Modelagem - MD

Jonatha Azevedo, Leonardo Filgueira, George Amarante e Rafael

A propsota dessa modelagem é aplicar conhecimentos e técnicas em modelos lineares obtidos no curso. A modelagem será feita com um banco de dados que contém informações, entre 2010 à 2016, sobre os municípios do estado de São paulo. Após a modelagem, tentaremos criar métricas de redução de óbtos nas estradas do estado.

A modelagem será feita em duas partes:

- Análise de cluster (iremos agrupar os municípios)
- Regressão linear multipla e tratamento dos resíduos

Análise de cluster

No banco de dados, não temos nenhuma variável que seja contextualizada para nosso problema, ou seja, não temos variáveis que remetem a óbtos, acidentes em rodovias, boletins de ocorrência e etc. a ideia da clusterização é nos ajudara criar uma variável proxy que, de alguma forma, nos indique uma medida de acidentes em estradas.

Metodologia utilizada na abordagem

Como esse é um relatório, não explicitaremos a metodologia. Porém, pra mais detalhes, há o trabalho em **PDF**.

```r
#carregando pacotes necessaios
require(corrplot)
require(normtest)

set.seed(6)

city_dataset<-read.csv2('city_dataset.csv')
city_dataset <- na.omit(city_dataset)
head(city_dataset)
```

```
##       cidade  ano       pib mat1517 veiculos motos populacao pop1519
## 1 Adamantina 2010 639090.51   94.96    19331  4556     33794    2673
## 2 Adamantina 2011 683689.22   91.96    20613  4987     33811    2573
## 3 Adamantina 2012 743683.94   89.34    21879  5389     33828    2477
## 4 Adamantina 2013 815579.34   93.64    22969  5608     33845    2382
## 5 Adamantina 2014 883055.34   96.98    23966  5845     33862    2289
```

Figure 1:

```
## 8     Adolfo 2010  61059.15   73.42     1390   273      3558     268
##   pop2024 pop2529 pop60p    ibge jovem   pjovem   pmotos      pmat rodovia
## 1    2760    2492  17.56 3500105  7925 23.45091 23.56836 0.2809966      23
## 2    2734    2538  17.93 3500105  7845 23.20251 24.19347 0.2719825      23
## 3    2711    2583  18.29 3500105  7771 22.97209 24.63092 0.2641007      23
## 4    2687    2628  18.67 3500105  7697 22.74191 24.41552 0.2766731      23
## 5    2662    2670  19.04 3500105  7621 22.50605 24.38872 0.2863977      23
## 8     288     260  15.88 3500204   816 22.93423 19.64029 2.0635188       4
```

Algumas variáveis como os grupos de jovens foram transformadas, além disso, criamos a densidade de veiculos e o pib percapito pra cada município.

```
pibpercapita <-city_dataset$pib/city_dataset$populacao
dens_vei     <-city_dataset$veiculos/city_dataset$rodovia
pop1519p     <-city_dataset$pop1519/city_dataset$populacao
pop2024p     <-city_dataset$pop2024/city_dataset$populacao
pop2529p     <-city_dataset$pop2529/city_dataset$populacao



base<-data.frame(cbind(pibpercapita,pop1519p,pop2024p,pop2529p,city_dataset$pjovem,city_dataset$pop60p,c
```

Depois de incorporar as transformações no bando de dados, podemos fazer a clusterização. Chegamos a uma inferência subjetiva de que teriamos mais chances de ter um óbto em um acidente de trânsito que envolve-se moto e jovens. Essa é uma informação útil para o agrupamento.

Usaremos um método não hierárquico , o algoritmo de k-means, para a clusterização. Os métodos de agrupamentos se baseam em distâncias.

