Forecast financial crisis in brasilian stock market using Naive Bayes

Marcos J Ribeiro

FEARP-USP

19/05/2020

My Machine Learning work

- I built Naive Bayes algorithm to solve classification problems
- I used R language, version 4.0, to do this
- This presentation was built using Beamer Rmarkdown
- My Naive Bayes algorithm and this presentation can be view in my Github. This presentation can also be seen in my Rpubs
- First, i will apply my algorithm in four data sets. The first two are simple
- After, i will apply my algorithm in brasilian stock market to identified crisis. This is my principal analysis
- I will try to predict the crash of brasilian stock market during COVID-19 pandemic
- There are two types of independent variables: Categorical and non-categorical
- The approach to classification in this context is diferent
- So, i built two functions to solve this, and put this two functions inside one

Naive Bayes function

- I created two functions: one to be used in datasets with categorical independent variables and the other to be used in datasets with non-categorical independent variables

```
naivef = function(k, df, cd=1){
    if(cd == 1){
      naive_marcos(k, df)
    }else if (cd == 0){
      naive_marcos2(k, df)
    }else{
      cat('Type cd = 1 for categorical dependent variables, \n
      and cd = 0 for non-categorical dependent variables.')
    }}
```

- If cd=1 the algorithm can be used in classification problems with categorical dependent variables (naive_marcos)
- If cd=0 the algorithm can be used in classification problems with non-categorical dependent variables (naive_marcos2)

Naive Bayes function

- k is the class
- df is the data frame that contains the dataset of interest
- My predict function can be see below

```
predf = function(k, df, df_n, cl, cclas=0, cd=1){
  if(cd == 1){
    pred_marcos(k, df, df_n, cl, cclas)
  }else if (cd == 0){
    pred_marcos2(k, df, df_n, cl, cclas)
  }else{
    cat('Type cd = 1 for categorical dependent variables,
    \n and cd = 0 for non-categorical dependent variables.')
  }}
```

- df_n is the new data set that we want to predict the class
- cl is the inductor
- cclas gives the class if cclas=1, and probabilities if cclas=0

My first example (default risk)

- There are three attributes in dependent variable (risco) and two independents variables. The independent variables (historia, divida) are categoricals as you can see in head table of my data set:

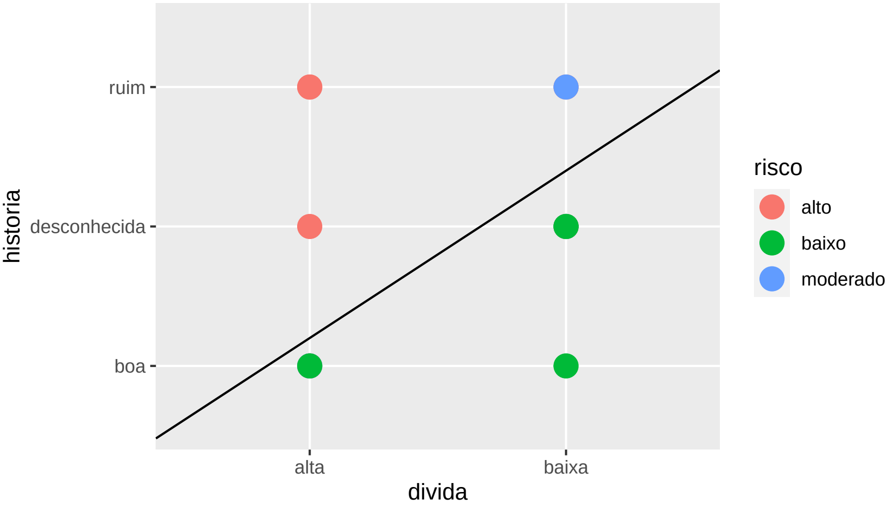
Table 1: Dataset with categorical independent variables

historia	divida	risco
ruim	alta	alto
desconhecida	alta	alto
desconhecida	baixa	moderado
desconhecida	baixa	alto
desconhecida	baixa	baixo
desconhecida	baixa	baixo

- Risco is a default risk that the bank runs when lending money
- Historia is customer credit history and divida is customer debt in market
- So, I will predict if the new customer is a good customer

Plots

- I use ggplot to plot my data set.
- Note that credit history is important to predict risk default



Inductor to categorical dependet variables

- I used naivef function in my dataset

```
cl = naivef('risco', df, cd=1)
```

```
## [1] "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-"
## [1] "Marcos Naive Bayes Classifier for Discrete Predictors"
## [1] "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-"
## A-priori probabilities:
##
##      alto     baixo  moderado
## 0.4285714 0.3571429 0.2142857
## Conditional Probabilities:
```

