

# Trabalho Modelagem Estatística 1

Importando Libs para o trabalho prático

```
library(zoo)
```

```
##  
## Anexando pacote: 'zoo'
```

```
## Os seguintes objetos são mascarados por 'package:base':  
##  
##   as.Date, as.Date.numeric
```

```
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':  
##   method      from  
##   as.zoo.data.frame zoo
```

```
library(data.table)
```

```
##  
## Anexando pacote: 'data.table'
```

```
## Os seguintes objetos são mascarados por 'package:zoo':  
##  
##   yearmon, yearqtr
```

```
library(ggplot2)
library(forecast)
library(gridExtra)
```

```
## Warning: pacote 'gridExtra' foi compilado no R versão 4.4.2
```

```
library(reshape2)
```

```
## Warning: pacote 'reshape2' foi compilado no R versão 4.4.2
```

```
##
## Anexando pacote: 'reshape2'
```

```
## Os seguintes objetos são mascarados por 'package:data.table':
##
##      dcast, melt
```

```
library(glmnet)
```

```
## Warning: pacote 'glmnet' foi compilado no R versão 4.4.2
```

```
## Carregando pacotes exigidos: Matrix
```

```
## Warning: pacote 'Matrix' foi compilado no R versão 4.4.2
```

```
## Loaded glmnet 4.1-8
```

Lendo Arquivo csv com a lib 'data.table'

```
dataset = fread("dataset/day.csv", sep = ',')
```

```
head(x = dataset, n=4)
```

instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp
<int>	<chr>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<dbl>
1	01-01-2018	1	0	1	0	6	0	2	14.110847
2	02-01-2018	1	0	1	0	0	0	2	14.902598
3	03-01-2018	1	0	1	0	1	1	1	8.050924
4	04-01-2018	1	0	1	0	2	1	1	8.200000

4 rows | 1-10 of 16 columns

Convertendo o atributo dteday para tipo date no R

```
dataset[, dteday := as.Date(dteday,format = '%d-%m-%Y')]
```

# Analise de dados

Verificando a quantidade de instancias e colunas do conj. de dados

```
data.frame(  
  val = c(  
    numero.de.colunas      = length(names(dataset)),  
    quantidade.de.instancias = length(dataset$dteday)  
  )  
)
```

	val
	<int>
numero.de.colunas	16
quantidade.de.instancias	730

2 rows

Verificando o tipo de cada atributo do conj. de dados, além disso, verificando se possui algum tipo de valor nulo.

```
# Fazendo a verificação
result_ = dataset[ ,
  lapply(.SD,
    function(x){
      return(c(class(x), any(is.null(x))))
    }
  )
]
# Resultado transposto
df = transpose(result_, keep.names = 'col')

# Modificando os nomes de cada coluna
setnames(
  x = df,
  new = c('Atributos', 'Tipo do atributo', 'Possui valores nulos?')
)
```

df

Atributos <chr>	Tipo do atributo <chr>	Possui valores nulos? <chr>
instant	integer	FALSE
dteday	Date	FALSE
season	integer	FALSE
yr	integer	FALSE
mnth	integer	FALSE
holiday	integer	FALSE
weekday	integer	FALSE
workingday	integer	FALSE

Atributos <chr>	Tipo do atributo <chr>	Possui valores nulos? <chr>
weathersit	integer	FALSE
temp	numeric	FALSE

1-10 of 16 rows

Previous 1 2 Next

Maior parte dos atributos são do tipo numéricos e inteiros, também, não possui valores nulos.

Pegando as colunas numéricas, date, etc.

```
numeros = df[ (`Tipo do atributo` == 'numeric') | (`Tipo do atributo` == 'integer') ]$Atributos
numeros
```

```
## [1] "instant"    "season"     "yr"         "mnth"       "holiday"
## [6] "weekday"    "workingday" "weathersit"  "temp"       "atemp"
## [11] "hum"        "windspeed" "casual"     "registered" "cnt"
```

Verificando a quantidade de valores únicos de cada atributo.

```
# Resultado é transposto para melhor visualização
transpose(dataset[,
  lapply( .SD,
    function(x){
      return(c(uniqueN(x), class(x)))
    }
  )
], keep.names = "colunas")
```

colunas <chr>	V1 <chr>	V2 <chr>
instant	730	integer

colunas <chr>	V1 <chr>	V2 <chr>
dteday	730	Date
season	4	integer
yr	2	integer
mnth	12	integer
holiday	2	integer
weekday	7	integer
workingday	2	integer
weathersit	3	integer
temp	498	numeric
1-10 of 16 rows		Previous 1 2 Next

- Season , holiday , weekday , workingday , weathersit são atributos catégoricos. e estão no tipo interger será passado para `as.factor` tanto para analise quanto para treinamento do modelo
- dteday é atributo do tipo data e possui valores únicos com mesma quantidade de instancias.
- instant é um chave primária pois os valores são únicos com mesma quantidade de instancias.
- Demais atributos são númericos.

**OBSERVAÇÃO** : como instant é único não será utilizado no modelo e nem será utilizada para demais análises estatísticas

Passando os dados categoricos para `as.factor`

```
dataset[, season :=as.factor(season)]
dataset[, holiday:=as.factor(holiday)]
dataset[, weekday:=as.factor(weekday)]
dataset[, workingday:=as.factor(workingday)]
dataset[, weathersit:=as.factor(weathersit)]
```

Verificando média, variância, desvio padrão de cada atributo do dataset.

```
describe = dataset[,
  lapply( .SD,
    function(x) return(
      list( as.numeric(mean(x)), max(x), min(x), sd(x), var(x))
    )
  ), .SDcols = names(dataset)[10:16]]

dt = transpose(describe, keep.names = c("colunas"),list.cols = TRUE)
setnames(
  dt, c("colunas", "média", 'max', "min", 'std', 'var')
)
```

```
dt[, lapply(.SD,
  function(x){
    if (class(x[1]) == 'character') return(x)
    return(round(as.numeric(x), 2))
  })]
```

colunas <chr>	média <dbl>	max <dbl>	min <dbl>	std <dbl>	var <dbl>
temp	20.32	35.33	2.42	7.51	56.35
atemp	23.73	42.04	3.95	8.15	66.43
hum	62.77	97.25	0.00	14.24	202.71
windspeed	12.76	34.00	1.50	5.20	27.00

<b>colunas</b> <chr>	<b>média</b> <dbl>	<b>max</b> <dbl>	<b>min</b> <dbl>	<b>std</b> <dbl>	<b>var</b> <dbl>
casual	849.25	3410.00	2.00	686.48	471254.62
registered	3658.76	6946.00	20.00	1559.76	2432847.29
cnt	4508.01	8714.00	22.00	1936.01	3748141.10
7 rows					

849.25+3658.76

## [1] 4508.01

A média de *cnt* é a soma das médias de *casual* com *registered* é um fato interessante, mais há frente haverá uma análise dessas duas colunas com *cnt*

Verificando a densidade dos dados dos atributos numéricos

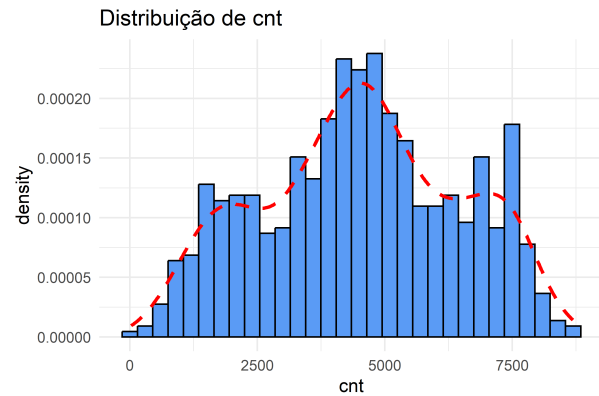
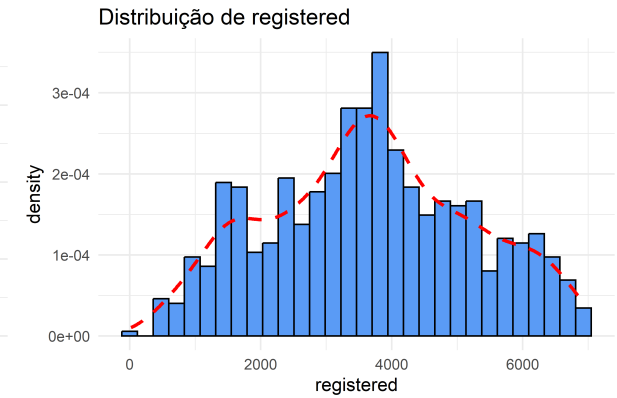
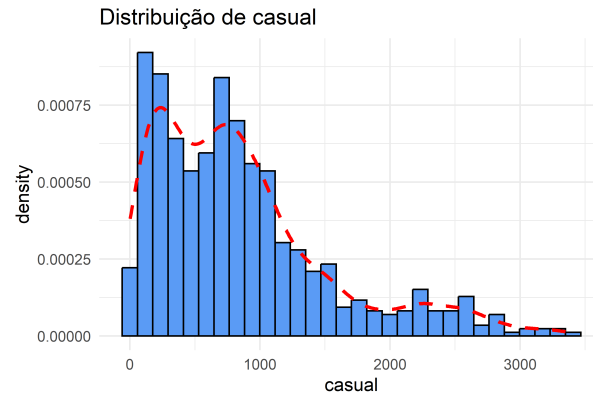
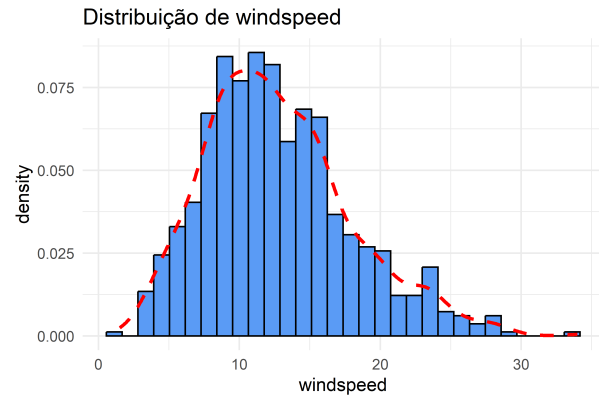
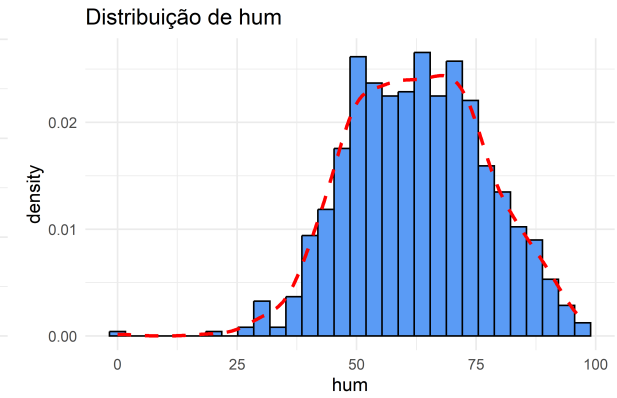
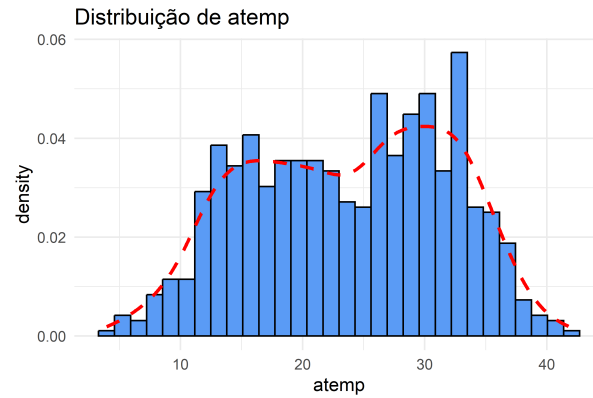
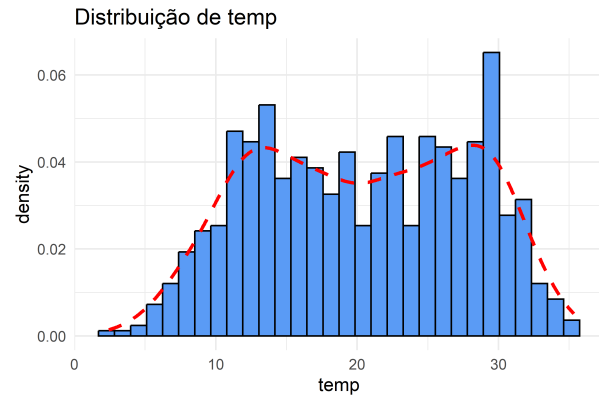
```
plots = lapply(numeros[9:15],
  function(col) {
    ggplot(dataset, aes_string(x = col)) +
      geom_histogram(aes(y = ..density..), bins = 30, fill = "#5c9ef6", color = "black") +
      geom_density(color = "#FF0000", linewidth = 1, linetype = "dashed", adjust = 1) +
      ggtitle(paste("Distribuição de", col)) +
      theme_minimal()
  })
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```
# Exibindo o primeiro gráfico como exemplo  
combined_plot <- do.call(grid.arrange, c(plots, ncol = 3))
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.  
## i Please use `after_stat(density)` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was  
## generated.
```