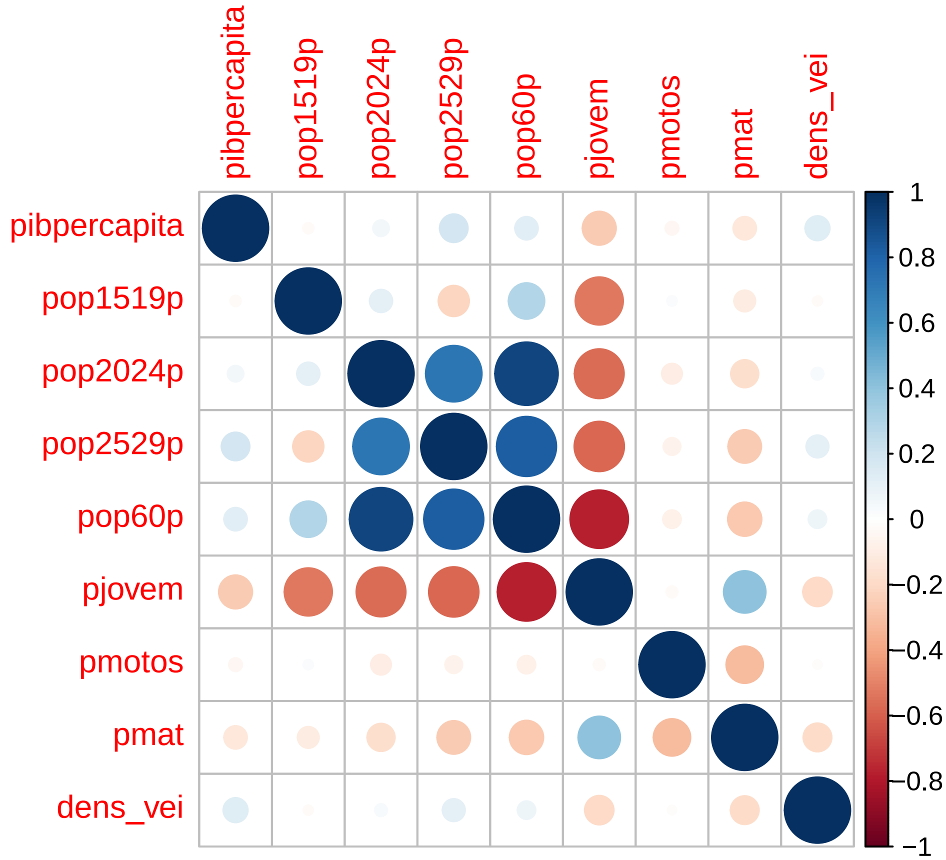
Basicamente, o modeloe de Kmeans consiste em em fazer uma escolha inicial dos k elementos que formam as sementes iniciais. Esta escolha pode ser feita da seguinte forma:

- Selecionado as k primeiras observações
- Selecionando k observações aleatorioamente; e
- Escolhendo k observações de modo que seus valores sejam bastente diferentes.

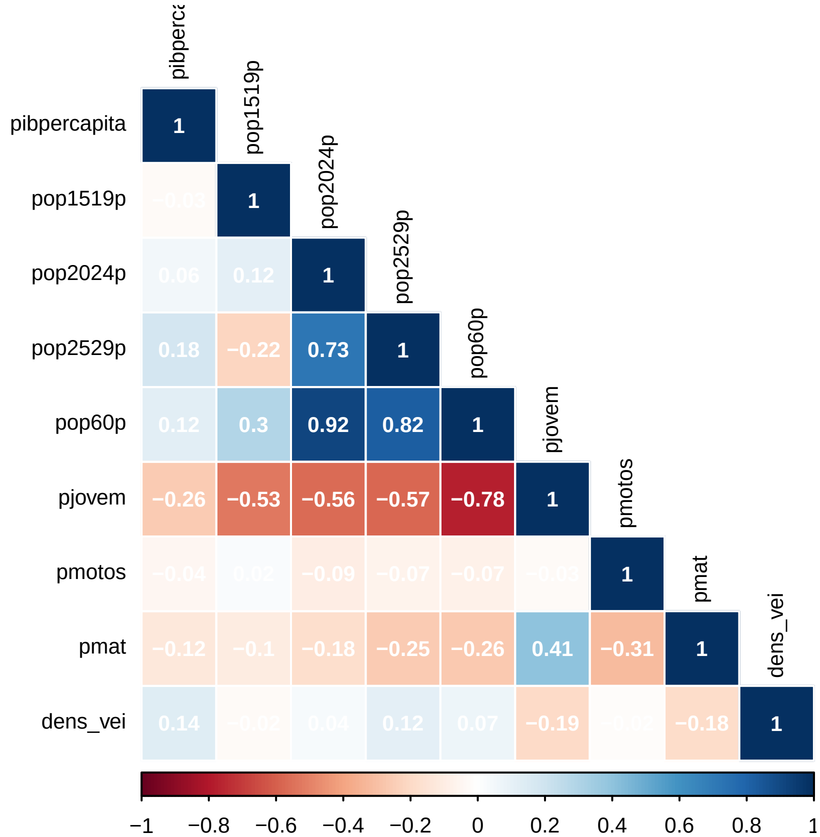
Escolhida as sementes iniciais, é calculada a distância de cada elemento em relação às sementes., agrupando o elemento ao grupo que possuir a menor distância e recalculando o centróide do mesmo. O procedimento, naturalmente, é repetido até que todos os elementos facam parte de um dos clusters.

```
names(base)[c(5:8)]<-c('pop60p','pjovem','pmotos','pmat')
omega<-cor(base,use = 'complete.obs')

corrplot(omega)
```




```r
corrplot(omega, method = "color", cl.pos = "b", type = "lower", addgrid.col = "white",
         addCoef.col = "white", tl.col = "black", tl.cex = 0.7, number.cex = 0.7, cl.cex = 0.7)
```




```
  base = cbind(base,cidade =city_dataset$cidade,ano = city_dataset$ano)

base<-data.frame(cbind(pibpercapita,pop1519p,pop2024p,pop2529p,city_dataset$pjovem,city_dataset$pop60p,
names(base)[c(5:8)]<-c('pop60p','pjovem','pmotos','pmat')
```

