Trabalho 1

Caio Gomes Alves e Daniel Krügel

2023-11-15

Descritiva

```
require(car)
require(tidyverse)
require(forecast)
```

Escolhemos os dados presentes no pacote "car", chamado "Prestige". Escolhemos esse conjunto de dados por seu fácil acesso e rápida compreensão das variáveis.

```
dados  <- (Prestige)
head(dados)
```

```
##                     education income women prestige census type
## gov.administrators      13.11  12351 11.16     68.8   1113 prof
## general.managers        12.26  25879  4.02     69.1   1130 prof
## accountants             12.77   9271 15.70     63.4   1171 prof
## purchasing.officers     11.42   8865  9.11     56.8   1175 prof
## chemists                14.62   8403 11.68     73.5   2111 prof
## physicists              15.64  11030  5.13     77.6   2113 prof
```

```
summary(dados)
```

```
##    education          income          women          prestige
##  Min.   : 6.380   Min.   :  611   Min.   : 0.000   Min.   :14.80
##  1st Qu.: 8.445   1st Qu.: 4106   1st Qu.: 3.592   1st Qu.:35.23
##  Median :10.540   Median : 5930   Median :13.600   Median :43.60
##  Mean   :10.738   Mean   : 6798   Mean   :28.979   Mean   :46.83
##  3rd Qu.:12.648   3rd Qu.: 8187   3rd Qu.:52.203   3rd Qu.:59.27
##  Max.   :15.970   Max.   :25879   Max.   :97.510   Max.   :87.20
##      census       type
##  Min.   :1113   bc  :44
##  1st Qu.:3120   prof:31
```

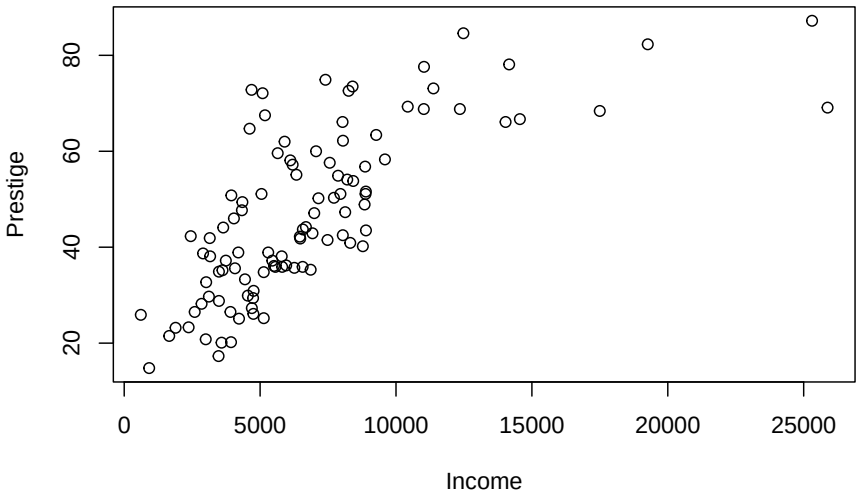
```
##  Median :5135   wc  :23
##  Mean   :5402   NA's: 4
##  3rd Qu.:8312
##  Max.   :9517
```

Estaremos ajustando como variável resposta a variável "Prestige" e como variável explicativa a variável "Income". Na documentação do pacote encontramos as seguintes definições:

Income - Average income of incumbents, dollars, in 1971.

Prestige - Pineo-Porter prestige score for occupation, from a social survey conducted in the mid-1960s.

Vamos montar um gráfico de dispersão para ver o rosto das nossas variáveis:



Agora que já vimos qual a cara dos nossos dados, vamos começar a ajustar algumas regressões vistas na disciplina.