Fazendo um teste de hipotese para verificar normalidade dos atributos

$H_0$  : Os dados seguem uma distribuição normal.

$H_1$  : Os dados não seguem uma distribuição normal

```
print("Para um nivel de significancia de 0.05")
```

```
## [1] "Para um nivel de significancia de 0.05"
```

```
lapply(numeros[9:15],  
      function(x){  
        value = shapiro.test(dataset[[x]])$p.value  
  
        if ( value > 0.05) return(c("Aceito H0",x))  
        else return(c("Rejeito H0 para ",x))  
      })
```

```
## [[1]]  
## [1] "Rejeito H0 para " "temp"  
##  
## [[2]]  
## [1] "Rejeito H0 para " "atemp"  
##  
## [[3]]  
## [1] "Rejeito H0 para " "hum"  
##  
## [[4]]  
## [1] "Rejeito H0 para " "windspeed"  
##  
## [[5]]  
## [1] "Rejeito H0 para " "casual"  
##  
## [[6]]  
## [1] "Rejeito H0 para " "registered"  
##  
## [[7]]  
## [1] "Rejeito H0 para " "cnt"
```

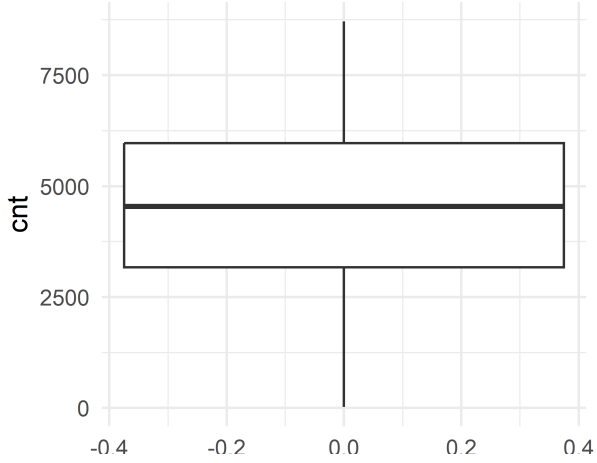
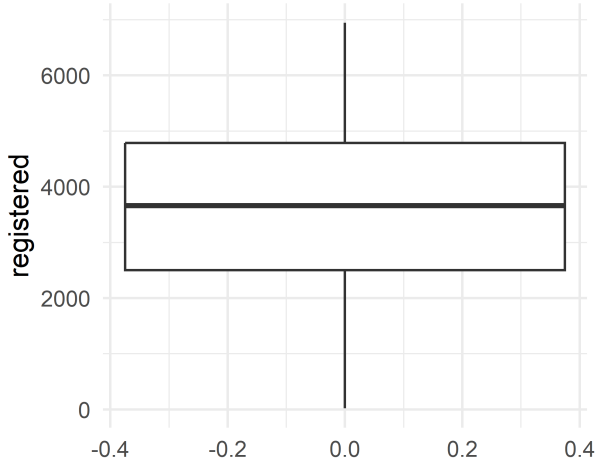
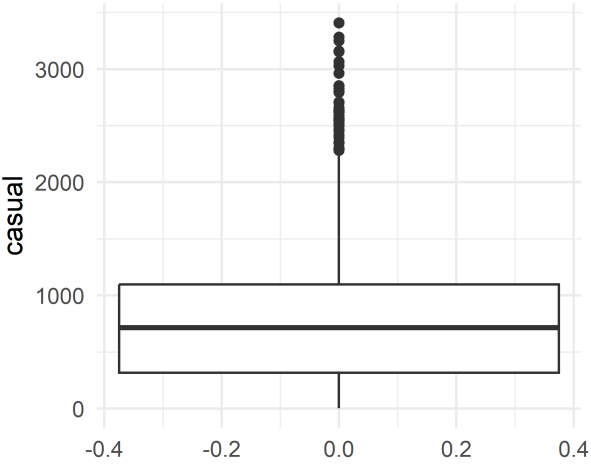
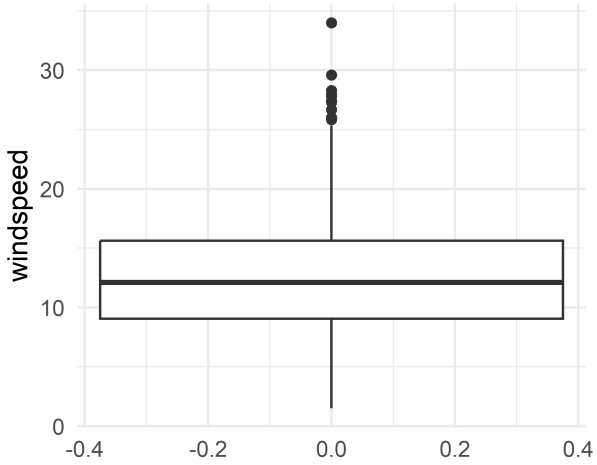
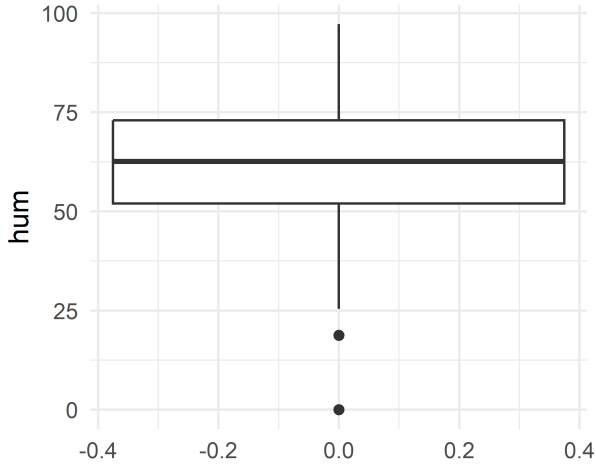
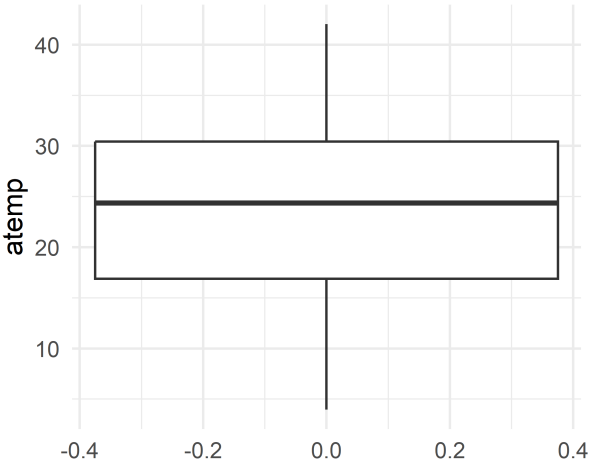
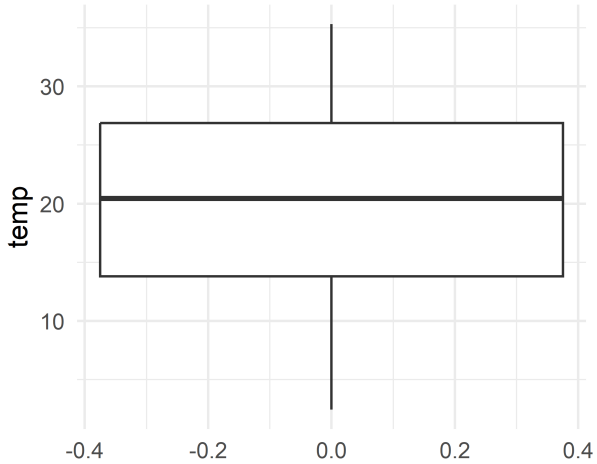
Para um nível de significancia de 0.05 para todos os atributos acima eu rejeito  $H_0$

Verificando se há existencia de outilers no conj. dados

```
numeros[9:15]
```

```
## [1] "temp"      "atemp"     "hum"       "windspeed" "casual"  
## [6] "registered" "cnt"
```

```
plots = lapply(  
  numeros[9:15],  
  function(col)  
  {  
    ggplot(dataset, aes_string(y = col)) +  
      geom_boxplot() +  
      theme_minimal()  
  })  
combined_plot <- do.call(grid.arrange, c(plots, ncol = 3))
```



Os atributos **windspeed**, **casual** e **hum** possuem outliers. Não serão removidos esses outliers pois posa ser que essas colunas sejam removidas futuramente. Se não forem será mantido mesmo assim os outliers.