Criando os clusters com a função

```
kmeans_out<-kmeans(na.omit(base[,c('pjovem','pmotos','pop60p')]),centers = 4)
kmeans_out$size
```

```
## [1] 722 907 920 501
```

Introduzindo os grupos definidos anteriormente no banco de dados:

```
membros <- kmeans_out$cluster
base<-base[rowSums(is.na(base[,c('pjovem','pmotos','pmat')]))==0,]
city_dataset_cluster <- cbind(base,grupos = membros)
head(city_dataset_cluster)
```

```
##   pibpercapita   pop1519p   pop2024p   pop2529p   pop60p pjovem   pmotos
## 1     18.91136 0.07909688 0.08167130 0.07374090 23.45091  17.56 23.56836
## 2     20.22091 0.07609949 0.08086126 0.07506433 23.20251  17.93 24.19347
## 3     21.98427 0.07322337 0.08014071 0.07635686 22.97209  18.29 24.63092
## 4     24.09748 0.07037967 0.07939134 0.07764810 22.74191  18.67 24.41552
## 5     26.07806 0.06759790 0.07861319 0.07884945 22.50605  19.04 24.38872
## 6     17.16109 0.07532322 0.08094435 0.07307476 22.93423  15.88 19.64029
##        pmat  dens_vei grupos
## 1 0.2809966  840.4783      1
## 2 0.2719825  896.2174      1
```

```
## 3 0.2641007  951.2609      1
## 4 0.2766731  998.6522      1
## 5 0.2863977 1042.0000      1
## 6 2.0635188  347.5000      1
```

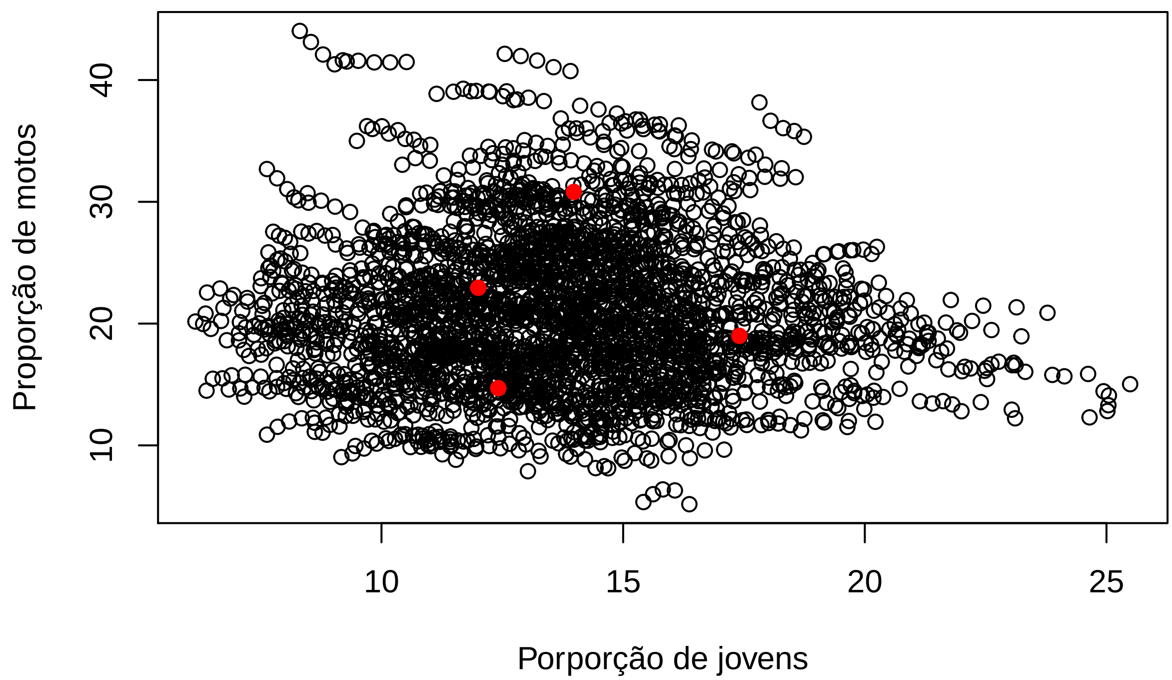
A proxy será uma média ponderada entre as variáveis que usamos para a criação dos centróides, ou seja,

$$k_{proxy} = peso_{motos}pmotos + peso_{jovens}pjovens + peso_{pop60p}pop60p$$

onde $pmotos$, $pjovens$ e $pop60p$ são, respectivamente, a proporção de motos nos municípios, a proporção de jovens e a número de pessoas com mais de 60 anos na população.

```
ind_acidente<-rowSums(kmeans_out$centers)
city_dataset_cluster<-cbind(city_dataset_cluster,NA)
names(city_dataset_cluster)[dim(city_dataset_cluster)[2]]<-'ind_acidente'

for(i in 1:10){
  city_dataset_cluster[city_dataset_cluster$membros==i,'ind_acidente']<-ind_acidente[i]
}
plot(city_dataset_cluster$pjovem,city_dataset_cluster$pmotos,xlab="Porporção de jovens", ylab="Proporçã
points(kmeans_out$centers,pch=19,col=2)
```



