%d")))
```

As said before, our datasets span over different periods of time: while gas consumption, our target value goes from January 1973 to July 2016, temperature data start in January 1895 and end in February 2017. Finally, data from Google Trends go from january 2004 and up to march 2017.

We will thus need to use the predicted values : - of min_temp_val and heat_days_val from February 2017 to July 2017 - of extreme_heat and extreme_cold from March 2017 to July 2017.

Loading these predictions obtained previously:

```
library(readxl)
inter_preds <- read_excel("data/inter_preds.xlsx")
inter_preds
```

```
## # A tibble: 12 × 5
##          date min_temp_val heat_days_val extreme_heat extreme_cold
##        <dttm>        <chr>         <chr>        <chr>        <chr>
## 1  2017-01-03     34.76901     409.66003           NA           NA
## 2  2017-01-04     42.88134     153.59266  0.149622092 -0.207865064
## 3  2017-01-05     51.74640             0  0.687561430 -0.001670723
## 4  2017-01-06     60.01983             0  0.765578706  0.157808081
## 5  2017-01-07     64.90640             0  0.997441984  0.210642955
## 6  2017-01-08     63.40806             0  0.537571551  0.173692380
## 7        <NA>         <NA>          <NA>         <NA>         <NA>
## 8        <NA>         <NA>          <NA>         <NA>         <NA>
## 9        <NA>         <NA>          <NA>         <NA>         <NA>
## 10       <NA>         <NA>          <NA>         <NA>         <NA>
## 11       <NA>         <NA>          <NA>         <NA>         <NA>
```

```
## 12       <NA>          <NA>           <NA>          <NA>          <NA>
```

Here extreme_heat and extreme_cold have NAs for april because we have the actual values in the "google" dataset.

Preparation

Let us merge the dataframes

```
gas_temp = merge(gas, temp, by = "date", all.x = TRUE)
gas_goo = merge(gas, google, by = "date", all = FALSE)
all = merge(gas_temp, google, by= "date",  all = FALSE)

extreme <- all[,c(16,17,18)]

# extreme_heat will be 1 when heatwave > mean(heatwave)
extreme_heat <- as.numeric(extreme$heatwave > mean(extreme$heatwave))
# extreme_cold will be 1 when snow_storm > mean(snow_storm) and extreme_weather > mean(extreme_weather)
extreme_cold <- as.numeric(extreme$snow_storm > mean(extreme$snow_storm) & extreme$extreme_weather > mea

all = cbind(all, extreme_heat, extreme_cold)
```

Let us create the test and train set

```
all_test = all[all$date >= '2015-06-15' & all$date < '2016-06-15',]
all_train = all[all$date < '2015-06-15',]
nrow(all_train)
```

```
## [1] 137
```

Multiple linear regression

We will test different combinasons.

```
train = all_train[,c("log_gas_cons", "min_temp_val", "heat_days_val", "extreme_heat", "extreme_cold")]
test = all_test[,c("log_gas_cons", "min_temp_val", "heat_days_val", "extreme_heat", "extreme_cold")]

linreg_all = lm(log_gas_cons ~., data = train)
pred_linreg = predict.lm(linreg_all, newdata = test)
accuracy(test$log_gas_cons, pred_linreg)
```

```
##                   ME      RMSE       MAE        MPE      MAPE
## Test set -0.01777704 0.1232155 0.0959117 -0.1098576 0.7682952
```

```
summary(linreg_all)
```

```
##
## Call:
## lm(formula = log_gas_cons ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28690 -0.09441  0.00308  0.07893  0.39249
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)   13.5281299  0.2302129  58.764  < 2e-16 ***
## min_temp_val  -0.0324437  0.0041250  -7.865 1.16e-12 ***
## heat_days_val  0.0011184  0.0001821   6.143 8.90e-09 ***
## extreme_heat   0.0987691  0.0334290   2.955  0.00371 **
## extreme_cold  -0.0821180  0.0358659  -2.290  0.02363 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1339 on 132 degrees of freedom
## Multiple R-squared:  0.9716, Adjusted R-squared:  0.9707
## F-statistic:  1129 on 4 and 132 DF,  p-value: < 2.2e-16
```