## Verificando a correlação entre target e as features

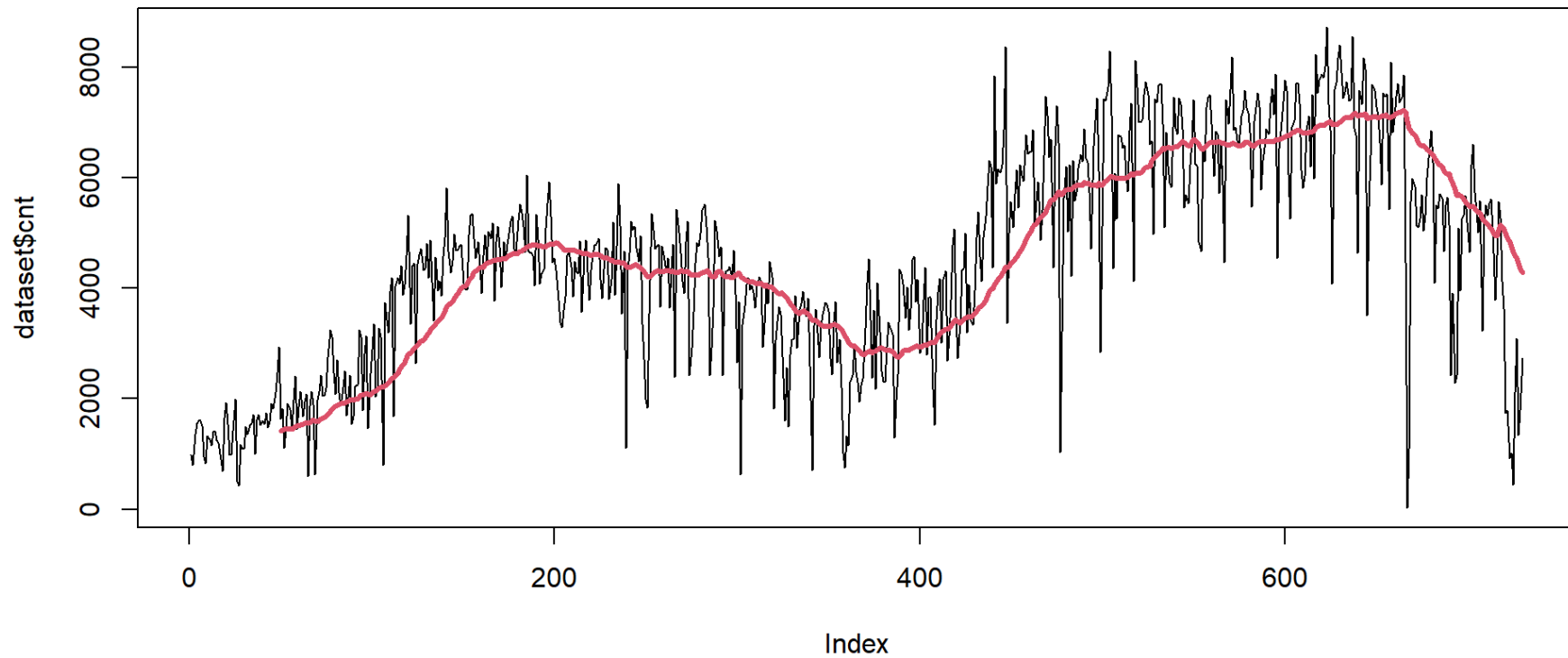
Modificando o atributo *dteday* para o formato ano-mes para fazer posteriormente um agrupamento de alugueis totais de cada mes.

```
dataset[, anoMes := format.Date(dataset$dteday, format = "%Y-%m")]
```

## Verificando a série historica dos alugueis

```
X = rollapply(dataset$cnt, width = 50, FUN = mean, align = "right", fill = NA)

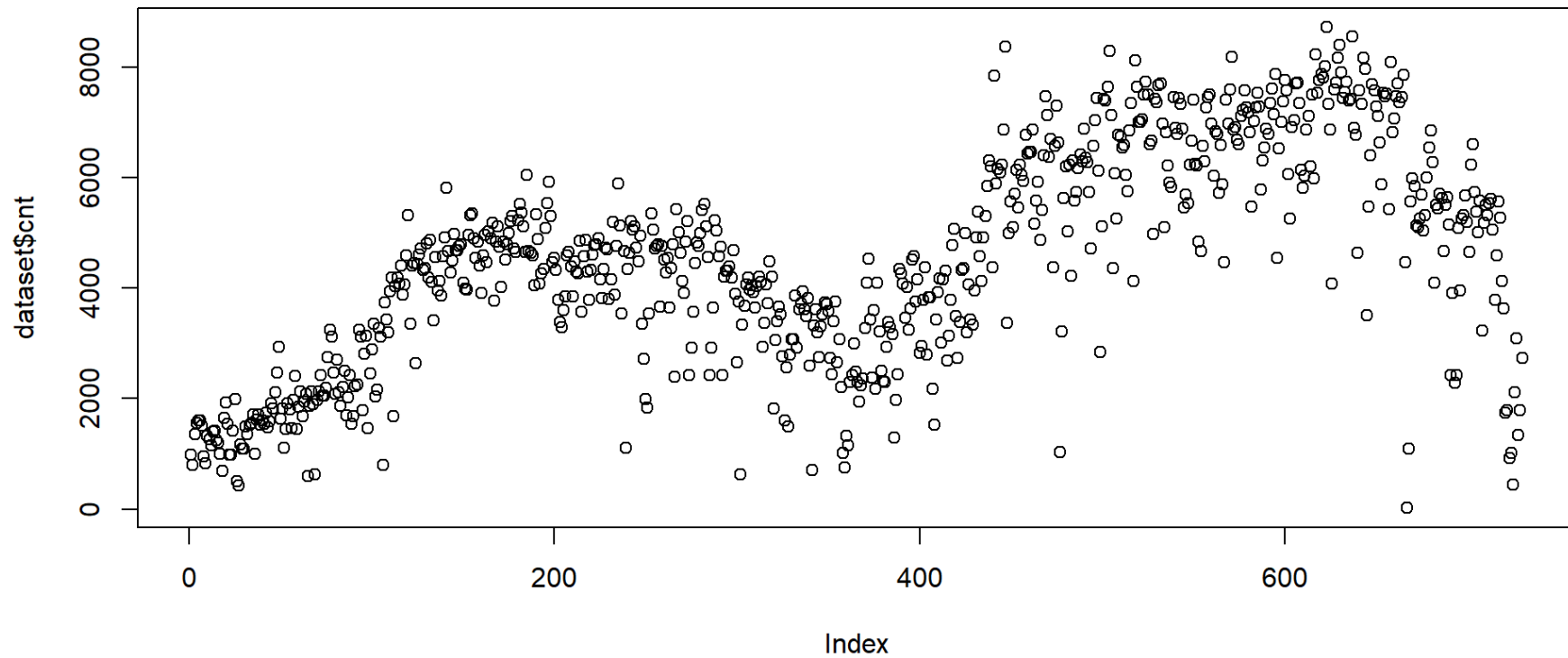
plot(dataset$cnt, type = 'l', col=1, lwd=1)
lines(X, col=2, lwd = 3)
```



Não podemos ver tendencias nessa serie ou sazonalidades

```
X = rollapply(dataset$cnt, width = 50, FUN = mean, align = "right", fill = NA)

plot(dataset$cnt, type = 'p', col=1, lwd=1, lty = 3)
```

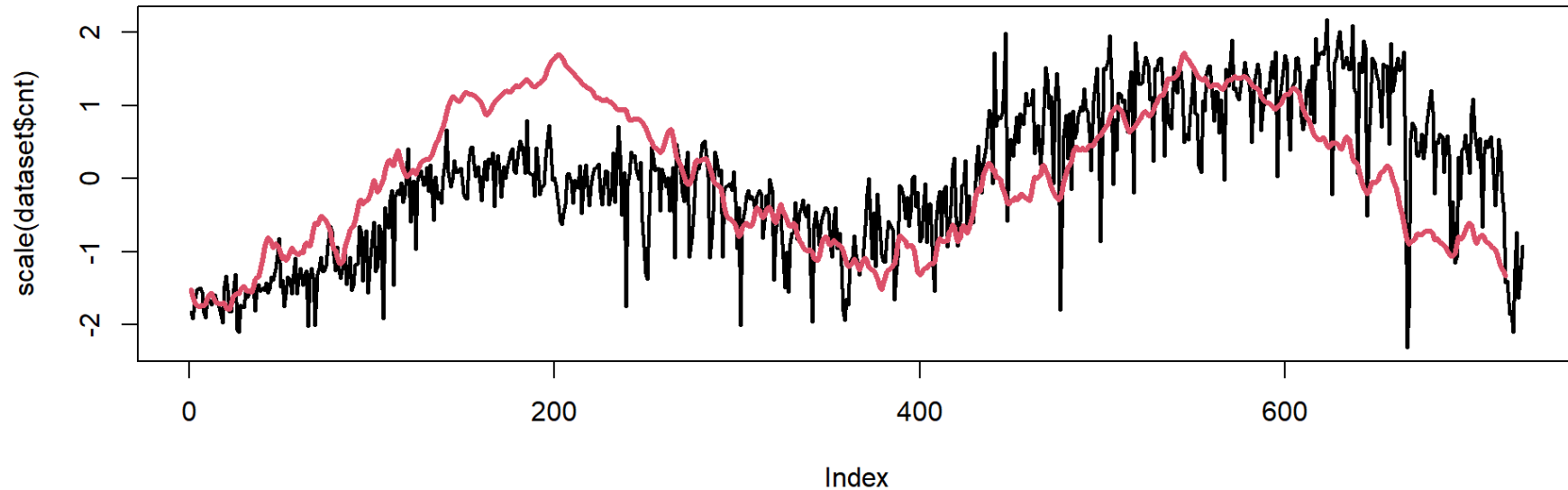


Podemos ver que o atributo `cnt` no decorrer do tempo não possui uma tendência ou sazonalidade

