Totvs Challenge

Rui Romanini

17 de dezembro de 2016

Objective

The objective of this challenge is using the given dataset of transactions (notas fiscais eletrônicas) from a restaurant:

1. Parse and extract the data.
2. Identify a pattern on any set of fields that can help predict how much a customer will spend.
3. Calculate a sales forecast for the next week.

The environment

It's important to pay atention to the environment, to make sure about reproducibility.

This code was produced using R version 3.3.1 and R Studio version 0.99.489

The Operational System it's Windows 10 Home

```
if(!require(jsonlite)){
  install.packages("jsonlite", repos ="http://cran.us.r-project.org")
  library(jsonlite)
}
```

```
## Loading required package: jsonlite

## Warning: package 'jsonlite' was built under R version 3.3.2
```

```
if(!require(plyr)){
  install.packages("plyr", repos ="http://cran.us.r-project.org")
  library(plyr)
}
```

```
## Loading required package: plyr

## Warning: package 'plyr' was built under R version 3.3.2
```

```
if(!require(lubridate)){
  install.packages("lubridate", repos ="http://cran.us.r-project.org")
  library(lubridate)
}
```

```
## Loading required package: lubridate
```

```
## Warning: package 'lubridate' was built under R version 3.3.2
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:plyr':
##
##     here
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
if(!require(nnet)){
  install.packages("nnet", repos ="http://cran.us.r-project.org")
  library(nnet)
}
```

```
## Loading required package: nnet
```

```
## Warning: package 'nnet' was built under R version 3.3.2
```

Getting data

We choose to download the zip file from github, as challengeTotvs.zip and unzip the content as sample.txt in the current directory.

```
temp <- "./challengeTotvs.zip"

fileURL = "https://github.com/TOTVS/MDMStatic/blob/master/code-challenge/TOTVS%20Labs%20-%20AI%20Challen

#download original file from github
download.file(fileURL,destfile=temp, mode="wb")

#unzip file sample.txt
td <- tempdir()
fname <- unzip(temp, list=TRUE)$Name[1]
unzip(zipfile = temp, files=fname, exdir=td,overwrite=TRUE)

#read the file sample.txt
setwd(td)
raw.data <- readLines("sample.txt", warn = "F")
```

Parsing the data

My choice was the jsonlite package to help our work in parsing the json code.

```r
rd <- fromJSON(raw.data)
```

Exploring the data

The records are structured in seven sections:
- complemento (valorTotal)
- dets (Detail about the operation)
- emit (Data about the "Emitente")
- ide (The operation and date)
- infAdic (Complementary information about the operation)
- total (Taxes)
- versaoDocumento (Versioning of document/record json)

According with the tests bellow, there is 1635 records in the document, and no null values in dataset.

The histogram and boxplot make clear that most of the sales have total between 30.98 and 120, but range from 9 until 608.

```r
# Document size / Number of records from file sample.txt
nrow(rd$ide)
```

```
## [1] 1635
```