```
train = all_train[,c("log_gas_cons", "min_temp_val", "heat_days_val")]
test = all_test[,c("log_gas_cons", "min_temp_val", "heat_days_val")]

linreg_2 = lm(log_gas_cons ~., data = train)
pred_linreg = predict.lm(linreg_2, newdata = test)
accuracy(test$log_gas_cons, pred_linreg)
```

```
##                    ME      RMSE        MAE         MPE     MAPE
## Test set -0.006035521 0.1141792 0.09048187 -0.02258079 0.726324
```

```
summary(linreg_2)
```

```
##
## Call:
## lm(formula = log_gas_cons ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28459 -0.09230 -0.00045  0.09242  0.40091
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   13.420381   0.225851  59.421  < 2e-16 ***
## min_temp_val  -0.029489   0.003946  -7.473 9.11e-12 ***
## heat_days_val  0.001108   0.000176   6.299 3.99e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.14 on 134 degrees of freedom
## Multiple R-squared:  0.9685, Adjusted R-squared:  0.968
## F-statistic:  2061 on 2 and 134 DF,  p-value: < 2.2e-16
```

```
train = all_train[,c("log_gas_cons", "min_temp_val", "heat_days_val","extreme_heat")]
test = all_test[,c("log_gas_cons", "min_temp_val", "heat_days_val", "extreme_heat")]

linreg_3 = lm(log_gas_cons ~., data = train)
pred_linreg = predict.lm(linreg_3, newdata = test)
accuracy(test$log_gas_cons, pred_linreg)
```

```
##                   ME      RMSE        MAE          MPE      MAPE
## Test set -0.00388973 0.1030723 0.08373292 -0.003553291 0.6731092
```