Correlação entre alugueis e temperatura média em relação aos 10 últimos pontos

```
roll = rollapply(scale(dataset$temp), FUN=mean, width=10, align='right')  
plot(scale(dataset$cnt), type='l', lwd=2)  
lines(roll, col=2, lwd=3, lty = 1)
```





Esse método só permite identificar visualmente se existe um comportamento semelhante entre as variáveis ao longo do tempo. Contudo, podemos ver uma pequena relação entre essas variáveis

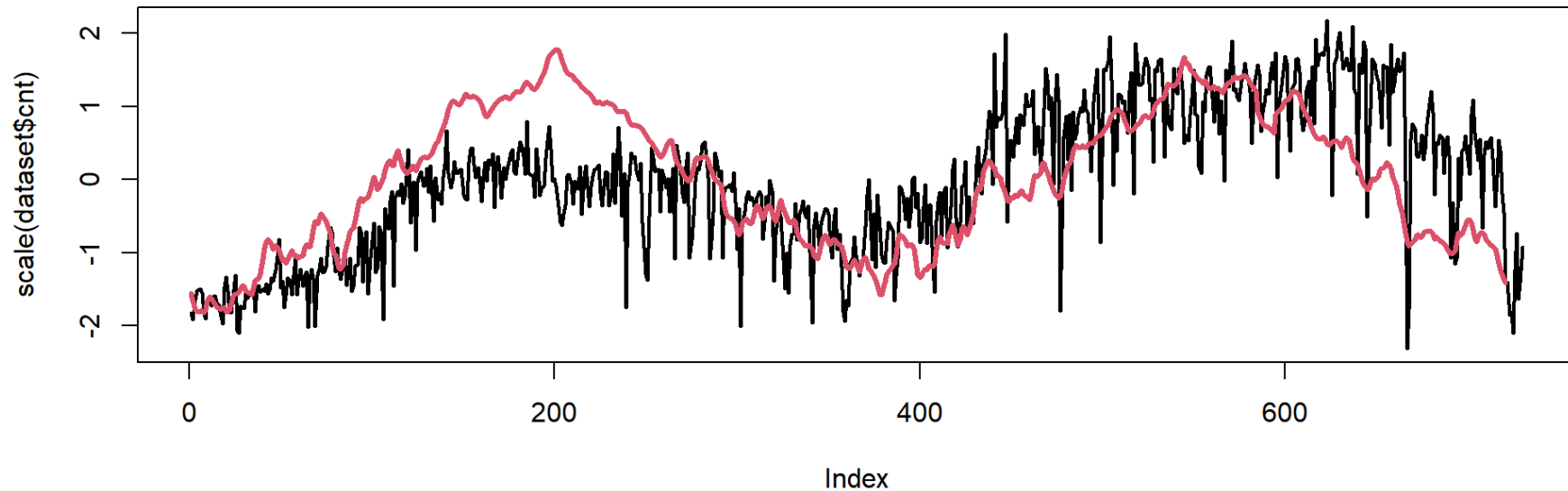
```
cor(dataset$temp, dataset$cnt, method = 'kendall')
```

```
## [1] 0.431533
```

Podemos ver uma no qual quando maior a temperatura maior será os alugueis.

Correlação entre alugueis e sensação de temperatura

```
roll = rollapply(scale(dataset$atemp), FUN=mean, width=10, align='right')  
plot(scale(dataset$cnt), type='l', lwd=2)  
lines(roll, col=2, lwd=3, lty = 1)
```



Podemos ver o comportamento entre essas duas variáveis no decorrer do tempo assim como alugueis e temperatura média em relação aos 10 ultimos pontos.

```
cor(dataset$temp, dataset$cnt, method = 'kendall')
```

```
## [1] 0.431533
```

Para treinamento do modelo será utilizado somente sensação termica(*atemp*) e não a temperatura real(*temp*), pois:

```
cor(dataset$temp, dataset$atemp)
```

```
## [1] 0.9916962
```

Os atributos *temp* e *atemp* possuem uma correlação altíssima e pode surgir um problema conhecido como **multicolinearidade**.

Possue alguma relação entre a diferença de *temp* e *atemp* e a target (*cnt*)

```
cor.test((dataset$temp-dataset$atemp), dataset$cnt, method = 'kendall')
```

```
##  
## Kendall's rank correlation tau  
##  
## data: (dataset$temp - dataset$atemp) and dataset$cnt  
## z = -13.848, p-value < 2.2e-16  
## alternative hypothesis: true tau is not equal to 0  
## sample estimates:  
##      tau  
## -0.3425459
```

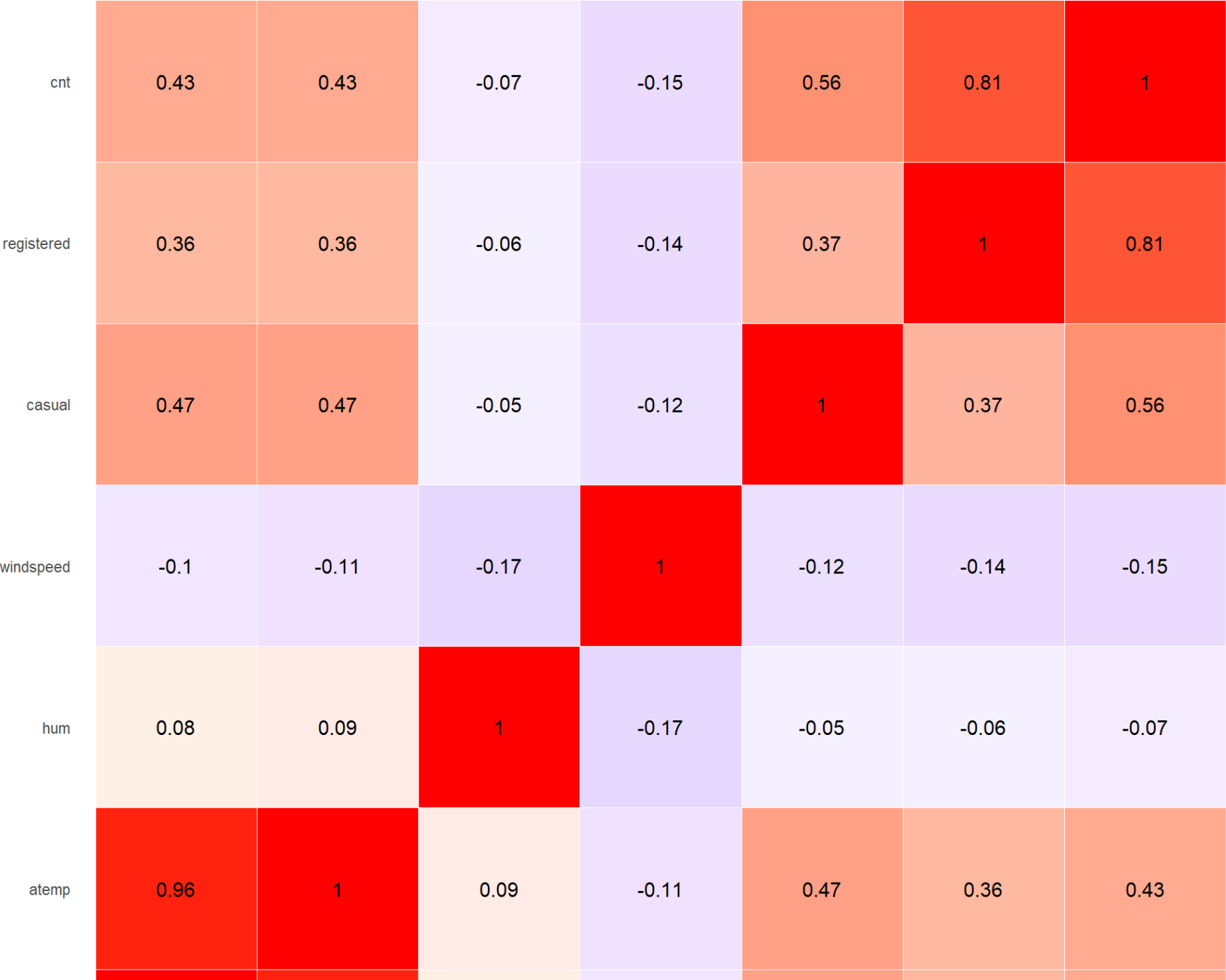
Correlação da target(*cnt*) e as demais features

