

# Lista 1 - Mineração

Victor Alves Dogo Martins, RA: 744878      Ana Beatriz Alves Monteiro, RA: 727838  
Larissa Torres, RA: 631914

03-07-2022

## Item 1

Como orientado no enunciado deste item, foi feita a normalização da covariável ‘PIB per capita’ através da seguinte fórmula:

$$\frac{x - x_{min}}{x_{max} - x_{min}}$$

Computacionalmente, esse procedimento foi feito através do comando `mutate` abaixo, onde temos o comentário `# Normalizando a covariavel`. Além disso, também segue uma parcela do banco de dados após a normalização para fins de demonstração:

```
### Carregando Pacotes

library(tidyverse)
library(knitr)
library(kableExtra)

# Lendo dados

df <- readr::read_csv('worldDevelopmentIndicators.csv') |>
  select(-CountryName) |>
  rename(y=LifeExpectancy,
         x=GDPpercapita) |>
  mutate(x = (x-min(x))/(max(x)-min(x))) # Normalizando a covariavel

# Mostrando dados

head(df) |>
  kable('latex', digits=4, align='cc',
        caption = 'Primeiras linhas do banco de dados após normalização') |>
  kable_styling(position="center",
                latex_options="HOLD_position")
```

Table 1: Primeiras linhas do banco de dados após normalização

y	x
60.5091	0.0042
51.4640	0.0505
77.3505	0.0362
69.9497	0.0710
76.9579	0.4000
76.0127	0.1093

## Item 2

```
# Criando lista com formulas de g(x) para cada valor de p

formulas <- list()

for (p in 1:30) {

  if(p==1){

    # Se p for igual a 1, a expressao mantem-se da forma abaixo

    formulas[[p]] <- "y~sin(2*pi*x)+cos(2*pi*x)"
  } else {

    # Para cada p maior do que 1, sua expressao g(x) sera dada pela expressao
# do p anterior MAIS os seno e cosseno de 2*pi*x vezes o valor de p da
# iteracao atual

    formulas[[p]] <- paste0(formulas[[p-1]],
                           "+sin(2*",p,
                           "*pi*x)+cos(2*",
                           p, "*pi*x)")
  }
}

# Definido tibble para guardarmos estimativas do risco das regressões de cada
# valor de p

tbl_result <- tibble(
  p=1:30,
  mse=as.double(1:30)
)

# Calculando erro quadrático médio para cada p via leave-one-out

for (p in 1:length(formulas)) {

  erros <- NULL

  model <- NULL
```

```

for (ii in 1:nrow(df)) {

  # Para o p atual, ajusta-se uma regressao retirando cada obs.
  # ii e utilizando o restante do banco de dados para validarmos
  # o modelo com a obs. ii retirada

  model <- lm(formulas[[p]], data=df[-ii,])

  erros <- c(erros,
             (df[ii,1] - predict(model, df[ii,]))^2)
}

# Fazendo a media dos erros dos ajustes para cada uma das observacoes
# retiradas

tbl_result[p,2] <- mean(unlist(erros))
}

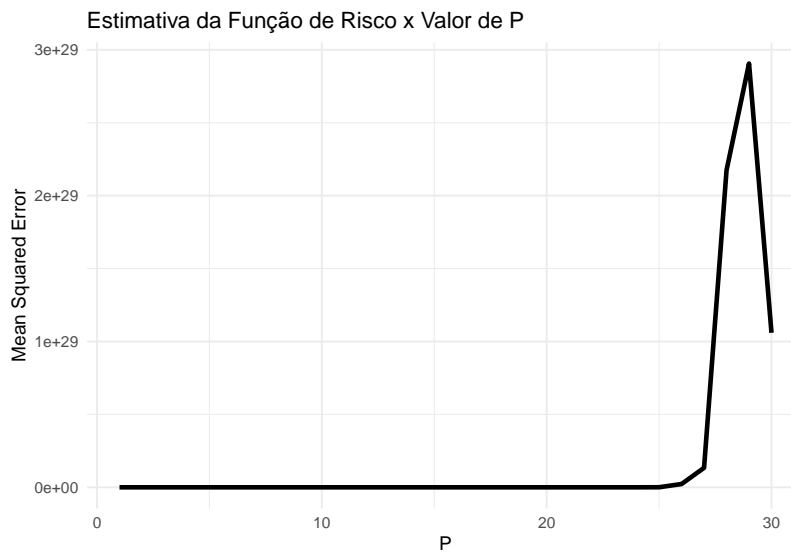
```

### Item 3

```

tbl_result |>
  ggplot()+
  aes(x=p,y=mse)+
  geom_line(size=1.25)+
  theme_minimal()+
  labs(x = 'P', y = "Mean Squared Error",
       title='Estimativa da Função de Risco x Valor de P')

```



```
tbl_result |>
  ggplot()+
  aes(x=p,y=mse)+
  geom_line(size=1.25)+
  geom_label(aes(label=paste0('mse = ', round(mse,3)))+
  coord_cartesian(ylim=c(40,50), xlim=c(1,6))+
  scale_x_continuous(breaks = 1:6)+
  theme_minimal()+
  labs(x = 'P', y = "Mean Squared Error",
  title='Estimativa da Função de Risco x Valor de P')
```