```
summary(linreg_3)
```

```
##
```

```
## Call:
## lm(formula = log_gas_cons ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.29332 -0.09345 -0.00673  0.08225  0.41752
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)   13.6243371  0.2299271  59.255  < 2e-16 ***
## min_temp_val  -0.0340485  0.0041293  -8.246 1.39e-13 ***
## heat_days_val  0.0009884  0.0001757   5.625 1.05e-07 ***
## extreme_heat   0.1011046  0.0339421   2.979  0.00344 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.136 on 133 degrees of freedom
## Multiple R-squared:  0.9705, Adjusted R-squared:  0.9698
## F-statistic:  1457 on 3 and 133 DF,  p-value: < 2.2e-16
```

The best linear regression model is thus obtained with "log_gas_cons", "min_temp_val", "heat_days_val" and "extreme_heat".

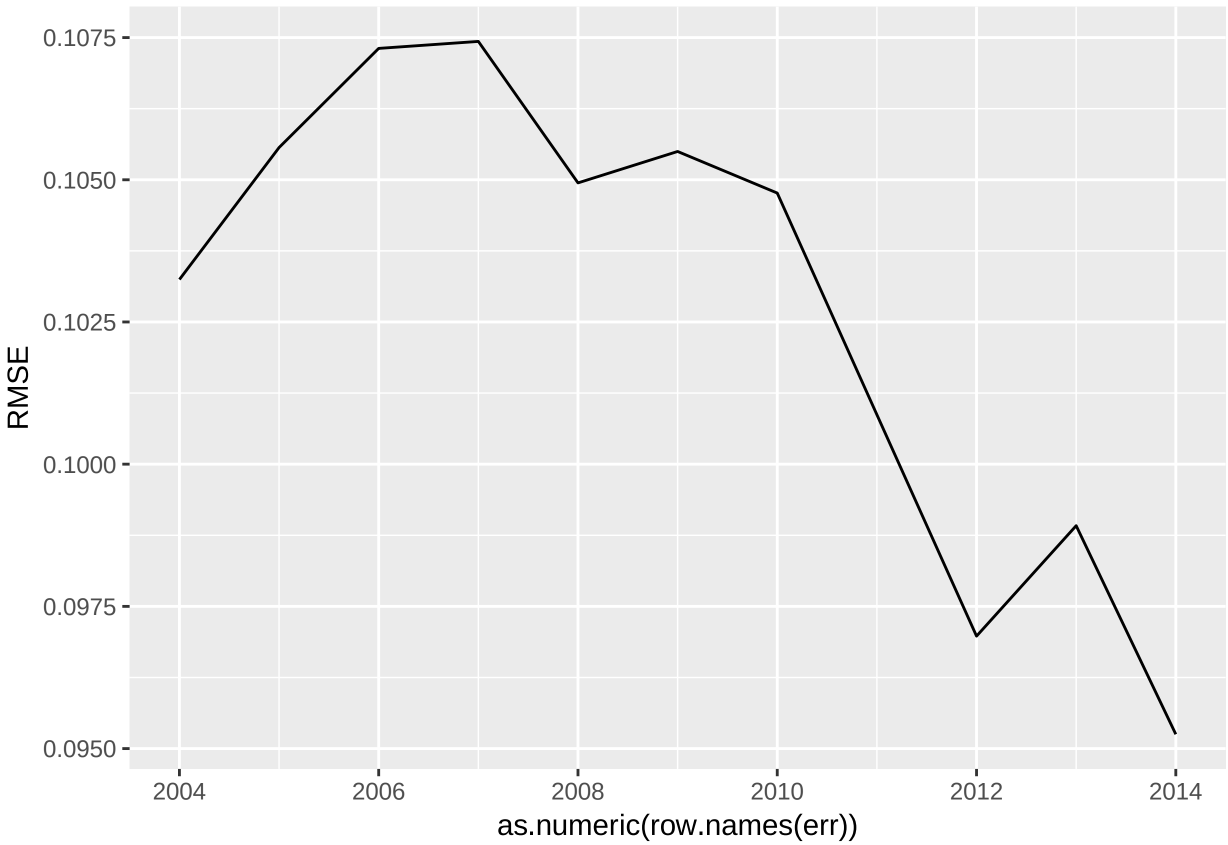
Let's convert in Time series

```
train = all_train[,c("log_gas_cons", "min_temp_val", "heat_days_val","extreme_heat")]
test = all_test[,c("log_gas_cons", "min_temp_val", "heat_days_val", "extreme_heat")]

strain = ts(train, start = c(2004, 1), frequency=12)
train = window(strain, start = c(2004, 1), end = c(2015, 2))
stest = ts(test, start = c(2015,3), frequency=12)
test = window(stest, start = c(2015,3), end = c(2016, 2))
```

Rolling Window Evaluation

To try more model, we would like to setup a rolling window. Let us find the optimal size of that window using the multiple linear regression we just selected.