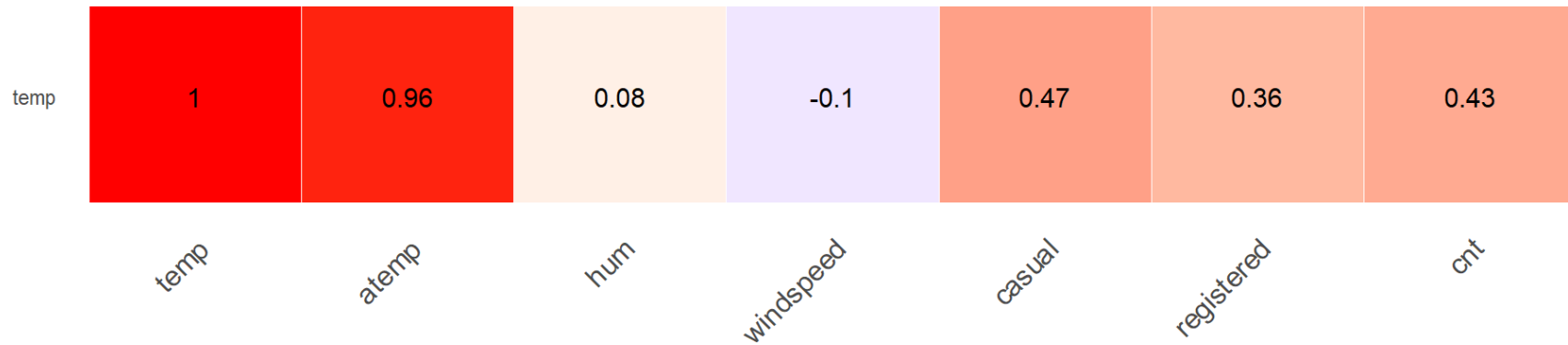
```
numeros = numeros[9:15]
```

```
dtCor = round(cor(dataset[,..numeros], method='kendall'), 2)  
dtCor = melt(dtCor)
```

```
ggheatmap <- ggplot(dtCor, aes(Var2, Var1, fill = value))+  
  geom_tile(color = "white")+  
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",  
    midpoint = 0, limit = c(-1,1), space = "Lab",  
    name="Pearson\nCorrelation") +  
  theme_minimal()+ # minimal theme  
  theme(axis.text.x = element_text(angle = 45, vjust = 1,  
    size = 12, hjust = 1))+  
  coord_fixed()  
  
ggheatmap +  
  geom_text(aes(Var2, Var1, label = value), color = "black", size = 4) +  
  theme(  
    axis.title.x = element_blank(),  
    axis.title.y = element_blank(),  
    panel.grid.major = element_blank(),  
    panel.border = element_blank(),  
    panel.background = element_blank(),  
    axis.ticks = element_blank(),  
    legend.justification = c(1, 0),  
    legend.position = c(1, 1),  
    legend.direction = "horizontal")+  
  guides(fill = guide_colorbar(barwidth = 7, barheight = 1,  
    title.position = "top", title.hjust = 0.5))
```

```
## Warning: A numeric `legend.position` argument in `theme()` was deprecated in ggplot2  
## 3.5.0.  
## i Please use the `legend.position.inside` argument of `theme()` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was  
## generated.
```





Será removido correlações baixo de 0.1 seguem abaixo as que serão removidas

- hum

## Casual e registered

```
length(dataset$instant)
```

```
## [1] 730
```

```
sum(( dataset$casual + dataset$registered) == (dataset$cnt))
```

```
## [1] 730
```

*cnt* é a soma de *casual* com *registered* com isso não faz sentido utilizar para o modelo de regressão já que coloca um viés no modelo

## Ajustando modelo

Será leito novamente o conj. de dados

```
data = fread("dataset/day.csv", sep = ',')
```

## Removendo atributos que não serão utilizados para treinamento do modelo

```
# removendo temp
data[, temp:=NULL]

# removendo dteday
data[, dteday:=NULL]

# removendo instant
data[, instant:=NULL]

data[, registered:=NULL]

data[, casual:=NULL]

data[, hum:=NULL]
```

## Modificando o tipo para as.factor

```
data[, season :=as.factor(season)]
data[, holiday:=as.factor(holiday)]
data[, weekday:=as.factor(weekday)]
data[, workingday:=as.factor(workingday)]
data[, weathersit:=as.factor(weathersit)]
data[, yr:=as.factor(yr)]
data[, mnth:=as.factor(mnth)]
```

data

season <fct>	yr <fct>	mnth <fct>	holiday <fct>	weekday <fct>	workingday <fct>	weathersit <fct>	atemp <dbl>	windspeed <dbl>	cnt <int>
1	0	1	0	6	0	2	18.181250	10.749882	985
1	0	1	0	0	0	2	17.686950	16.652113	801
1	0	1	0	1	1	1	9.470250	16.636703	1349

season	yr	mnth	holiday	weekday	workingday	weathersit	atemp				windspeed				cnt
<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<dbl>				<dbl>				<int>
1	0	1	0	2	1	1	10.606100				10.739832				1562
1	0	1	0	3	1	1	11.463500				12.522300				1600
1	0	1	0	4	1	1	11.660450				6.000868				1606
1	0	1	0	5	1	2	10.441950				11.304642				1510
1	0	1	0	6	0	2	8.112700				17.875868				959
1	0	1	0	0	0	1	5.808750				24.250650				822
1	0	1	0	1	1	1	7.544400				14.958889				1321
1-10 of 730 rows						Previous	1	2	3	4	5	6	...	73	Next

Utilizando o forward para encontrar o melhor modelo e parametros desse modelo

```
# Criando modelo com somente o intercepto
modelInicial = lm(cnt ~ 1, data=data)

# Criando modelo com todas os atributos
modelCompleto = lm(cnt~., data = data)

# Selecionando o melhor modelo com stepwise
bestForward = step(modelInicial,
                    scope=list(
                        lower=modelInicial,
                        upper=modelCompleto
                    ),
                    direction="forward")
```



```

## Start:  AIC=11050.84
## cnt ~ 1
##
##           Df Sum of Sq      RSS   AIC
## + atemp    1 1086848335 1645546526 10683
## + mnth     11 1063738316 1668656545 10713
## + season    3  944051737 1788343124 10747
## + yr        1  886909480 1845485381 10766
## + weathersit 2  269371947 2463022914 10979
## + windspeed 1 151066708 2581328153 11011
## + holiday   1  12920003 2719474858 11049
## + workingday 1  10687682 2721707179 11050
## <none>                2732394861 11051
## + weekday   6  17958746 2714436115 11058
##
## Step:  AIC=10682.65
## cnt ~ atemp
##
##           Df Sum of Sq      RSS   AIC
## + yr        1 798400233  847146293 10200
## + weathersit 2 162666940 1482879586 10611
## + season    3 152430786 1493115740 10618
## + mnth     11 133505174 1512041351 10643
## + windspeed 1  40158159 1605388367 10667
## + holiday   1  6338610 1639207916 10682
## <none>                1645546526 10683
## + workingday 1  2328780 1643217746 10684
## + weekday   6 15518416 1630028110 10688
##
## Step:  AIC=10199.97
## cnt ~ atemp + yr
##
##           Df Sum of Sq      RSS   AIC
## + season    3 164789487 682356807 10048
## + mnth     11 152313486 694832807 10077
## + weathersit 2 120904291 726242003 10092
## + windspeed 1  39092408 808053885 10168
## + holiday   1  7803329 839342964 10195

```

```

## + weekday      6 16645729 830500565 10198
## + workingday   1 2823410 844322883 10200
## <none>                847146293 10200
##
## Step: AIC=10048.05
## cnt ~ atemp + yr + season
##
##           Df Sum of Sq      RSS      AIC
## + weathersit  2 150931305 531425502 9869.6
## + mnth       11 44881070 637475736 10020.4
## + windspeed   1 23546165 658810641 10024.4
## + weekday     6 16926340 665430467 10041.7
## + holiday     1 7115065 675241742 10042.4
## + workingday  1 2878108 679478699 10047.0
## <none>                682356807 10048.1
##
## Step: AIC=9869.56
## cnt ~ atemp + yr + season + weathersit
##
##           Df Sum of Sq      RSS      AIC
## + mnth       11 56530261 474895241 9809.5
## + windspeed   1 15942298 515483204 9849.3
## + weekday     6 21427552 509997950 9851.5
## + holiday     1 10002522 521422980 9857.7
## + workingday  1 6094907 525330595 9863.1
## <none>                531425502 9869.6
##
## Step: AIC=9809.46
## cnt ~ atemp + yr + season + weathersit + mnth
##
##           Df Sum of Sq      RSS      AIC
## + windspeed   1 17050673 457844568 9784.8
## + weekday     6 21636566 453258675 9787.4
## + holiday     1 7570738 467324503 9799.7
## + workingday  1 6262816 468632425 9801.8
## <none>                474895241 9809.5
##
## Step: AIC=9784.77

```

```
## cnt ~ atemp + yr + season + weathersit + mnth + windspeed
##
##           Df Sum of Sq      RSS   AIC
## + weekday    6  21746217 436098351 9761.2
## + holiday     1   7408346 450436223 9774.9
## + workingday  1   6005119 451839449 9777.1
## <none>                        457844568 9784.8
##
## Step: AIC=9761.24
## cnt ~ atemp + yr + season + weathersit + mnth + windspeed + weekday
##
##           Df Sum of Sq      RSS   AIC
## + holiday     1   5533611 430564740 9753.9
## + workingday  1   5533611 430564740 9753.9
## <none>                        436098351 9761.2
##
## Step: AIC=9753.92
## cnt ~ atemp + yr + season + weathersit + mnth + windspeed + weekday +
##       holiday
##
##           Df Sum of Sq      RSS   AIC
## <none>                        430564740 9753.9
```

```
#
summary(bestForward)
```

```
##
## Call:
## lm(formula = cnt ~ atemp + yr + season + weathersit + mnth +
##      windspeed + weekday + holiday, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4197.5  -374.5    62.7   486.8  2893.6
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   600.947    182.319   3.296 0.001030 **
## atemp         81.304     8.447   9.625 < 2e-16 ***
## yr1          2078.551    58.433  35.571 < 2e-16 ***
## season2       834.811    182.400   4.577 5.58e-06 ***
## season3       855.493    216.677   3.948 8.66e-05 ***
## season4      1584.552    184.131   8.606 < 2e-16 ***
## weathersit2   -696.328     63.201 -11.018 < 2e-16 ***
## weathersit3 -2400.105    180.497 -13.297 < 2e-16 ***
## mnth2         212.945    146.593   1.453 0.146773
## mnth3         666.007    166.469   4.001 6.98e-05 ***
## mnth4         646.651    248.972   2.597 0.009593 **
## mnth5         920.368    266.767   3.450 0.000594 ***
## mnth6         897.389    273.981   3.275 0.001107 **
## mnth7         389.368    307.825   1.265 0.206326
## mnth8         749.736    296.179   2.531 0.011579 *
## mnth9        1143.587    264.225   4.328 1.72e-05 ***
## mnth10        579.632    244.615   2.370 0.018078 *
## mnth11       -102.242    234.699  -0.436 0.663239
## mnth12       -119.913    185.241  -0.647 0.517627
## windspeed     -31.301     5.934  -5.275 1.77e-07 ***
## weekday1      217.940    111.376   1.957 0.050768 .
## weekday2      333.045    108.897   3.058 0.002310 **
## weekday3      424.611    109.404   3.881 0.000114 ***
## weekday4      447.491    108.815   4.112 4.38e-05 ***
## weekday5      497.846    108.625   4.583 5.42e-06 ***
## weekday6      472.254    108.256   4.362 1.48e-05 ***
## holiday1     -549.844    182.926  -3.006 0.002743 **
```

```
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## Residual standard error: 782.6 on 703 degrees of freedom  
## Multiple R-squared:  0.8424, Adjusted R-squared:  0.8366  
## F-statistic: 144.5 on 26 and 703 DF,  p-value: < 2.2e-16
```