Regressão linear

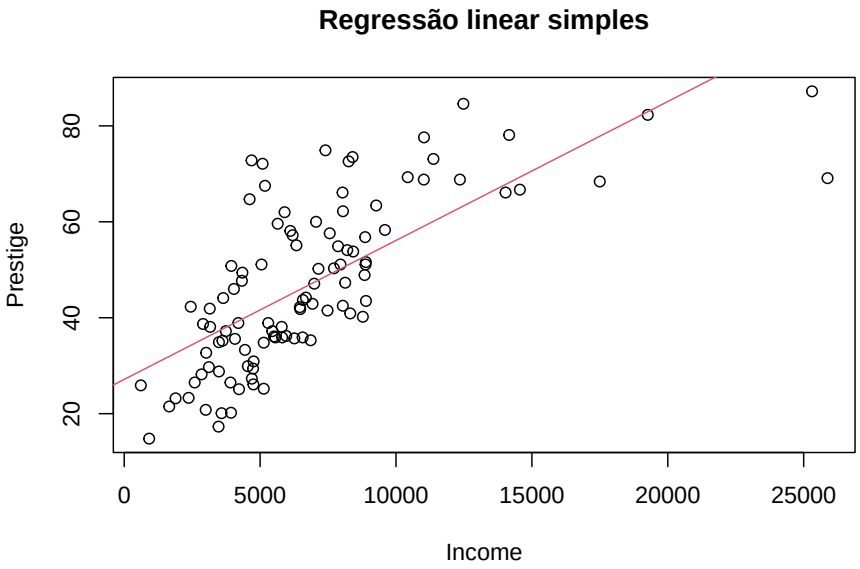
Vamos começar ajustando uma reta, sem mais de longas nos dados e ver como ela se saí:

```
fit01 <- lm(prestige ~ income, data = dados )

summary(fit01)
```

```
##
## Call:
## lm(formula = prestige ~ income, data = dados)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -33.007  -8.378  -2.378   8.432  32.084
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.714e+01  2.268e+00   11.97   <2e-16 ***
## income      2.897e-03  2.833e-04   10.22   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.09 on 100 degrees of freedom
## Multiple R-squared:  0.5111, Adjusted R-squared:  0.5062
## F-statistic: 104.5 on 1 and 100 DF,  p-value: < 2.2e-16
```

Vemos que o modelo aparece com um p valor significativo da estatística F, podendo sugerir que fornece um bom ajuste Porém os R quadrado e R quadrado ajustado aparecem relativamente baixos.

```
plot(dados$income, dados$prestige,
     xlab = "Income", ylab = "Prestige", main = "Regressão linear simples")
abline(coef(fit01), col = 2)
```



Quando colocamos a regressão em cima dos pontos vêmos o motivo, a reta ficou sobreposta de forma grosseira em cima dos dados.

Para contra balancear vamos ver os modelos polinomiais e dar uma certa maleada nesta curva.

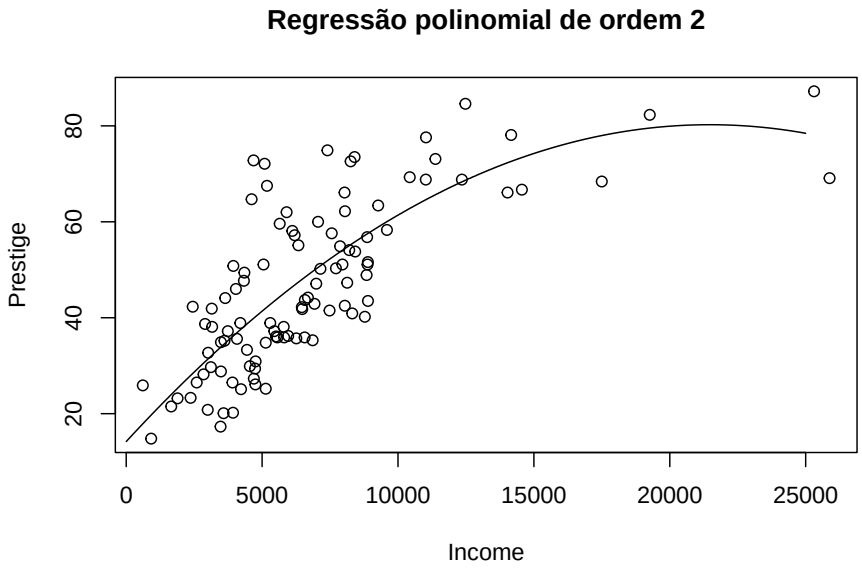
Modelo Polinomial

```
fit02 <- lm(prestige ~ income + I(income^2) ,data = dados )
summary(fit02)
```

```
##
## Call:
## lm(formula = prestige ~ income + I(income^2), data = dados)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.963  -7.967  -2.303   7.847  32.928
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.418e+01  3.515e+00   4.035 0.000108 ***
## income       6.154e-03  7.593e-04   8.104 1.43e-12 ***
## I(income^2) -1.433e-07  3.141e-08  -4.562 1.45e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.04 on 99 degrees of freedom
## Multiple R-squared:  0.596,  Adjusted R-squared:  0.5879
## F-statistic: 73.03 on 2 and 99 DF,  p-value: < 2.2e-16
```