Agora, de fato, criremos os pesos e a variável proxy ***"proxy_id_acidentes"***.

```
pesos<-kmeans_out$centers/ind_acidente;pesos
```

```
##      pjovem    pmotos    pop60p
## 1 0.2925638 0.3193037 0.3881325
## 2 0.2329689 0.2757550 0.4912761
## 3 0.1965028 0.3755964 0.4279008
## 4 0.2012828 0.4436606 0.3550566
```

```
aux = as.data.frame(pesos)
```

```r
proxy_ind_acidente = rep(0,times = nrow(city_dataset_cluster))
dataset_final <- cbind(city_dataset_cluster,proxy_ind_acidente = proxy_ind_acidente)

for( i in 1:nrow(city_dataset_cluster)){
  if(city_dataset_cluster$grupo[i]  == 1){
    dataset_final$proxy_ind_acidente[i] = aux$pmotos[1]*dataset_final$pmotos[i] + aux$pjovem[1]*dataset_
  }else if(city_dataset_cluster$grupo[i]  == 2){
    dataset_final$proxy_ind_acidente[i] = aux$pmotos[2]*dataset_final$pmotos[i] + aux$pjovem[2]*dataset_
  }else if(city_dataset_cluster$grupo[i]  == 3){
    dataset_final$proxy_ind_acidente[i] = aux$pmotos[3]*dataset_final$pmotos[i] + aux$pjovem[3]*dataset_
  }else{
    dataset_final$proxy_ind_acidente[i] = aux$pmotos[4]*dataset_final$pmotos[i] + aux$pjovem[4]*dataset_
  }
}
```

Em estatística, uma proxy (ou variável proxy) é uma variável que não é diretamente relevante por si só, mas atua no lugar de uma variável não observável ou não mensurável para descobrir um resultado provável

Para que a variável em questão seja uma boa proxy, é preciso que haja uma forte correlação, não necessariamente uma correlação linear, com a variável que se busca analisar. Essa correlação pode ser tanto positivo quanto negativa.

Com a proxy criada, vamos criar uma modelo linear pra verificar quais variáveis explicam a nossa proxy.

```r
modelo <-  lm(formula = proxy_ind_acidente~pibpercapita+pmat+dens_vei,data = dataset_final)
summary(modelo)
```

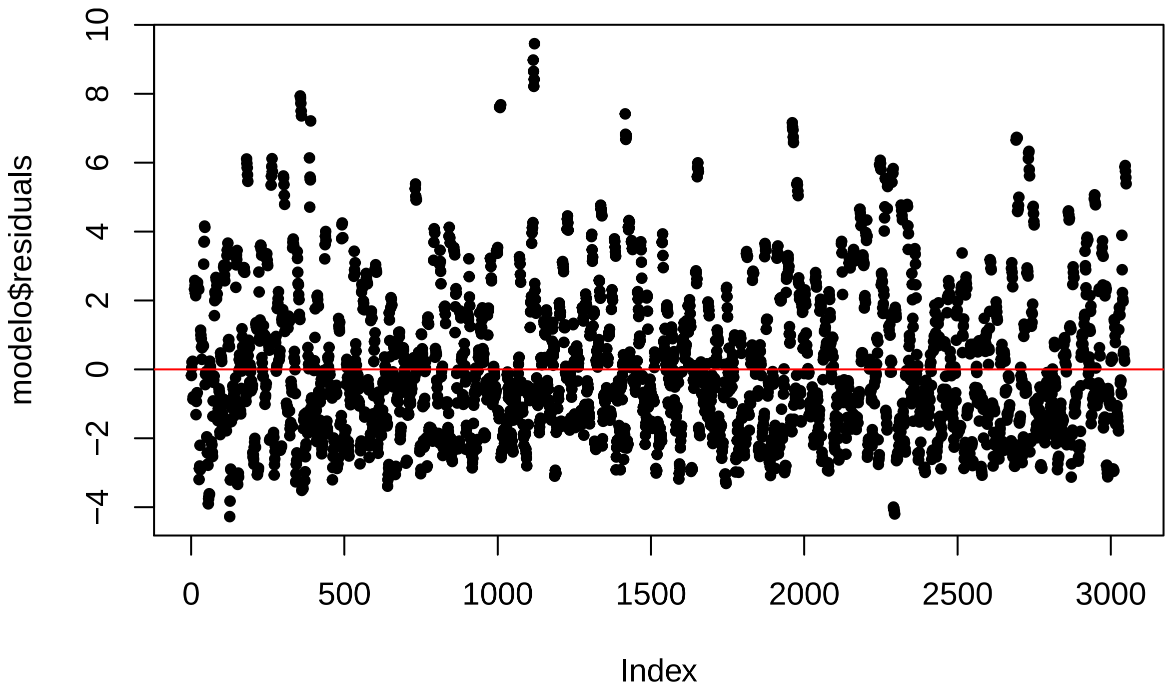
```
##
## Call:
## lm(formula = proxy_ind_acidente ~ pibpercapita + pmat + dens_vei,
##     data = dataset_final)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.274 -1.579 -0.456  1.149  9.454
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.238e+01  7.396e-02 302.610  < 2e-16 ***
## pibpercapita -9.139e-03  2.145e-03  -4.261 2.10e-05 ***
## pmat         -7.341e-01  3.773e-02 -19.455  < 2e-16 ***
## dens_vei     -7.056e-05  1.493e-05  -4.727 2.38e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.118 on 3046 degrees of freedom
## Multiple R-squared:  0.1128, Adjusted R-squared:  0.1119
## F-statistic: 129.1 on 3 and 3046 DF,  p-value: < 2.2e-16
```