```
library(ggplot2)
err = data.frame("RMSE" = rep(0, 2014-2004+1), row.names = seq(2004, 2014))
for (y in 2004:2014){
    model = lm(log_gas_cons ~., data = window(train, start=c(y,1)))
    rmse = accuracy(predict.lm(model, newdata = test),test[,1])[2]
    err[as.character(y),] = rmse
}
ggplot(err)+geom_line(aes(x=as.numeric(row.names(err)), y=RMSE))
```



So we'll take 2 years of data (since the result is 2012).

Final prediction : including both the predicted (when missing) and the actual temperatures

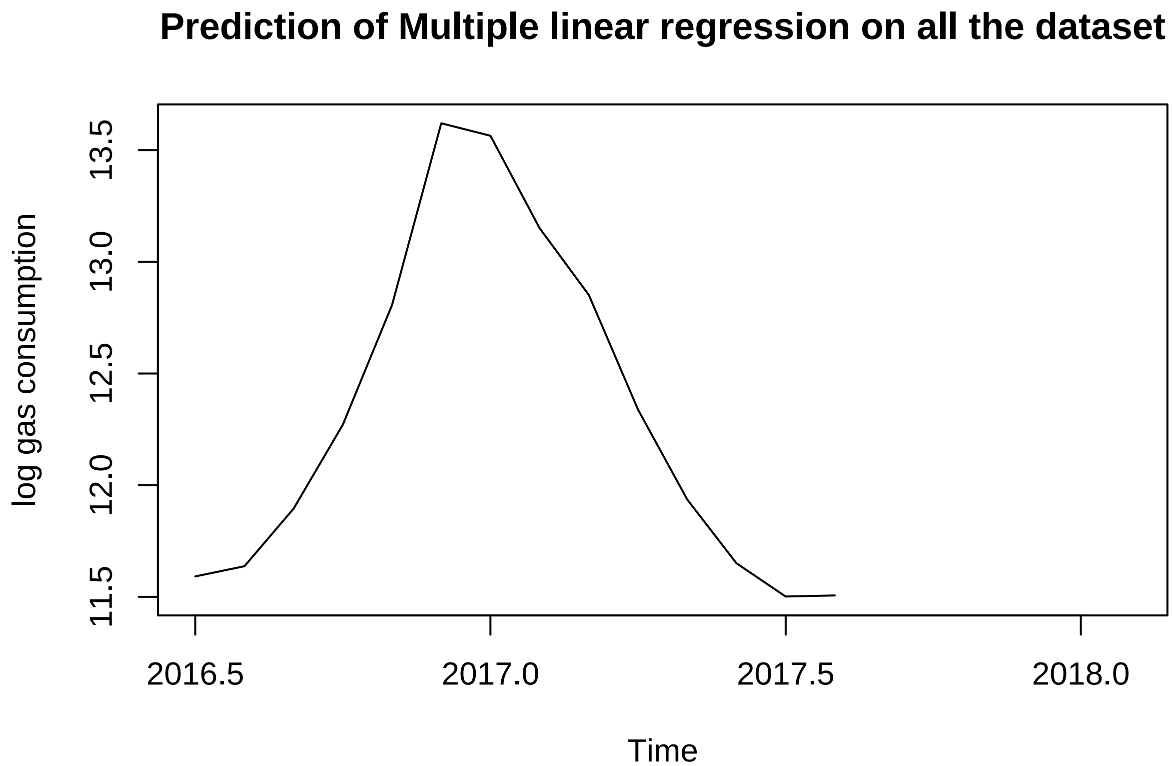
Preparation of the databases (complicated here because we have a datapoint more in extreme_heat, ie google database):

```
true_extreme_heat <- ts(as.numeric(google$heatwave > mean(google$heatwave)), start = c(2004, 1), freque
pred_extreme_heat <- ts(inter_preds$extreme_heat[-1], start = c(2017, 4), frequency=12)

true_min_temp_val = ts(temp$min_temp_val, start = c(1895, 1), frequency=12)
true_heat_days_val = ts(temp$heat_days_val, start = c(1895, 1), frequency=12)

pred_min_temp_val = ts(inter_preds$min_temp_val, start = c(2017, 3), frequency=12)
pred_heat_days_val = ts(inter_preds$heat_days_val, start = c(2017, 3), frequency=12)

tslog_gas_cons = ts(gas$log_gas_cons, start = c(1973,1), frequency = 12)

data = as.data.frame(cbind(
    window(tslog_gas_cons, start = c(2004,1), end = c(2016, 6)),
    window(true_min_temp_val, start = c(2004,1), end = c(2016, 6)),
    as.double(window(true_heat_days_val, start = c(2004,1), end = c(2016, 6))),
    as.double(window(true_extreme_heat, start = c(2004,1), end = c(2016, 6)))
    ))
colnames(data) = c("log_gas_cons", "min_temp_val", "heat_days_val", "extreme_heat")