Aqui já conseguimos ver uma melhora no R quadrado, vamos ver como isso reflete no nosso diagrama de dispersão:

```
plot(dados$income, dados$prestige,
     xlab = "Income", ylab = "Prestige", main = "Regressão polinomial de ordem 2")

nd <- data.frame(income = 1:25000)
nd$y <- predict(fit02, nd)
with(nd, lines(y ~ income, col=1))
```

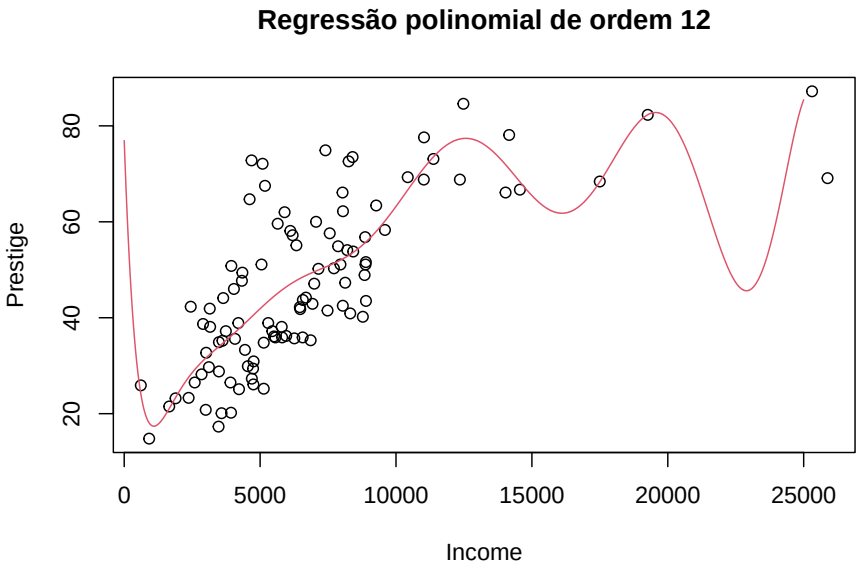


Aqui já vimos uma melhora considerável em relação ao modelo linear anterior, o modelo polinomial conseguiu pegar a curvatura dos dados.

Vamos ajustar alguns modelos de ordem maior para tentar fazer um overfitting dos dados e comparálo

```
fit03 <- lm(prestige ~ poly(income, 12),data = dados )
nd2 <- data.frame(income = 1:25000)
nd2$y <- predict(fit03, nd2)
```

```
plot(dados$income, dados$prestige,
     xlab = "Income", ylab = "Prestige", main = "Regressão polinomial de ordem 12")

with(nd2, lines(y ~ income, col=2))
```



Legal, agora vamos ver se adiantou de algo colocar tantos graus na nossa regressão. Como os modelos são encaixados, posso testar via anova se a diferença entre os dois é significativa ou não:

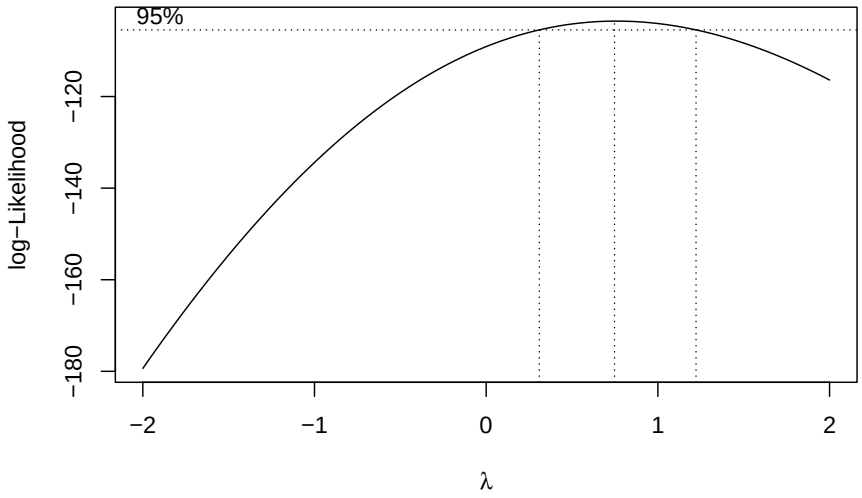
```
anova(fit02, fit03)
```

```
## Analysis of Variance Table
##
## Model 1: prestige ~ income + I(income^2)
## Model 2: prestige ~ poly(income, 12)
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1     99 12077
## 2     89 11329 10    748.26 0.5878 0.8199
```

A diferença dos resíduos apareceu com uma estatística F não significativa, ou seja, os modelos são iguais, portanto ficaremos com o modelo polinomial de ordem 2 para a nossa regressão e diremos que ele é o melhor modelo para esta categoria.

Transformação Box Cox

A ideia é transformar o modelo linear inicial com a sugestão do método box cox para tentarmos um ajuste mais digno

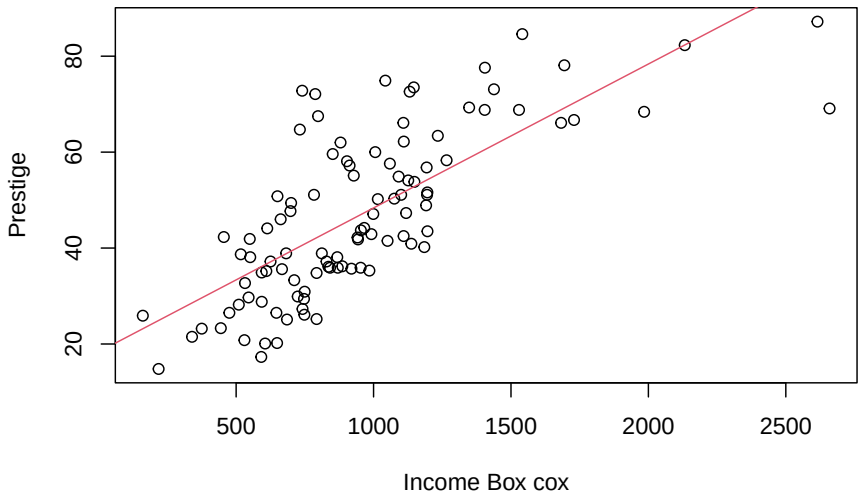


A transformação ficou próxima de 1, o que indica que provavelmente não é necessária a transformação, mas vamos realizá-la para testes

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

dados $income_boxcox <- ((dados $income^lambda) - 1)/lambda

fit0bc <- lm(prestige ~ income_boxcox, data = dados )
```

A transformação é dada por: $(income^{\lambda} - 1)/\lambda$