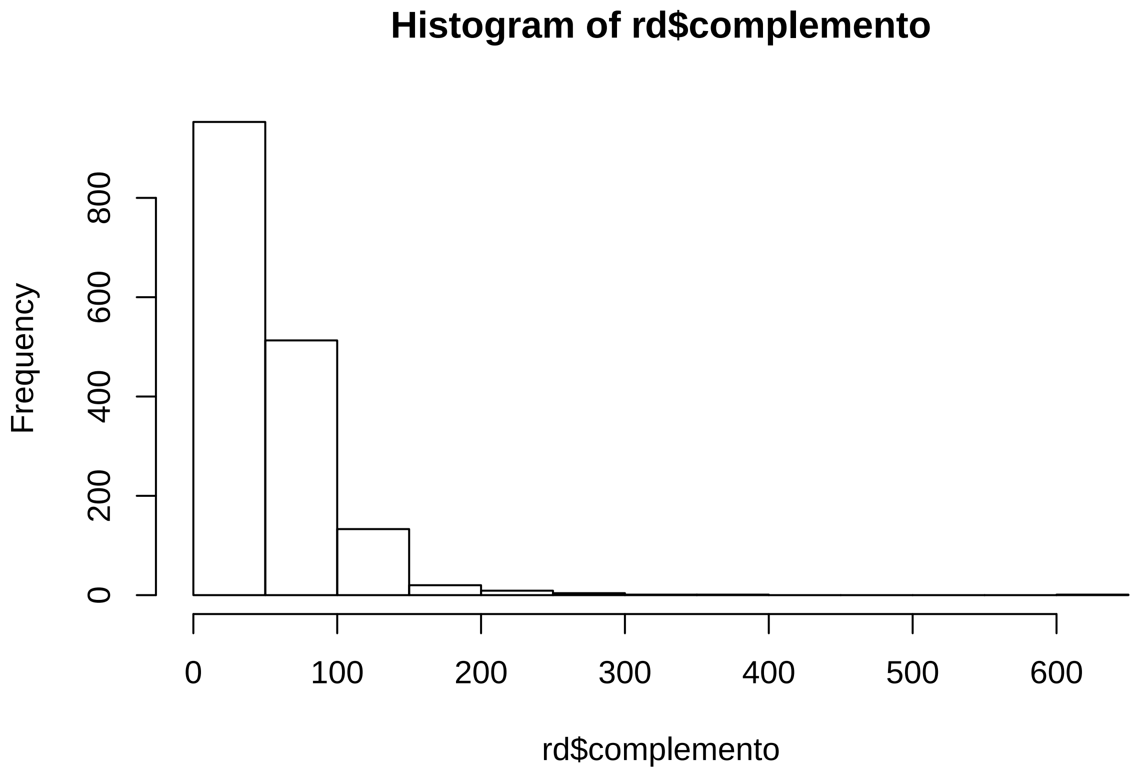
```r
# Number of null values
length(which(is.na(rd) == TRUE))
```

```
## [1] 0
```

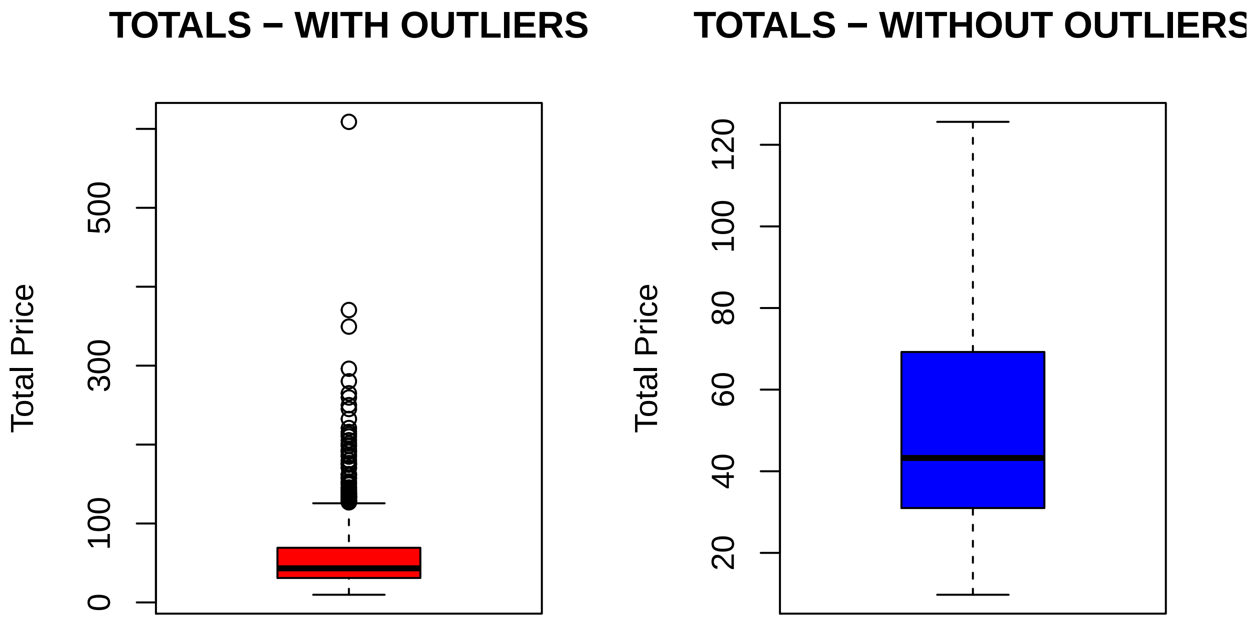
```r
#summary from valorTotal, in complemento section
summary(rd$complemento)
```

```
##    valorTotal
##  Min.   :  9.74
##  1st Qu.: 30.98
##  Median : 43.26
##  Mean   : 55.50
##  3rd Qu.: 69.23
##  Max.   :608.91
```

```r
#Conversion necessary for plot a histogram
rd$complemento <- data.matrix(rd$complemento)

#Distribution of Totals Sales (rd$complemento)
hist(rd$complemento)
```



```r
par(mfrow=c(1,2))

#Resume about main statistics related to Totals (rd$complemento)
boxplot(rd$complemento,ylab="Total Price",main= toupper("Totals - With outliers"),col=c("red"))

boxplot(rd$complemento,ylab="Total Price",main= toupper("Totals - Without outliers"),col=c("blue"),outl
```