Os parametros maior são estatisticacamente significativo para um nivel de significancia de 0.001

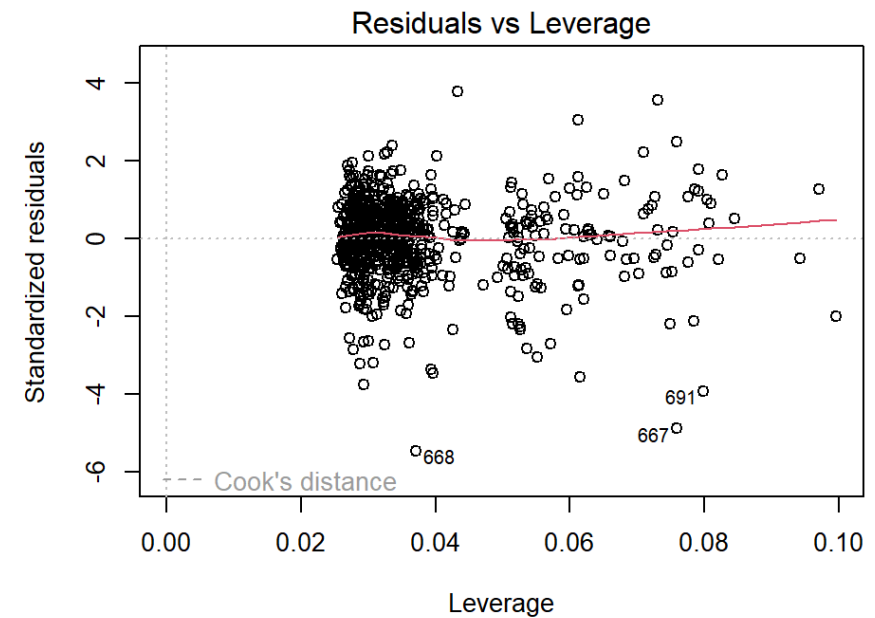
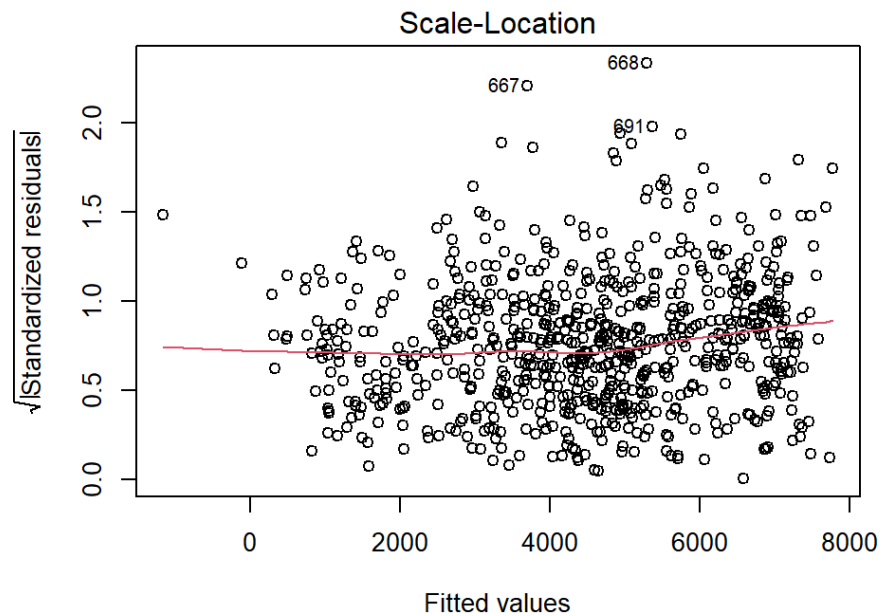
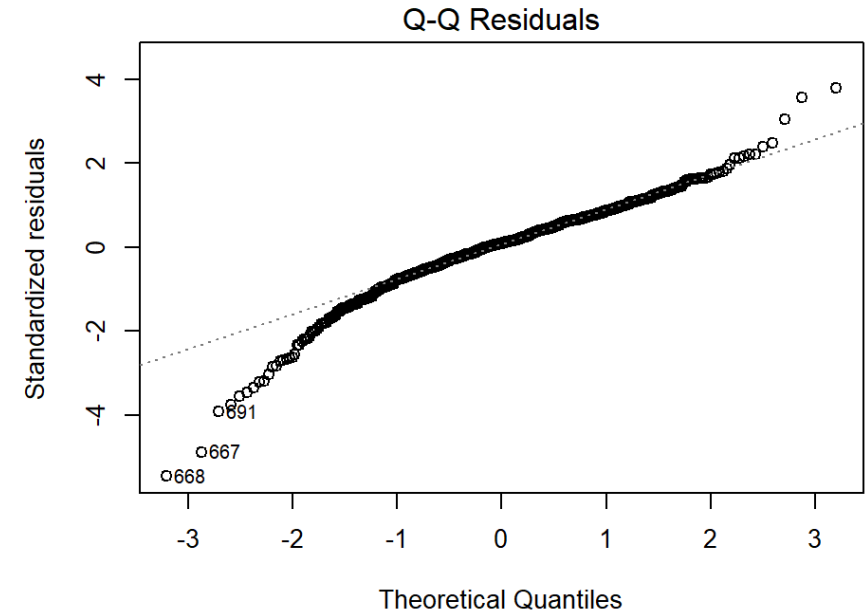
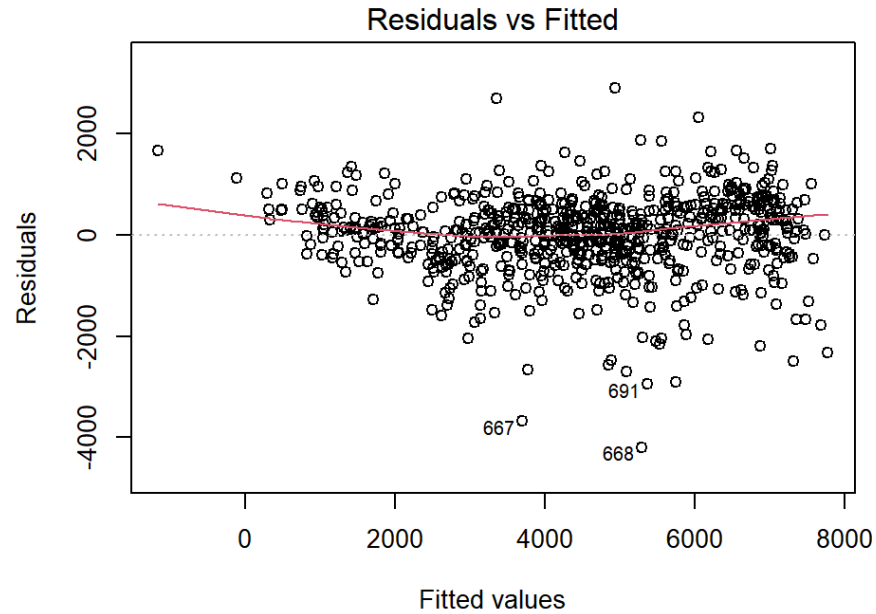
## Interpretação dos coeficientes

1. **atemp** : Para cada aumento de 1 grau na sensação térmica, o número de bicicletas alugadas aumenta em aproximadamente 81, mantendo todas as outras variáveis constantes. Isso indica que dias com sensação térmica mais agradável tendem a aumentar a demanda por bicicletas.
2. **mnth** : dependendo da mes maior será o numero de alugueis, pois há meses em que o valor do coeficiente é maior.
3. **weekday** : dependedo do dia do mes maior será o número de alugueis
  - A mesma logica para weathersit e season
4. **windspeed** : Para cada aumento de 1 unidade na velocidade do vento, o número de bicicletas alugadas diminui em aproximadamente 31, mantendo todas as outras variáveis constantes. Ventos mais fortes desencorajam o uso de bicicletas.

bestForward

```
##
## Call:
## lm(formula = cnt ~ atemp + yr + season + weathersit + mnth +
##      windspeed + weekday + holiday, data = data)
##
## Coefficients:
## (Intercept)          atemp             yr1        season2        season3        season4
##      600.9           81.3        2078.6         834.8         855.5        1584.6
## weathersit2 weathersit3         mnth2         mnth3         mnth4         mnth5
##     -696.3     -2400.1         212.9         666.0         646.7         920.4
##      mnth6      mnth7      mnth8      mnth9      mnth10      mnth11
##      897.4      389.4      749.7     1143.6      579.6       -102.2
##     mnth12  windspeed  weekday1  weekday2  weekday3  weekday4
##     -119.9     -31.3      217.9      333.0      424.6      447.5
##   weekday5  weekday6  holiday1
##     497.8     472.3    -549.8
```

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



```
formula(bestForward)
```

```
## cnt ~ atemp + yr + season + weathersit + mnth + windspeed + weekday +  
##      holiday
```