```r
plot(dados $income_boxcox, dados $prestige,
     xlab = "Income Box cox", ylab = "Prestige");
abline(coef(fit0bc), col = 2)
```

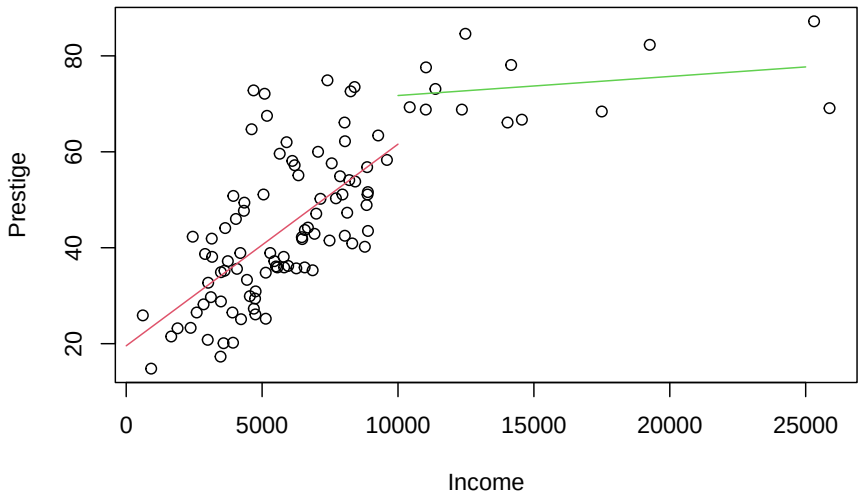


E estes são os dados plotados com a transformação realizada. Realmente não parece ter uma mudança drástica do caso linear comum.

Regressão por partes

Aqui precisamos primeiro decidir onde será feito o corte, decidimos em torno dos 10000 já que parece haver uma mudança na tendencia da curva em torno deste valor. Então precisamos começar a organizar os dados e as regressões cortando a base de dados

```r
# Cortando os dados
dado_filtrado_low <- dados %>%
  filter(income < 10000)

dado_filtrado_hig <- dados %>%
  filter(income > 10000)

# Ajustando as regressões para cada parte
fit_low <- lm(prestige ~ income, data = dado_filtrado_low)
fit_hig <- lm(prestige ~ income, data = dado_filtrado_hig)

# Predizendo so valores que usaremos para plotar
nd_low <- data.frame(income = 1:10000)
nd_low$y <- predict(fit_low, nd_low)

nd_hig <- data.frame(income = 10000:25000)
nd_hig$y <- predict(fit_hig, nd_hig)

plot(dados $income, dados $prestige,
     xlab = "Income", ylab = "Prestige");
with(nd_low, lines(y ~ income, col=2));
with(nd_hig, lines(y ~ income, col=3))
```



Utilizando library Segmented

Como a regressão por partes é muito custosa a se fazer na mão procuramos uma biblioteca que automatiza esse fator para nós e encontramos a Segmented, ela necessita de um valor inicial de chute para cada nó e tenta minimizar a função de verossimilhança iterativamente

```
library(segmented)
```

```
## Warning: package 'segmented' was built under R version 4.3.2
```

```
## Carregando pacotes exigidos: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Carregando pacotes exigidos: nlme
```

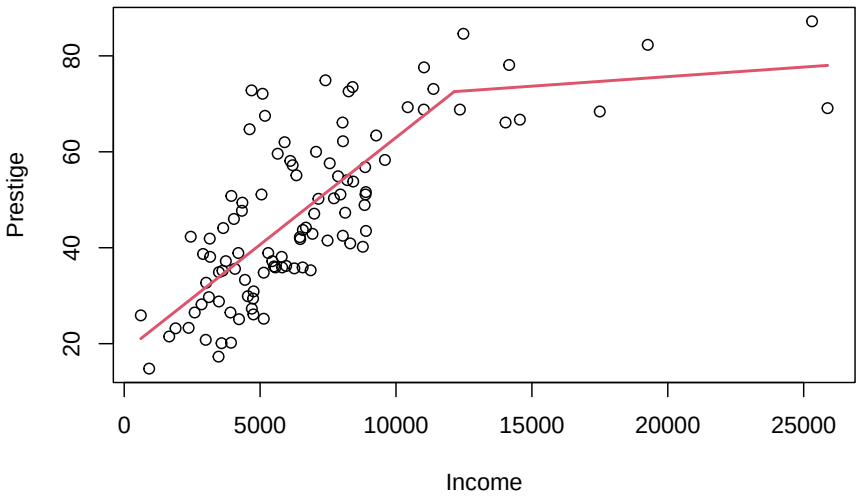
```
##
## Attaching package: 'nlme'