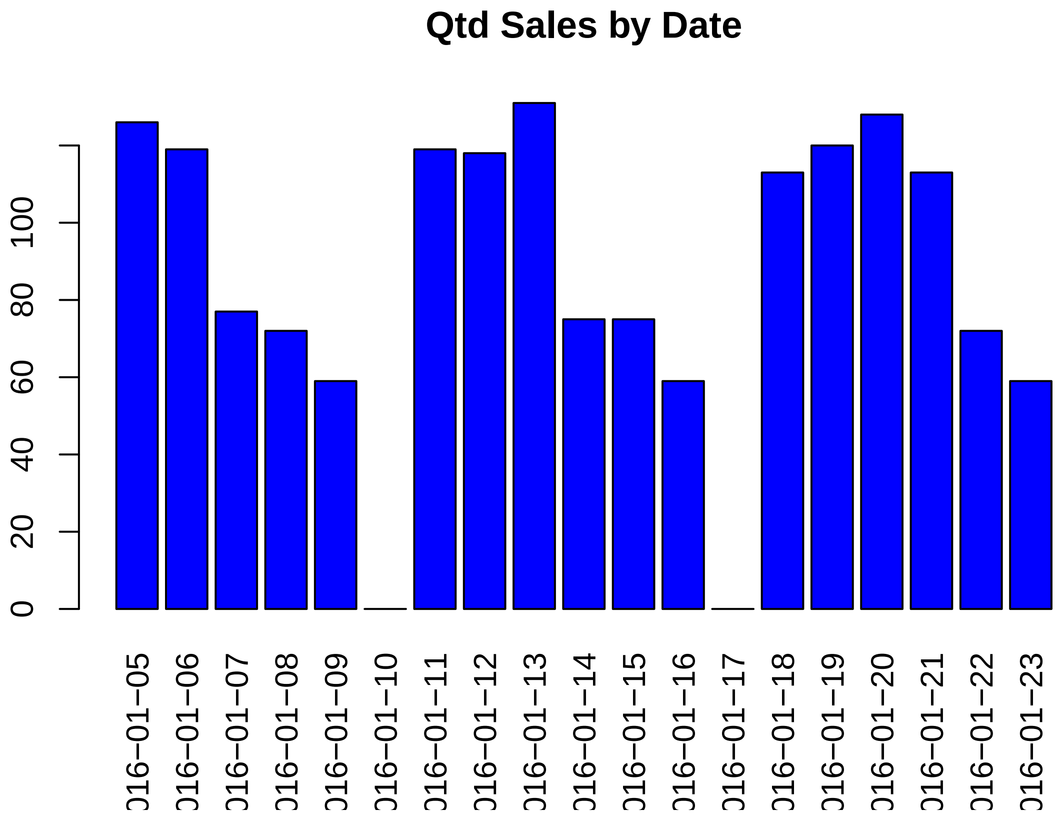
Identify a pattern that can help predict how much a customer will spend.