nd_extreme_heat = ts(c(window(true_extreme_heat, start = c(2016, 7)),pred_extreme_heat), start = c(2016
nd_min_temp_val = ts(c(window(true_min_temp_val, start = c(2016, 7)),pred_min_temp_val), start = c(2016
```

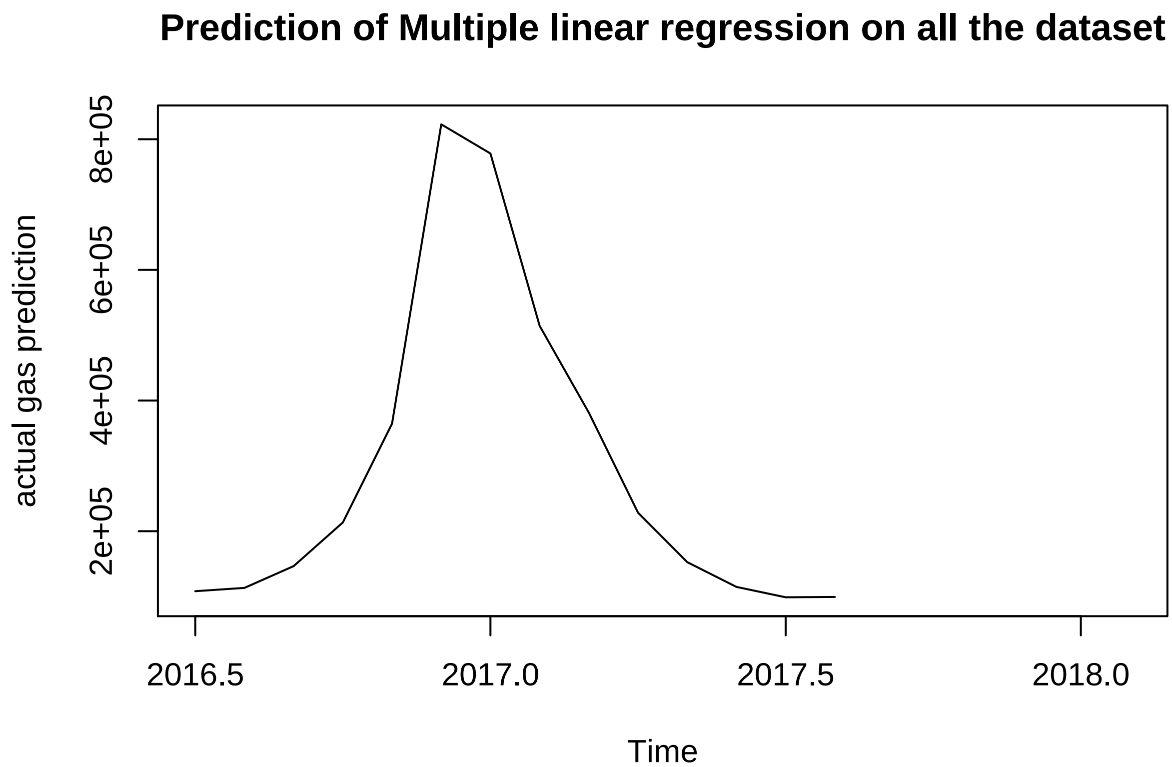
```
nd_heat_days_val = ts(c(window(true_heat_days_val, start = c(2016, 7)),as.double(pred_heat_days_val)), s

newdata = as.data.frame(cbind(as.double(nd_min_temp_val), as.double(nd_heat_days_val), as.double(nd_extr
colnames(newdata) = c("min_temp_val", "heat_days_val", "extreme_heat")
```

Fitting the model on all the dataset and plotting the predictions :