## The following object is masked from 'package:forecast':
##
##     getResponse

## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
fit.Segmentada <- segmented(fit01, seg.Z = ~income, psi = 10000)
summary(fit.Segmentada)
```

```
##
##  ***Regression Model with Segmented Relationship(s)***
##
## Call:
## segmented.lm(obj = fit01, seg.Z = ~income, psi = 10000)
##
## Estimated Break-Point(s):
##                  Est.   St.Err
## psi1.income 12142.1 1536.788
##
## Meaningful coefficients of the linear terms:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 18.3215272  3.0247901   6.057 2.57e-08 ***
## income       0.0044655  0.0004842   9.222 5.90e-15 ***
## U1.income   -0.0040673  0.0008909  -4.565       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.03 on 98 degrees of freedom
## Multiple R-Squared: 0.6014,  Adjusted R-squared: 0.5892
##
## Boot restarting based on 6 samples. Last fit:
## Convergence attained in 2 iterations (rel. change 2.2288e-14)
```

Com o chute inicial o algoritmo convergiu para um nó em 12.142,1 Plotando em cima dos dados

```
plot(dados $income, dados $prestige,
     xlab = "Income", ylab = "Prestige")
plot(fit.Segmentada, add = T)
```

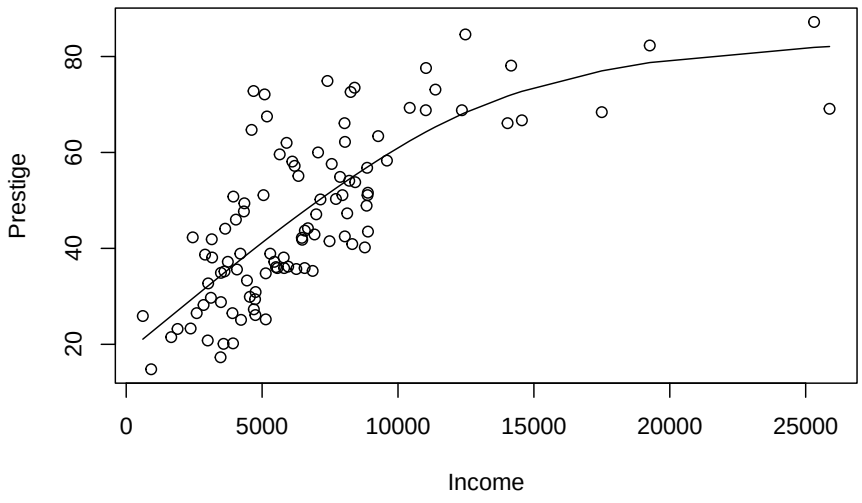


Vêmos que o pacote tem opções para mais formas de regressão, não encontramos nada para evitar que o nó ligue entre cada secção então ela acaba beirando o spline, que será o próximo ajuste a ser testado.

Smoothing Spline

A diferença entre regressão segmentada e Spline é que a regressão é continua. Como já fizemos uma regressão contínua no exemplo anterior pularemos direto para a Smoothing Spline, que utiliza a penalização da segunda derivada da curva no ponto nó para a soma de minimos quadrados. Para isso utilizaremos a função "smoothing.spline"

```
library(splines)

plot(dados $income, dados $prestige,
     xlab = "Income", ylab = "Prestige")
fitss <- smooth.spline(x = dados$income, y = dados$prestige, cv = F)

lines(fitss)
```



Desabilitamos a validação cruzada para conseguir contrastar melhor com a polinomial de grau 2, já que ambas as regressões ficaram muito parecidas

Mas para efeito de comparação, por que não plotar ambas?