According the plots below, there is clearly a pattern about the quantity of sales and the day of the week.

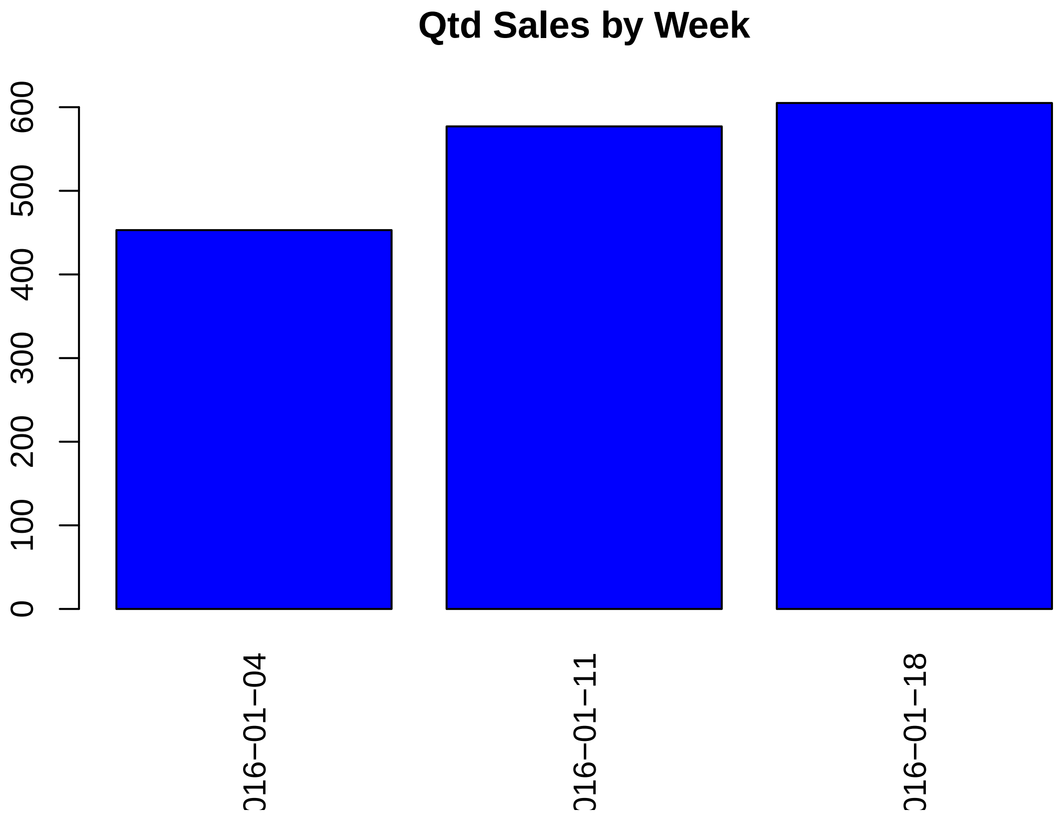
As example, you can see that in 2016-01-10 and 2016-01-17 there is no sales. Both dates are Sundays and maybe this is a behavior. Families would stay at home in Sundays.

The sample is not so big because there is only 3 weeks in this dataset.

```r
qtdSalesByDay <- strptime(as.matrix(rd$ide[,1]),format = "%Y-%m-%d")
tab <- table(cut(qtdSalesByDay, 'day'))
tab2 <- table(cut(qtdSalesByDay, 'week'))

barplot(tab, col=cm(5), las=3,main = "Qtd Sales by Date")
```



```
barplot(tab2, col=cm(5), las=3,main = "Qtd Sales by Week")
```



Another thing that we can realize is the higher probability to have customer between 12:00 and 14:00 (Lunch time) and between 20:00 and 22:00 (dinner), according to the table below.