```
model = lm(log_gas_cons ~., data = data)
prediction = predict.lm(model, newdata = newdata)
prediction = ts(prediction, start = c(2016,7), frequency = 12)
plot(prediction, main = "Prediction of Multiple linear regression on all the dataset", ylab = "log gas c
```

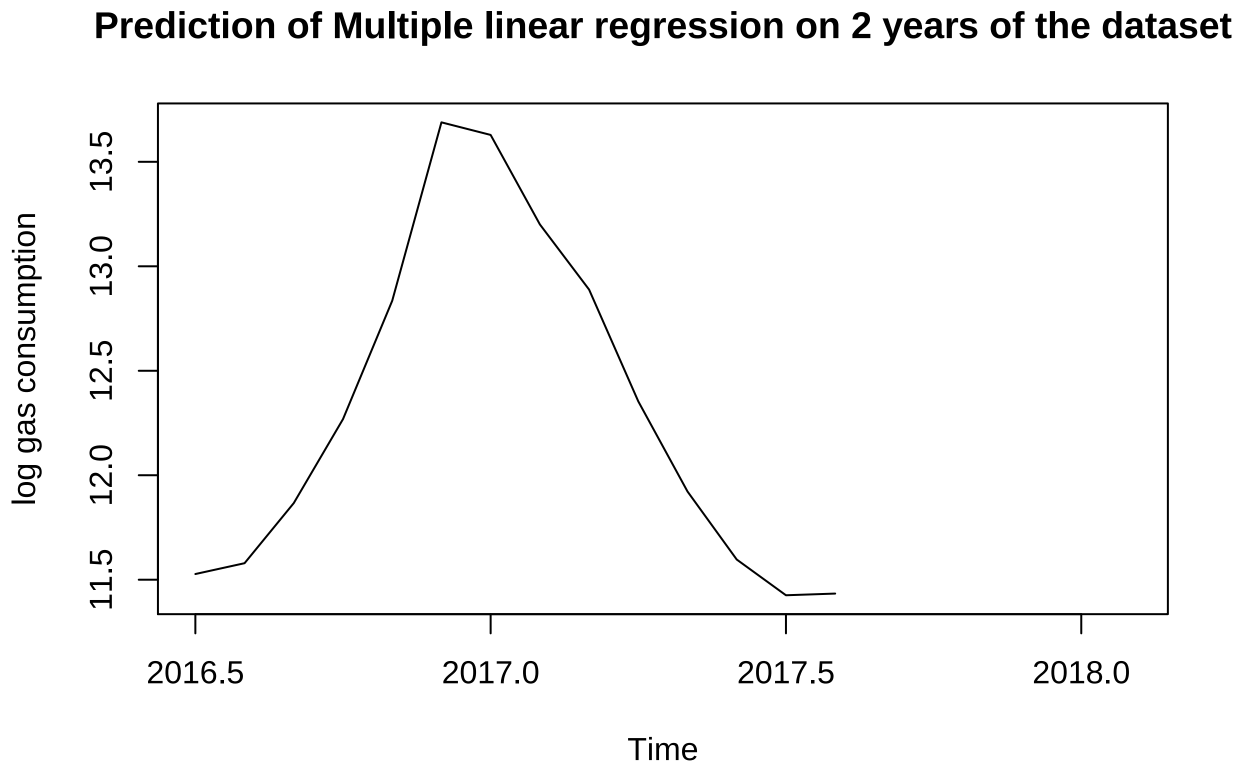



```
plot(exp(prediction), main = "Prediction of Multiple linear regression on all the dataset", ylab = "actu
```



Fitting the model on only the two last years and plotting the predictions :

```
data2 = data[127:150,]
model2 = lm(log_gas_cons ~., data = data2)
prediction2 = predict.lm(model2, newdata = newdata)
prediction2 = ts(prediction2, start = c(2016,7), frequency = 12)
plot(prediction2, main = "Prediction of Multiple linear regression on 2 years of the dataset", ylab = "
```




```r
plot(exp(prediction2), main = "Prediction of Multiple linear regression on 2 years of the dataset", yla
```