## Comparando modelo proposto

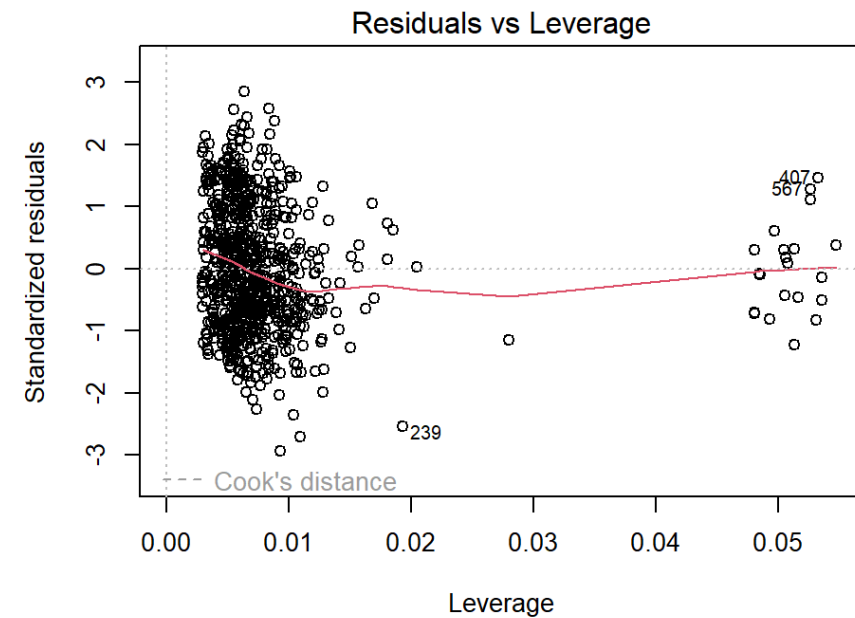
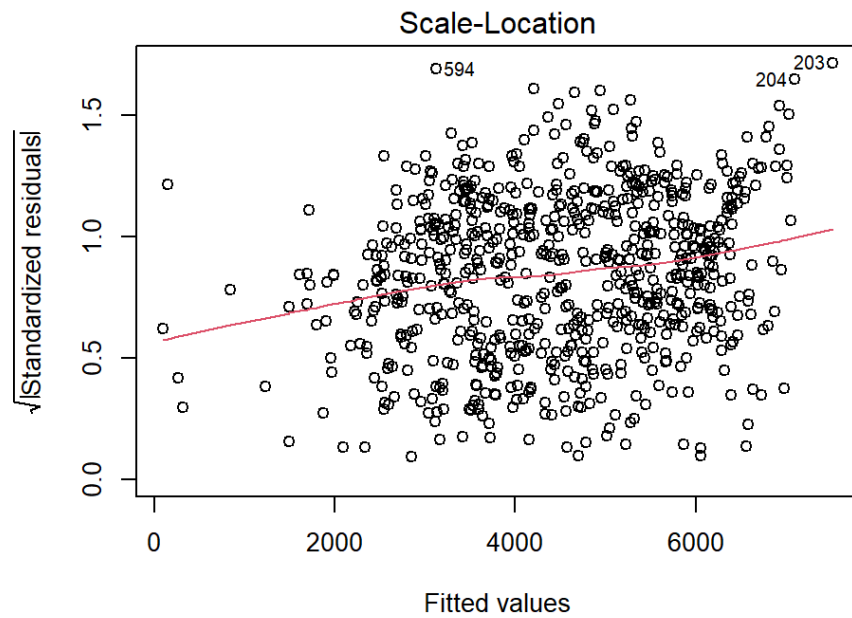
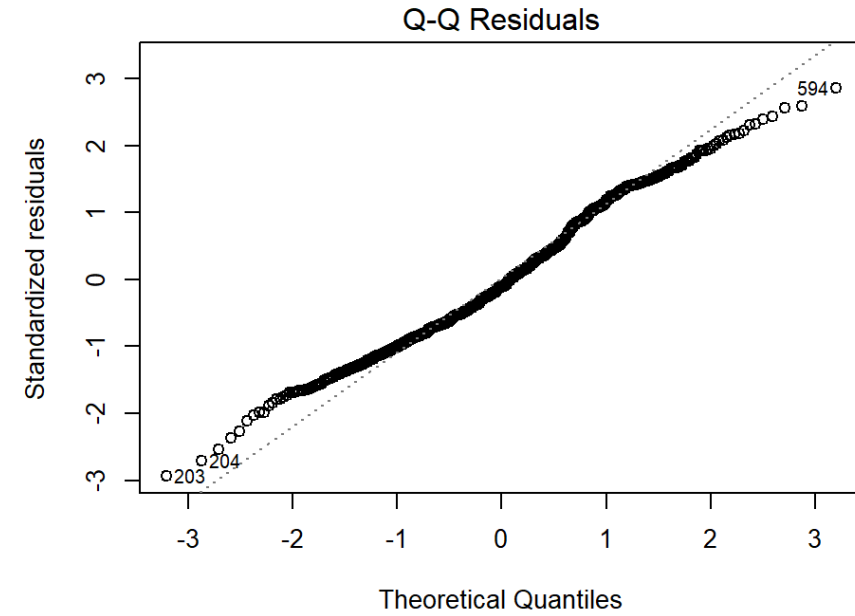
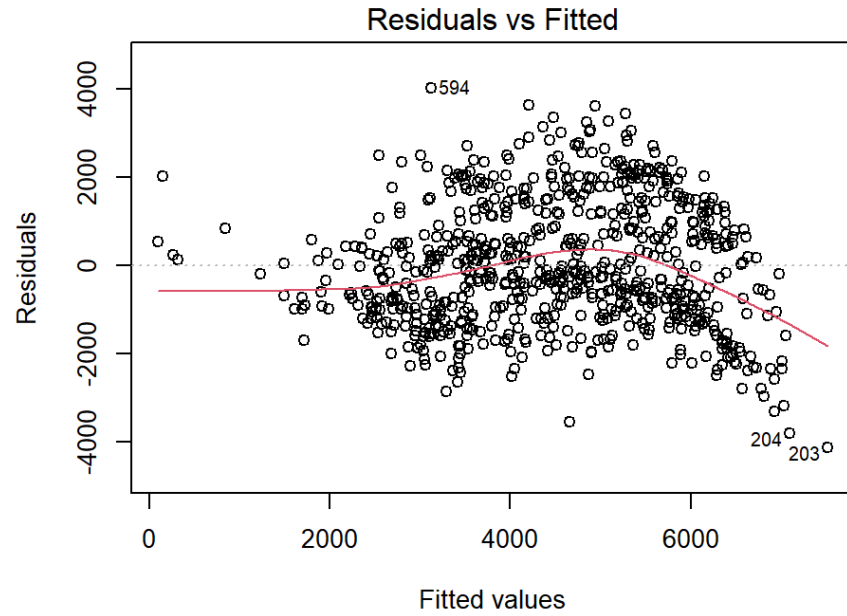
$$alugueis = \beta_0 + \beta_1 \cdot temperatura + \beta_2 \cdot vento + \beta_3 \cdot precipitacao + \beta_4 \cdot feriado + \varepsilon$$

```
model.proposto = lm(cnt~atemp+windspeed+weathersit+workingday,data = data)  
  
summary(model.proposto)
```



```
##
## Call:
## lm(formula = cnt ~ atemp + windspeed + weathersit + workingday,
##     data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4136  -1018   -156   1094   4021
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1900.982    244.147   7.786 2.39e-14 ***
## atemp         138.022     6.592  20.939 < 2e-16 ***
## windspeed     -40.396     10.321  -3.914 9.94e-05 ***
## weathersit2    -607.991    112.413  -5.409 8.64e-08 ***
## weathersit3   -2426.293    318.261  -7.624 7.76e-14 ***
## workingday1    179.266    112.923   1.588  0.113
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1414 on 724 degrees of freedom
## Multiple R-squared:  0.4704, Adjusted R-squared:  0.4668
## F-statistic: 128.6 on 5 and 724 DF, p-value: < 2.2e-16
```

```
par(mfrow=c(2, 2))
plot(model.proposto)
```



O modelo proposto apesar de ter maioria dos coeficiente estatisticamente significativos com exclusão workingday. Modelos não desempenhou bons resultados, por exemplo, R2\_score é ruim.

## Usando python para criação do modelo

Será feito o mesmo processo no R, com só a mudança na escolha de variáveis e modelos para treinar. Abaixo mostra as libs que serão utilizadas para essa etapa.

```
import keras
import tensorflow as tf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import Ridge
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.feature_selection import RFE
from sklearn.preprocessing import StandardScaler
```

Lendo o arquivo csv com o pandas

```
data = pd.read_csv("dataset/day.csv", sep = ',')
print(data.head(3))
```

```
##      instant      dteday  season  yr  ...  windspeed  casual  registered  cnt
## 0         1  01-01-2018      1    0  ...   10.749882    331           654    985
## 1         2  02-01-2018      1    0  ...   16.652113    131           670    801
## 2         3  03-01-2018      1    0  ...   16.636703    120          1229   1349
##
## [3 rows x 16 columns]
```

Fazendo o dummies das colunas categoricas (no r foi as.factor)

```
col = ["season", "yr", "mnth", "holiday", "weekday", "workingday", "weathersit"]  
  
data = pd.get_dummies(data, columns=col, dtype=int)
```

```
columns = data.drop(columns=['cnt', 'instant', 'dteday', 'temp', 'casual', 'registered', 'hum']).columns
```

Removendo as mesmas colunas que fiz usando R

```
X = data.drop(columns=['cnt', 'instant', 'dteday', 'temp', 'casual', 'registered', 'hum']).values  
y = data['cnt'].values.reshape(-1, 1)
```

```
X.shape, y.shape
```

```
## ((730, 34), (730, 1))
```

## Dividindo treino e teste

```
xtrain, xtest, ytrain, ytest = train_test_split(X, y, train_size=0.8, random_state=42)
```

```
xtrain.shape[0], xtest.shape[0]
```

```
## (584, 146)
```

# Selecionando as features por meio *RFE*(Eliminação Recursiva de Características)

```
# Usando florestas extra para seleção de caracterisitcas
extra = ExtraTreesRegressor(n_estimators=150, max_features='sqrt', random_state=42)

# Chamando a função RFE
rfe = RFE(extra, n_features_to_select=10)

# treinando
col = rfe.fit_transform(xtrain,ytrain.ravel())
```

```
# As colunas que foram selecionadas foram...
columns[rfe.get_support()]
```

```
## Index(['atemp', 'windspeed', 'season_1', 'season_3', 'yr_0', 'yr_1', 'mnth_1',
##       'mnth_2', 'weathersit_1', 'weathersit_3'],
##       dtype='object')
```

Essa foram as colunas escolhidas

Escalonando as variaveis numericas



```
# Selecionando colunas para as caracteristicas e escalonando elas
Xtrain = StandardScaler().fit_transform(xtrain[:, rfe.get_support()])
Xtest = StandardScaler().fit_transform(xtest[:, rfe.get_support()])
```