- This function return a table that contains conditional probabilities
- This table was save in cl object

Inductor to categorical dependet variables

- cl object is a tensor
- The tensor has a dimension equal to the number of the class attributes. In this case three
- You can see below the first dimension of this tensor
- The first row of first column is:

$$P(alta, boa|alto) = 0.04761$$

```
head(cl[, ,1])
```

```
##                     alta      baixa
## boa           0.04761905 0.02380952
## desconhecida 0.09523810 0.04761905
## ruim          0.14285714 0.07142857
```

Predict

- I have six new customers and i want to know if they are good payers
- My new data set can be see below

Table 2: New data set with categorical independent variables

historia	divida
boa	baixa
boa	alta
ruim	baixa
ruim	alta
desconhecida	baixa
desconhecida	alta

- To do this i used predf function

Predict

- Here, i used cclas = 0, so, my function return the probabilities of risk associated with my new client

```r
predf('risco', df, df_teste, cl, cclas = 0, cd=1)
```

```
##            alto     baixo  moderado
## [1,] 0.1190476 0.6428571 0.2380952
## [2,] 0.3030303 0.5454545 0.1515152
## [3,] 0.6000000 0.0000000 0.4000000
## [4,] 0.8571429 0.0000000 0.1428571
## [5,] 0.2631579 0.4736842 0.2631579
## [6,] 0.5405405 0.3243243 0.1351351
```

Predict

- Here, i used cclas = 1, so, my function return the attribute of my new client
- Recall that cd=1 is to categorical independent variables

```
predf('risco', df, df_teste, cl, cclas = 1, cd=1)
```

```
## [1] "baixo" "baixo" "alto"  "alto"  "baixo" "alto"
```

Quality control

- Here only to verify if my algorith is correct i compared to Naive Bayes produced by library e1071

```
library(e1071)
clas2 = naiveBayes(x=df[-3], y = as.factor(df$risco))
prev2 = predict(clas2, newdata = df_teste)
print(prev2)
```

```
## [1] baixo baixo alto  alto  baixo alto
## Levels: alto baixo moderado
```

- The answers of my algorithm and e1071 are identical

My second example

- I have two attributes in dependent variable (sex) and two independent variables, weight and height
- Weight and height are non-categorical as you can see in head table of my data set

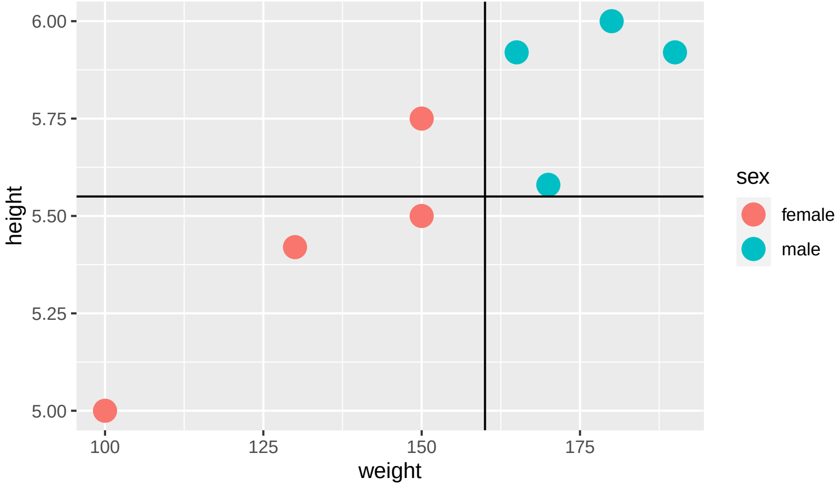
Table 3: Data set with non categorical independent variables

height	weight	sex
6.00	180	male
5.92	190	male
5.58	170	male
5.92	165	male
5.00	100	female
5.50	150	female

- Height and weight are characteristics of the individual
- And i will predict your sex based in this variables

Plots

- I use ggplot to plot my data set. Note that male is havier than female
- And on average, the man is taller



Inductor to non-categorical dependet variables

```
cl2 = naivef('sex', teste, cd=0)
```

```
## [1] "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-"
## [1] "Marcos Naive Bayes Classifier for Discrete Predictors"
## [1] "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-"
## A-priori probabilities:
##
## female   male
##    0.5    0.5
```

- This function return a table that contains conditional probabilities
- This table was save in cl2 object