Bom, temos que as variáveis pib percapto, quantidade de jovens matriculados no ensino médio e a densidade de veículos são estatisticamente significantes para a proxy. Tirando, claro, as variáveis que foram usadas para definir o processo gerador.

Qualidade do ajuste

Temos que olhar os resíduos para verificar se nosso modelo é adequado, ou pelo menos, razoavel.

```r
plot(modelo$residuals,pch=20)
abline(h=0,col=2)
```

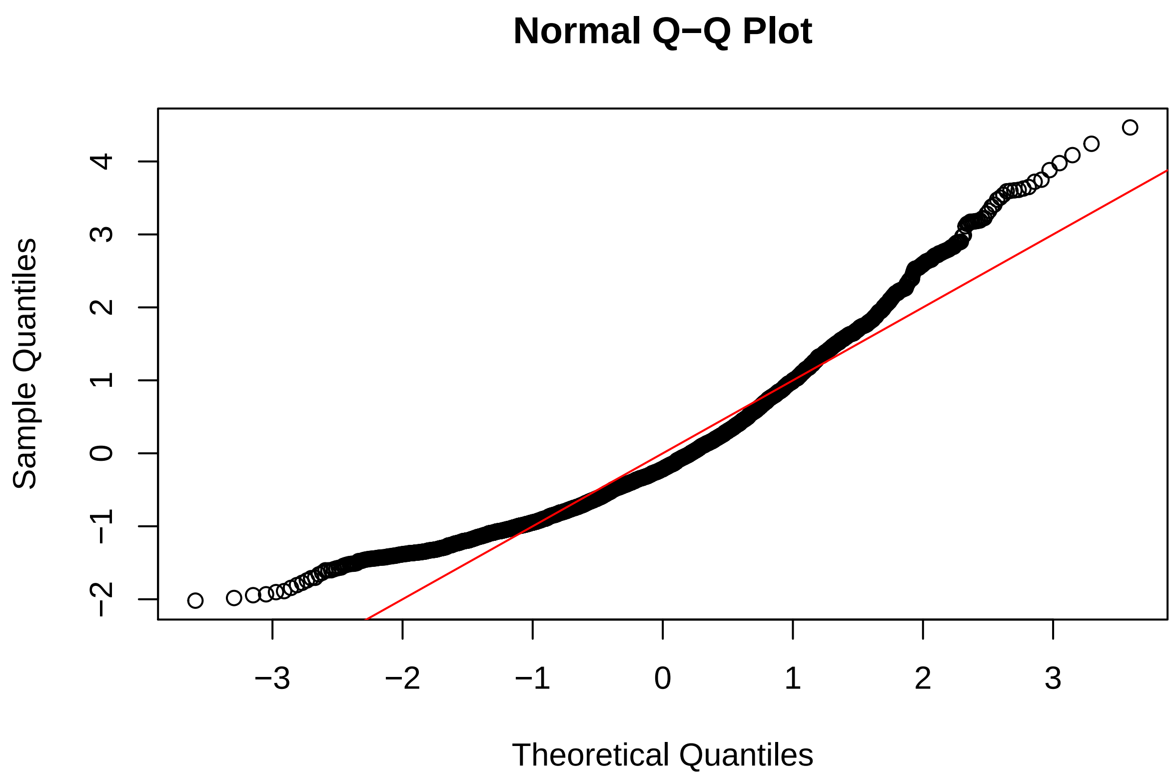


Verificando normalidade dos resíduos com o `shapiro.test()`, temos :

```r
shapiro.test(modelo$residuals)
```