Usando o modelo de Ridge e fazendo o gridsearch para escolha dos melhores hiperparametros



```

grid = GridSearchCV(
    estimator = Ridge(),
    param_grid=dict(
        alpha=np.around(np.random.RandomState(seed=1).uniform(0.1,1,10),2),
        max_iter = np.arange(500, 1000, 200),
        solver = ['cholesky', 'lsqr','sag']
    ),
)
grid.fit(Xtrain, ytrain)

```

▶ **GridSearchCV**     
 (https://scikit-learn.org/1.5/modules/generated/sklearn.model\_selection.GridSearchCV.html)

▶ **best\_estimator\_: Ridge**

▶ **Ridge**     
 (https://scikit-learn.org/1.5/modules/generated/sklearn.linear\_model.Ridge.html)

Escolhendo os melhores hiperparâmetros do modelo Ridge



```
grid.best_estimator_, grid.best_score_
```

```
## (Ridge(alpha=np.float64(0.75), max_iter=np.int64(900), solver='sag'), np.float64(0.7879958212377995))
```

```

ridge = Ridge(alpha=np.float64(0.75), max_iter=np.int64(500), solver='lsqr')
ridge.fit(Xtrain, ytrain)

```

▼ **Ridge**     
 (https://scikit-learn.org/1.5/modules/generated/sklearn.linear\_model.Ridge.html)

```
Ridge(alpha=np.float64(0.75), max_iter=np.int64(500), solver='lsqr')
```

Métricas para os dados de *train*

```
predTrain = ridge.predict(Xtrain)
```

```
r2 = keras.metrics.R2Score()
r2.update_state(predTrain, ytrain)

mse = keras.metrics.MeanSquaredError()
mse.update_state(predTrain, ytrain)

mae = keras.metrics.MeanAbsoluteError()
mae.update_state(predTrain, ytrain)

print(f"""
R2 Score.....: {r2.result():.5f}
Error quadratico médio..: {mse.result():.5f}
Erro médio absoluto....: {mae.result():.5f}
""")
```

```
##
## R2 Score.....: 0.75609
## Error quadratico médio..: 747324.00000
## Erro médio absoluto....: 639.07880
```

Fazendo validação cruzada para ver se as metricas são constante a cada iteração

```

from sklearn.model_selection import ShuffleSplit

shuffle = ShuffleSplit(n_splits=12, test_size=0.25)

iter    = []
r2List  = []
mseList = []
for i,(train, test) in enumerate(shuffle.split(Xtrain)):
    iter.append(i)

    xtrain_, xtest_ = Xtrain[train,:], Xtrain[test, :]
    ytrain_, ytest_ = ytrain[train,:], ytrain[test, :]

    # treinando modelo
    ridge_ = Ridge(alpha=np.float64(0.8903056927518509), max_iter=np.int64(500), solver='lsqr')
    ridge_.fit(xtrain_, ytrain_)

    # predizendo
    pred_ = ridge_.predict(xtest_)

    r2 = keras.metrics.R2Score()
    r2.update_state(predTrain, ytrain)

    mse = keras.metrics.MeanSquaredError()
    mse.update_state(predTrain, ytrain)

    # Adicionando resultados
    r2List.append(r2.result())
    mseList.append(mse.result())

```



### Ridge



```

Ridge(alpha=np.float64(0.8903056927518509), max_iter=np.int64(500),
      solver='lsqr')

```

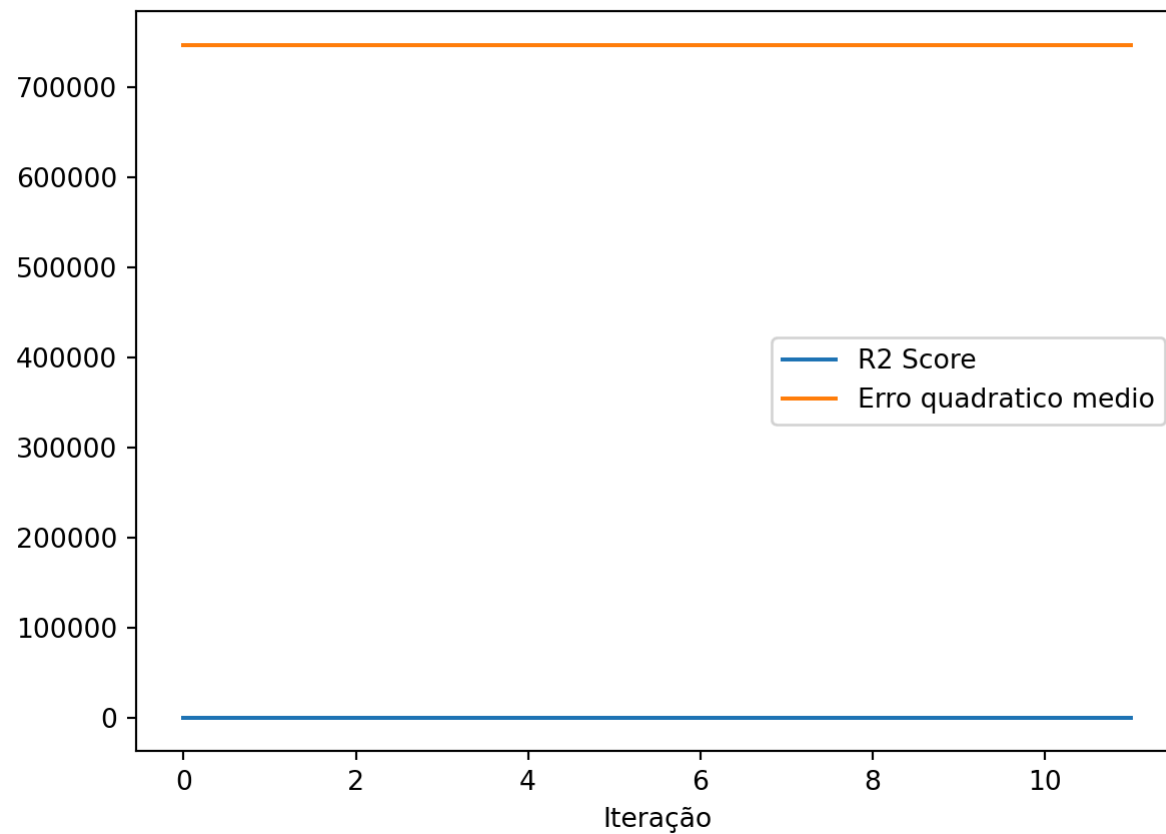
([https://scikit-learn.org/1.5/modules/generated/sklearn.linear\\_model.Ridge.html](https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.Ridge.html))

Criando um modelo gradiente descendente para minimizar os erros



```
plt.close()

plt.plot(r2List, label='R2 Score')
plt.plot(mseList, label='Erro quadratico medio')
plt.xlabel('Iteração')
plt.legend()
plt.show()
```



O modelo está tendo um bom desempenho agora vamos usar o dados teste que não foram utilizados

```
pred = ridge.predict(Xtest)

r2 = keras.metrics.R2Score()
r2.update_state(predTrain, ytrain)

mse = keras.metrics.MeanSquaredError()
mse.update_state(predTrain, ytrain)

mae = keras.metrics.MeanAbsoluteError()
mae.update_state(predTrain, ytrain)

print(f"""
R2 Score.....: {r2.result():.5f}
Error quadratico médio..: {mse.result():.5f}
Erro médio absoluto....: {mae.result():.5f}
""")
```

```
##
## R2 Score.....: 0.75609
## Error quadratico médio..: 747324.00000
## Erro médio absoluto....: 639.07880
```

Apesar do bom resultado e não apresentar overfitting o modelo não é melhor que o modelo criado no R via forward step, dessa forma a escolha do modelo do R é a melhor opção.

Poderia ser utilizado outros modelos que poderiam apresentar um resultado melhor do que no R como redes neurais, xboost-gradient, métodos ensemble. Prefiro utilizar Ridge que foi ensinado em sala de aula e se encaixa com o tema da cadeira.

## Conclusão

```
summary(bestForward)
```

```
##
## Call:
## lm(formula = cnt ~ atemp + yr + season + weathersit + mnth +
##      windspeed + weekday + holiday, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4197.5  -374.5    62.7   486.8  2893.6
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   600.947    182.319   3.296 0.001030 **
## atemp         81.304      8.447   9.625 < 2e-16 ***
## yr1          2078.551     58.433  35.571 < 2e-16 ***
## season2       834.811    182.400   4.577 5.58e-06 ***
## season3       855.493    216.677   3.948 8.66e-05 ***
## season4      1584.552    184.131   8.606 < 2e-16 ***
## weathersit2   -696.328     63.201 -11.018 < 2e-16 ***
## weathersit3 -2400.105    180.497 -13.297 < 2e-16 ***
## mnth2         212.945    146.593   1.453 0.146773
## mnth3         666.007    166.469   4.001 6.98e-05 ***
## mnth4         646.651    248.972   2.597 0.009593 **
## mnth5         920.368    266.767   3.450 0.000594 ***
## mnth6         897.389    273.981   3.275 0.001107 **
## mnth7         389.368    307.825   1.265 0.206326
## mnth8         749.736    296.179   2.531 0.011579 *
## mnth9        1143.587    264.225   4.328 1.72e-05 ***
## mnth10        579.632    244.615   2.370 0.018078 *
## mnth11       -102.242    234.699  -0.436 0.663239
## mnth12       -119.913    185.241  -0.647 0.517627
## windspeed     -31.301      5.934  -5.275 1.77e-07 ***
## weekday1      217.940    111.376   1.957 0.050768 .
## weekday2      333.045    108.897   3.058 0.002310 **
## weekday3      424.611    109.404   3.881 0.000114 ***
## weekday4      447.491    108.815   4.112 4.38e-05 ***
## weekday5      497.846    108.625   4.583 5.42e-06 ***
## weekday6      472.254    108.256   4.362 1.48e-05 ***
## holiday1     -549.844    182.926  -3.006 0.002743 **
```

```
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## Residual standard error: 782.6 on 703 degrees of freedom  
## Multiple R-squared:  0.8424, Adjusted R-squared:  0.8366  
## F-statistic: 144.5 on 26 and 703 DF,  p-value: < 2.2e-16
```

É o modelo escolhido pois desempenhou bons resultados.

```
formula(bestForward)
```

```
## cnt ~ atemp + yr + season + weathersit + mnth + windspeed + weekday +  
##      holiday
```