Inductor to non-categorical dependet variables

- cl2 is a tensor that contains the two first moments of height and weight by sex. As you can see below
- The tensor has a dimension equal to the number of the class attributes

```
cl2
```

```
## , , female
##
##              mean   variance
## [1,]    5.4175  0.3118092
## [2,] 132.5000 23.6290781
##
## , , male
##
##             mean   variance
## [1,]    5.855  0.1871719
## [2,] 176.250 11.0867789
```

Predict

- I have four new people and i want to know if they are male or female
- My new data set can be see below

Table 4: New data set with non categorical independent variables

height	weight
5.4	170
5.8	183
6.0	188
5.0	188

- So, i used predf function to predict the attribute of people

Predict

- cclas = 1 returns the attribute of new people

```
predf('sex',teste, dfn, cl2, cclas =1, cd=0)
```

```
##      [,1]
## [1,] "female"
## [2,] "male"
## [3,] "male"
## [4,] "female"
```

Predict

- cclas = 0 returns the probabilities of the people to be male or female

```r
predf('sex',teste, dfn, cl2, cclas =0, cd=0)
```

```
##           female        male
## [1,] 0.642353175 0.357646825
## [2,] 0.016711702 0.983288298
## [3,] 0.007327301 0.992672699
## [4,] 0.997700955 0.002299045
```

Quality control

- Here only to verify if my algorith is correct i compared to Naive Bayes produced by library e1071

```
library(e1071)
clas3 = naiveBayes(x=teste[-3], y = teste$sex)
prev3 = predict(clas3, newdata = dfn, 'raw')
print(prev3)
```

```
##            female       male
## [1,] 0.642353175 0.357646825
## [2,] 0.016711702 0.983288298
## [3,] 0.007327301 0.992672699
## [4,] 0.997700955 0.002299045
```

- The answers of my algorithm and e1071 are identical

My third example

- I have two attributes in dependent variable (income) and two independent variables, occupation and education. The levels of variables can be seen on the next slide
- My independent variables are categorical
- So, I want to predict the income based in occupation and education
- My data set have 30162 observations

Table 5: Census dataset head

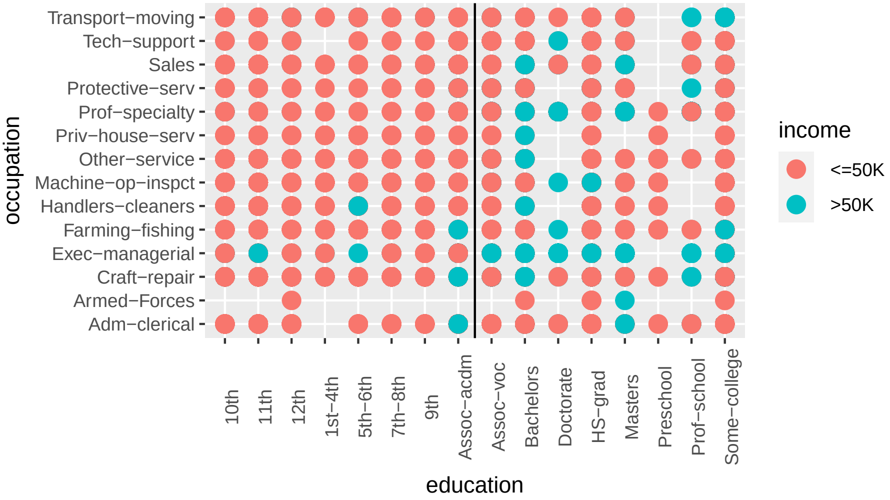
education	occupation	income
Bachelors	Adm-clerical	<=50K
Bachelors	Exec-managerial	<=50K
HS-grad	Handlers-cleaners	<=50K
11th	Handlers-cleaners	<=50K
Bachelors	Prof-specialty	<=50K
Masters	Exec-managerial	<=50K

My third example

education	occupation	income
10th	Adm-clerical	$<=$50K
11th	Armed-Forces	$>$50K
12th	Craft-repair	
1st-4th	Exec-managerial	
5th-6th	Farming-fishing	
7th-8th	Handlers-cleaners	
9th	Machine-op-inspct	
Assoc-acdm	Other-service	
Assoc-voc	Priv-house-serv	
Bachelors	Prof-specialty	
Doctorate	Protective-serv	
HS-grad	Sales	
Masters	Tech-support	
Preschool	Transport-moving	
Prof-school		
Some-college		

Plots

- How do you see in figure below, education seems to be an important factor in determining the level of income