Between 00:00 and 11:00 there is no customers.

```
qtdSalesByHour <- strptime(as.matrix(rd$ide[,1]),format = "%Y-%m-%dT%H:%M")
count(hour(qtdSalesByHour))
```

```
##     x freq
## 1  11    8
## 2  12  393
## 3  13  578
## 4  14  112
## 5  15   14
## 6  18    3
## 7  19   53
## 8  20  122
## 9  21  181
## 10 22  147
## 11 23   24
```

We can apply this rule to try foresee customer visit based on day of the week and hour.

Fitting Models

We need predict how much a customer will spend then we need a regression algorithm. There is several options of machine learning for regression and supervised learning.

Here we will try multiple linear regression and neural network.

Firstly we tried with multiple linear regression. There are 3 variables that can explain how much will be spent: - the product price (vProd)
- the quantity (it's a unity in KG or UN) (qCom)
- the product (xProd)

The product is a categorical variable. Some products have a higher probability of been consumed. BUFFET, per example, is high consumed product.

```r
#Create an unique data frame with the prods for help in linear regression and neural network
x = 0

#Create an unique data frame with the prods for help in linear regression and neural network
for (obj in rd$dets){
  if (x<1) {
    dfProd <- obj$prod
    x = 1
    } else {
    dfProd <- rbind(dfProd,obj$prod)
  }
}

#First model, using regression
#vUnCom explained by qCom,vProd and xProd variables
firstModel <- lm(vUnCom~qCom+vProd+xProd,dfProd)
summary(firstModel)
```

```
##
## Call:
## lm(formula = vUnCom ~ qCom + vProd + xProd, data = dfProd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.685  -0.494  -0.086   0.731 257.637
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           4.606368   0.284431  16.195  < 2e-16 ***
## qCom                 -0.870397   0.098114  -8.871  < 2e-16 ***
## vProd                 0.073742   0.004358  16.922  < 2e-16 ***
## xProdBACARDI          2.747833   4.788340   0.574  0.56610
## xProdBUFFET          61.173479   0.357932 170.908  < 2e-16 ***
## xProdBULE CHA         3.859343   1.708657   2.259  0.02396 *
## xProdCAFE EXPRESSO   -0.371609   1.095827  -0.339  0.73454
## xProdCAIPIRINHA       7.374977   1.824053   4.043 5.39e-05 ***
## xProdCAIPIROSKA      10.440456   1.613926   6.469 1.12e-10 ***
## xProdCERVEJA          1.759585   0.550669   3.195  0.00141 **
## xProdCERVEJA LATA     3.280344   0.642571   5.105 3.48e-07 ***
## xProdCHA            -2.603521   0.570727  -4.562 5.25e-06 ***
## xProdDOCINHOS       -0.590683   0.723432  -0.817  0.41427
## xProdHARUMAKI        2.686083   2.771846   0.969  0.33258
## xProdLIMONADA        0.557430   0.683084   0.816  0.41453
## xProdREFRIGERANTE    0.868165   0.290412   2.989  0.00281 **
## xProdSAKE           12.724990   0.837445  15.195  < 2e-16 ***
## xProdSASHIMI        16.271198   2.773470   5.867 4.86e-09 ***
## xProdSOBREMESA       6.004375   1.185770   5.064 4.32e-07 ***
## xProdSUCO            0.926605   0.423373   2.189  0.02869 *
## xProdSUSHI ESPECIAL 11.872865   0.746839  15.897  < 2e-16 ***
## xProdTEMAKI          8.953315   1.302513   6.874 7.36e-12 ***
## xProdURAMAKI        10.157896   4.788749   2.121  0.03397 *
## xProdVINHO          13.862927   4.789049   2.895  0.00382 **
## xProdWHISKY          8.953579   1.220998   7.333 2.78e-13 ***
## xProdYAKISSOBA      36.278367   2.411356  15.045  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.782 on 3521 degrees of freedom
## Multiple R-squared:  0.9779, Adjusted R-squared:  0.9778
## F-statistic:  6245 on 25 and 3521 DF,  p-value: < 2.2e-16
```