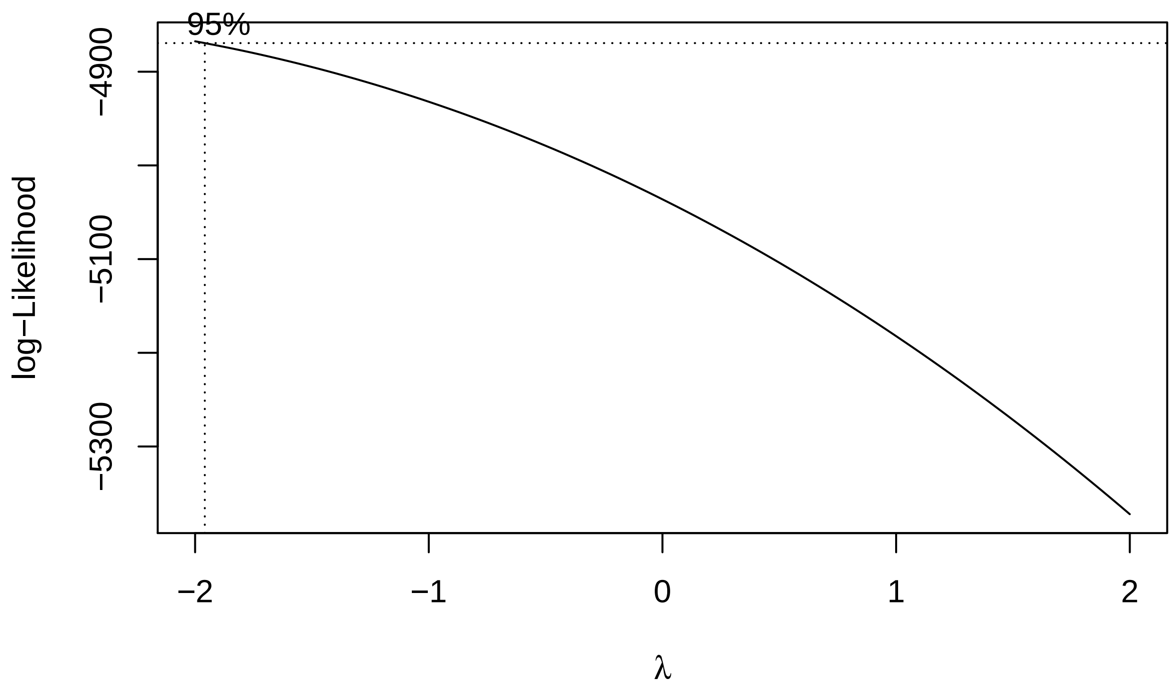
```
##
##  Shapiro-Wilk normality test
##
## data:  modelo$residuals
## W = 0.93749, p-value < 2.2e-16
```

```r
normalizando = (modelo$residuals - mean(modelo$residuals))/sd(modelo$residuals)
qqnorm(normalizando)
abline(c(0,1),col=2)
```



Ou seja, pelo p-value < 2.2e-16 rejeitamos a hipótese de que os resíduos seguem uma distribuição normal. Precisamos melhorar isso. Utilizando a função `coxbox` ela nos ajuda a obter uma transformação dos dados. Tranformação essa que pode nos ajudar a obter algumas propriedades nos resíduos.

```
require(MASS)
a = MASS::boxcox(modelo)
```




```
lambda <- a$x[which.max(a$y)];lambda
```

```
## [1] -2
```

Iremos usar o varlor de x que maximiza y com a formula proposta por Box e Cox, da seguinte forma:

$$y(\lambda) = \begin{cases} \frac{(Y+\lambda_1)^{\lambda_1}-1}{\lambda_1}, & \text{se} \quad \lambda \neq 0; \\ log(y+\lambda_2), & \text{se} \quad \lambda = 0 \end{cases}$$