Inductor to categorical dependet variables

- I used 28000 observations to train, and 2161 to test my algorithm

```
tr1 = census[1:28000, ]
tst1 = census[28001:30162, ]
cl4 =  naivef('income', tr1, cd=1)
```

```
## [1] "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-"
## [1] "Marcos Naive Bayes Classifier for Discrete Predictors"
## [1] "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-"
## A-priori probabilities:
##
##      <=50K      >50K
## 0.7521786 0.2478214
## Conditional Probabilities:
```

Predict

- I used cclas = 0, so, my function return the probabilities of income be >50K or <=50k

```
head(predf('income', tr1, tst1, cl4, cclas=0, cd=1))
```

```
##          <=50K      >50K
## [1,] 0.8195526 0.1804474
## [2,] 0.2398712 0.7601288
## [3,] 1.0000000 0.0000000
## [4,] 1.0000000 0.0000000
## [5,] 1.0000000 0.0000000
## [6,] 0.3256560 0.6743440
```

Predict

- Here I used cclas = 1, so, my function return the class the attribute

```
ndp = predf('income', tr1, tst1, cl4, cclas=1, cd=1)
head(ndp)
```

```
## [1] " <=50K" " >50K"  " <=50K" " <=50K" " <=50K" " >50K"
```

Predict

- The acurracy of my algorithm in this case is 75.16%

```
acurracy1 = (sum((ndp==tst1[,'income'])*1)/length(tst1[,1])) *100
acurracy1
```

```
## [1] 75.16189
```

Predict financial crisis using my algoritm

- I want predict crisis in brasilian stock market
- One of the most important models in finance is CCAPM. The complete derivation of the model can be view in my Github
- The principal equation of the model is:

$$E(R^i_{t+1}) - R^f_{t+1} = \lambda_{g_{t+1}} \beta_{i,g_{t+1}} \tag{1}$$

where

$$\beta_{i,g_{t+1}} = \left(\frac{Cov_t(g_{t+1}, R_{t+1})}{Var_t(g_{t+1})} \right) \tag{2}$$

and

$$\lambda_{g_{t+1}} = \gamma Var_t(g_{t+1}) \tag{3}$$

Predict financial crisis using my algoritm

- R^i_{t+1} is the return of asset i
- R^f_{t+1} is the risk free asset
- The left side of the equation is known as the risk premium
- g_{t+1} is the consumption growth
- t is a time subscript
- γ is the risk aversion and β the price of risk
- I will not go into the details of the model so as not to lose the focus of the work
- My claim is that i can predict crisis in brasilian stock market using risk aversion γ
- Just create a variable that represents crisis in the stock market brasilian, create a proxy for risk aversion γ, and choice other dependent variable

Crisis proxy

- To make a crisis proxy i create CMAX algorithm to detects extreme price levels, in Ibovespa returns, over a given period (12 months for example)
- CMAX equation can be see below

$$CMAX_t = \frac{p_t}{max(p_{t-12}, \cdots, p_t)} \tag{4}$$

Crisis proxy

- And my CMAX algorithm is:

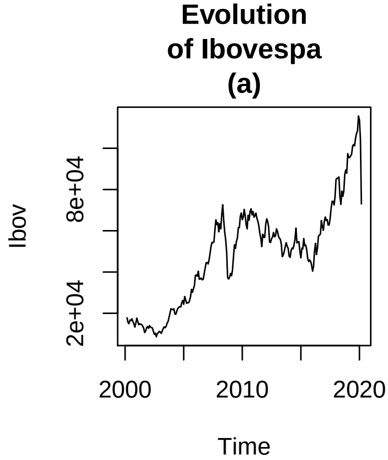
```
CMAX = function(w, n, s){
  l = matrix(nrow=n,ncol = (w+1))
  max = matrix(nrow=n, ncol = 1)
  cmax = matrix(nrow=n, ncol = 1)
  for (j in 1:n){

    l[j, 1:(w+1)] = s[j:(w+j)]
    max[j] = max(l[j, 1:(w+1)])

    cmax[j] = l[j, (w+1)]/max(max[j])
  }
  return(cmax)
}
```

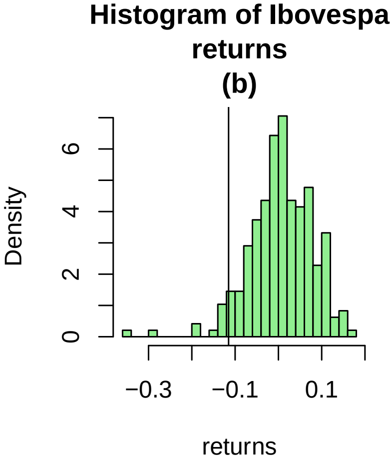
Crisis proxy

- w is the window size
- n is the number of windows
- s is the vector that i will pass the function
- If the CMAX exceeds a certain limit, we can say that it is a crisis period and the crisis proxy will be equal to 1. Otherwise, it will be zero
- This limit can be the Value at Risk in 5%