The summary of this model show some good statistics about. Example: - R Squared near 1: 0.9778
- Most of the coeficients showing *** (highly significant p-value)
- t value far from 0 (Except BACARDI, EXPRESSO, DOCINHOS, HARUMAKI AND LIMONADA)

Now, we will try with neural network

```
secondModel <- nnet(vUnCom~qCom+vProd+xProd,data=dfProd,size=2)
```

```
## # weights:  55
## initial  value 7769786.731708
## final  value 7619410.630000
## converged
```

```
summary(secondModel)
```

```
## a 25-2-1 network with 55 weights
## options were -
##    b->h1   i1->h1   i2->h1   i3->h1   i4->h1   i5->h1   i6->h1   i7->h1
##     0.53    -0.26     2.19     0.16     0.54     0.52    -0.60     0.44
##   i8->h1   i9->h1  i10->h1  i11->h1  i12->h1  i13->h1  i14->h1  i15->h1
##    -0.52     0.13     0.07    -0.07    -0.30     0.63     0.38     0.12
##  i16->h1  i17->h1  i18->h1  i19->h1  i20->h1  i21->h1  i22->h1  i23->h1
##    -0.43     0.27     0.63     0.70     0.20     0.39    -0.12     0.51
##  i24->h1  i25->h1
##    -0.41     0.41
##    b->h2   i1->h2   i2->h2   i3->h2   i4->h2   i5->h2   i6->h2   i7->h2
##    19.86    19.35   103.80     0.51     1.70     0.46    -0.34    -0.30
##   i8->h2   i9->h2  i10->h2  i11->h2  i12->h2  i13->h2  i14->h2  i15->h2
##    -0.59     0.57     1.20     1.20     0.64     0.15     0.30    10.45
##  i16->h2  i17->h2  i18->h2  i19->h2  i20->h2  i21->h2  i22->h2  i23->h2
##     0.65     0.15    -0.23     0.34     0.28     0.73     0.20     0.44
##  i24->h2  i25->h2
##     0.40     0.54
##     b->o    h1->o    h2->o
## 11295.25    70.93    36.52
```

Mean squared error

```
#Using multiple linear regression
firstModel.predict <- predict(firstModel)

# mean squared error
mean((firstModel.predict - dfProd$vUnCom)^2)
```

```
## [1] 22.69995
```

```
#Using neural network
secondModel.predict <- predict(secondModel)

# mean squared error
mean((secondModel.predict - dfProd$vUnCom)^2)
```

```
## [1] 2148.128
```

Sales forecast for the next week

We have data about 3 weeks.

Using linear regression is possible to predict the fourth week using the ammount of each week.

```
#Building the dataframe
day <- c(5,6,7,8,9,11,12,13,14,15,16,18,19,20,21,22)
values = c()
i = 1
valueRestaurant = 0
currentDate = substr(rd$ide[i,1]$`$date`,1,10)
#summarize by day

for (obj in rd$complemento){
  if(substr(rd$ide[i,1]$`$date`,1,10) != currentDate)
  {
    values <- c(values,valueRestaurant)
    valueRestaurant = 0
    currentDate = substr(rd$ide[i,1]$`$date`,1,10)

  }
  valueRestaurant = valueRestaurant + rd$complemento[i]
  i = i+1
}
```

```r
df <- data.frame(values,day)

lm.fit <- lm(values~day,df)

#Predicting the last week of January
new.df <- data.frame(day=c(25,26,27,28,29,30))

#Predict using the linear Model
#Days 25 to 30 January. January, 31 it's Sunday then we consider 0 sales
predict(lm.fit, new.df)
```

```
##        1        2        3        4        5        6
## 5689.278 5712.281 5735.284 5758.288 5781.291 5804.295
```

Conclusion

In this challenge, a dataset about customer's spent was analysed and some insights were discovered:

- On Sundays, there were no sales, since families probably stay at home or in another event
- The highest consumption was observed in Lunch and dinner times.
- The ammount spent in restaurant can be explained by Quantity, Unity price and by kind of product (BUFEET is more expensive and more consumed)
- for predicting the last week, we used the ammount spent by day and considered 0 for Sundays.