Com isso , temos:

```r
dataset_t = as.data.frame(matrix(rep(0,times = length(dataset_final)),nrow = nrow(dataset_final),ncol=nc
names(dataset_t) = names(dataset_final)
for(i in 1:nrow(dataset_final)){
  for(j in 1:ncol(dataset_final)){
    aux = (((dataset_final[i,j] +lambda)^lambda)-1)/lambda
    dataset_t[i,j] = aux
  }
}

# dataset_t = log(dataset_final)
head(dataset_t)
```

```
##   pibpercapita  pop1519p  pop2024p  pop2529p    pop60p    pjovem    pmotos
## 1    0.4982517 0.3644938 0.3641299 0.3652463 0.4989134 0.4979349 0.4989252
## 2    0.4984940 0.3649157 0.3642446 0.3650610 0.4988878 0.4980297 0.4989849
## 3    0.4987480 0.3653187 0.3643464 0.3648796 0.4988632 0.4981158 0.4990237
## 4    0.4989760 0.3657154 0.3644523 0.3646980 0.4988378 0.4982007 0.4990049
## 5    0.4991376 0.3661017 0.3645620 0.3645287 0.4988109 0.4982780 0.4990025
## 6    0.4978247 0.3650247 0.3642328 0.3653395 0.4988591 0.4974047 0.4983932
##           pmat  dens_vei grupos ind_acidente proxy_ind_acidente
## 1    0.3307937 0.4999993      0           NA          0.4987201
## 2    0.3325544 0.4999994      0           NA          0.4987470
## 3    0.3340715 0.4999994      0           NA          0.4987663
## 4    0.3316417 0.4999995      0           NA          0.4987606
## 5    0.3297254 0.4999995      0           NA          0.4987616
## 6 -123.4267359 0.4999958      0           NA          0.4984252
```

Logo, aplicamos novamente a regressão linear multipla:

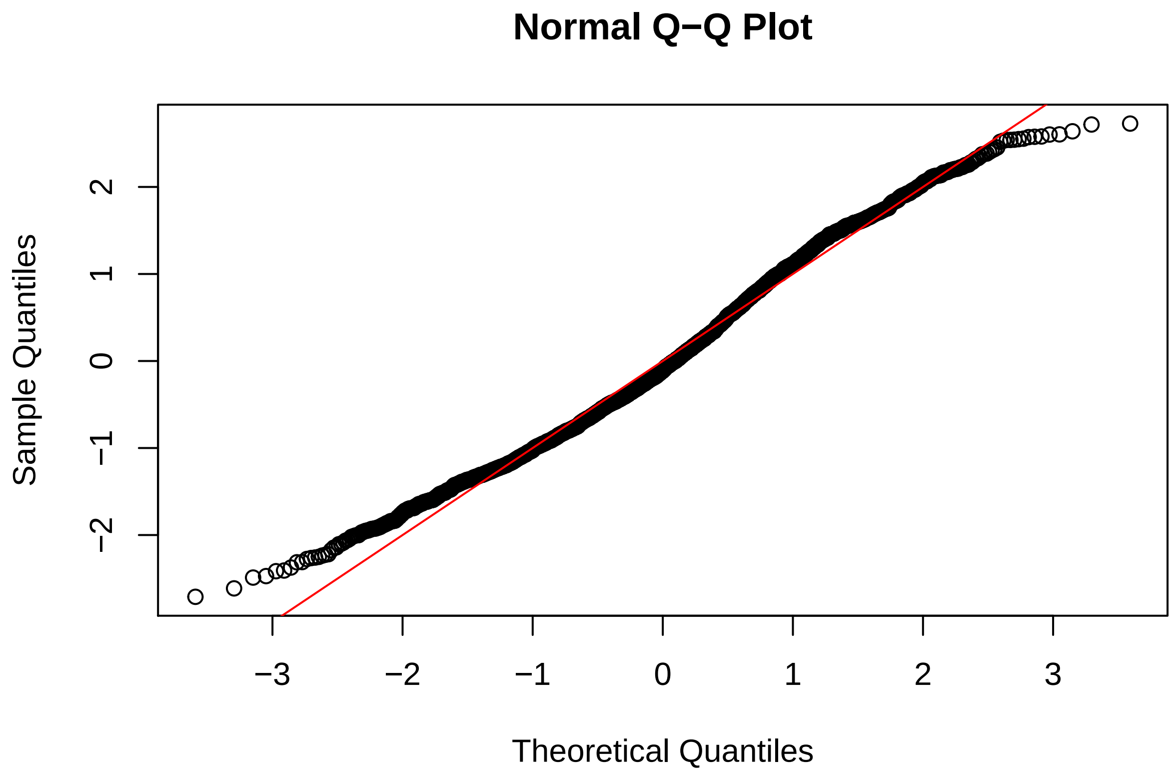
```r
modelo2 <-  lm(formula = proxy_ind_acidente~pibpercapita+pmat+dens_vei,data = dataset_t)
summary(modelo2)
```

```
##
## Call:
## lm(formula = proxy_ind_acidente ~ pibpercapita + pmat + dens_vei,
##     data = dataset_t)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -7.576e-04 -2.134e-04 -2.989e-05  2.052e-04  7.627e-04
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.564e-01  1.034e-02  44.157  < 2e-16 ***
## pibpercapita -5.357e-03  1.270e-03  -4.218 2.53e-05 ***
## pmat         -1.561e-10  9.130e-10  -0.171    0.864
## dens_vei      8.981e-02  2.099e-02   4.279 1.93e-05 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0002797 on 3046 degrees of freedom
## Multiple R-squared:  0.009286,   Adjusted R-squared:  0.00831
## F-statistic: 9.517 on 3 and 3046 DF,  p-value: 2.962e-06
```