Ibovespa returns

- Note the big drop of ibov in 2020 in the figure (a) below. The vertical line in figure (b) is the limit used to define crisis. This is the quantile of 0.05 of Ibovespa returns. This approach is known as Value at Risk (Var)



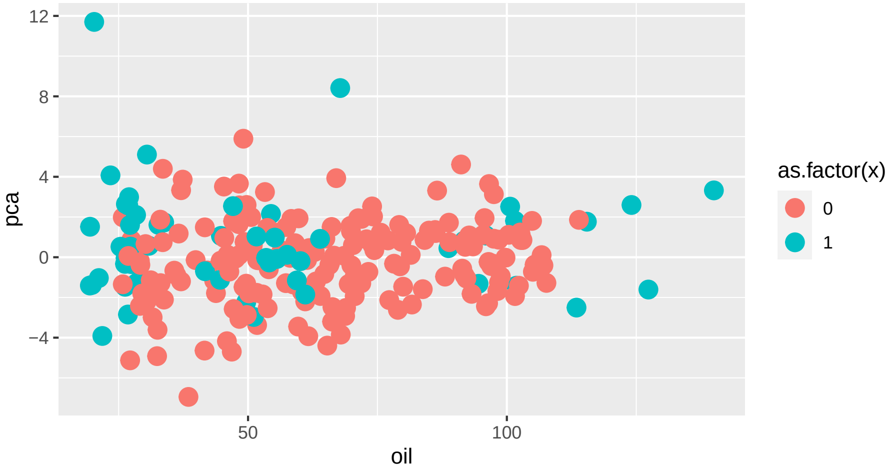


Non-categorical dependent variables

- The other two variables that i choose is PCA and oil price
- PCA was constructed using Principal Component Analysis of return of 23 assets that compose Ibovespa index. The construction of this variable can be view in my Github
- Oil price i get in Yahoo finance using quantmod library
- My data is a monthly time series from 2000-03 to 2020-03
- I use the data 2000-03 to 2019-05 to train model. And 2019-06 to 2020-03 to test model
- This last time interval cover COVID-19 pandemic. During this pandemic (2020-01 to 2020-03) the Ibovespa fell sharply

Plots

- We can see in the data that there is no obvious pattern that allows the prediction of falls in the Brazilian stock market. But it seems that the stock market falls are associated with lower oil prices



Inductor to crisis forecast in brasilian stock market

- So, i used naivef function in my dataset

```
tr = find2[1:231, ]
tst = find2[232:241, ]
cl3 = naivef('x',tr, cd=0)
```

```
## [1] "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="
## [1] "Marcos Naive Bayes Classifier for Discrete Predictors"
## [1] "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-="
## A-priori probabilities:
##
##         0         1
## 0.7965368 0.2034632
```

Predict

- I used predf funtion to predict crisis
- I used cclas = 0, so, my function return the probabilities of crisis (x=1)

```
predf('x', tr, tst, cl3, cclas=0, cd=0)
```

```
##                 0         1
##  [1,] 6.593307e-45 1.0000000
##  [2,] 2.914477e-42 1.0000000
##  [3,] 4.459197e-39 1.0000000
##  [4,] 7.096824e-38 1.0000000
##  [5,] 1.143163e-37 1.0000000
##  [6,] 1.204856e-39 1.0000000
##  [7,] 3.048610e-47 1.0000000
##  [8,] 1.431212e-34 1.0000000
##  [9,] 3.424599e-29 1.0000000
## [10,] 5.420900e-07 0.9999995
```

Predict

- Here I used cclas $= 1$, so, my function return the class the attribute

```r
predf('x', tr, tst, cl3, cclas=1, cd=0)
```

```
##       [,1]
##  [1,] "1"
##  [2,] "1"
##  [3,] "1"
##  [4,] "1"
##  [5,] "1"
##  [6,] "1"
##  [7,] "1"
##  [8,] "1"
##  [9,] "1"
## [10,] "1"
```

Predict

- The accuracy of my model is 100%

```
prev = predf('x', tr, tst, cl3, cclas=1, cd=0)
accuracy = (sum((prev == tst[,1])*1)/length(tst[,1]) )*100
accuracy
```

```
## [1] 100
```

- This accuracy may have been caused by the fact that the fall in the Brazilian stock market was very sharp

Thanks!