Plotando os resíduos com os dados transformados, temos

```
normalizando2 = (modelo2$residuals - mean(modelo2$residuals))/sd(modelo2$residuals)
qqnorm(normalizando2)
abline(c(0,1),col=2)
```



Verificando a transformação com log.

```
dataset_t_log = log(dataset_final)
```

Logo, aplicamos novamente a regressão linear multipla:

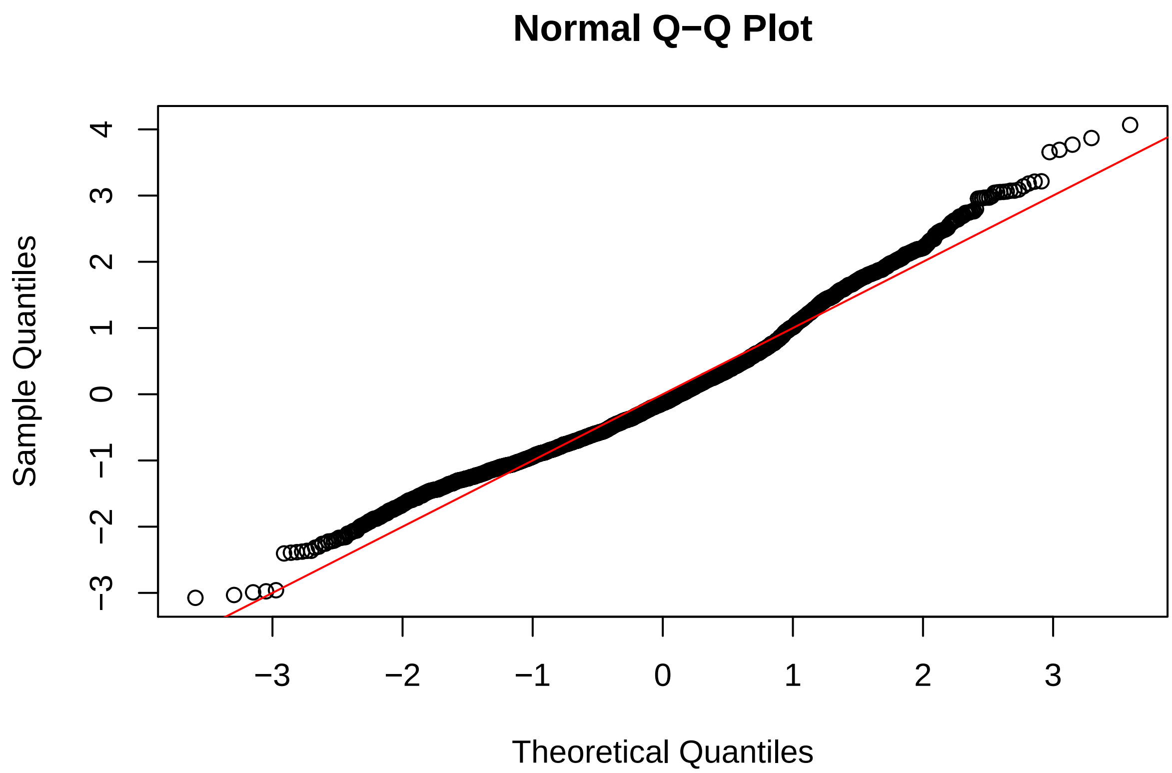
```
modelo3 <-  lm(formula = proxy_ind_acidente~pibpercapita+pmat+dens_vei,data = dataset_t_log)
summary(modelo3)
```

```
##
## Call:
## lm(formula = proxy_ind_acidente ~ pibpercapita + pmat + dens_vei,
##     data = dataset_t_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28683 -0.06592 -0.01233  0.05206  0.37946
##
## Coefficients:
```

```
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.192355   0.013468 237.030  < 2e-16 ***
## pibpercapita -0.029877   0.003375  -8.852  < 2e-16 ***
## pmat         -0.037485   0.002016 -18.598  < 2e-16 ***
## dens_vei     -0.013169   0.002224  -5.921 3.56e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09334 on 3046 degrees of freedom
## Multiple R-squared:   0.14,  Adjusted R-squared:  0.1392
## F-statistic: 165.3 on 3 and 3046 DF,  p-value: < 2.2e-16
```

Plotando os resíduos com os dados transformados, temos

```
normalizando3 = (modelo3$residuals - mean(modelo3$residuals))/sd(modelo3$residuals)
qqnorm(normalizando3)
abline(c(0,1),col=2)
```



É notável que a primeira transformação é melhor se olharmos a calda superior do qqplot dos resíduos, mas ela não da conta da outra "calda". Podemos aplicar novamente o teste de transformação, boxcox, e verificar se conseguimos ajeitar esse problema.

. . . .