# Exercicio Aprendizagem de Máquina

#

# Aluno: Fernando Albuquerque / José Wellington

#

# Configurando o diretório de trabalho

# Coloque entre aspas o diretório de trabalho que você está usando no seu computador

# Não use diretórios com espaço no nome

setwd("C:/\_wellington/\_Ciencia\_Dados/\_DSA/01\_Big\_Data\_Analytics\_com\_R\_e\_Microsoft\_Azure\_Machine\_108h/12\_Microshoft\_Azure\_Machine\_Learning")

getwd()

**# Regressão Linear**

**# Definição do Problema:** Prever as notas dos alunos com base em diversas métricas

<https://archive.ics.uci.edu/ml/datasets/Student+Performance>

**# Dataset com dados de estudantes**

# Atributos para os conjuntos de dados student-mat.csv (curso de matemática) e student-por.csv (curso de idioma português):

# 1 escola - escola do aluno (binário: 'GP' - Gabriel Pereira ou 'MS' - Mousinho da Silveira)

# 2 sexo - sexo do aluno (binário: 'F' - feminino ou 'M' - masculino)

# 3 idade - idade do aluno (numérica: de 15 a 22)

# 4 Endereço tipo de endereço residencial do aluno (binário: 'U' - urbano ou 'R' - rural)

# 5 famsize - tamanho da família (binário: 'LE3' - menor ou igual a 3 ou 'GT3' - maior que 3)

# 6 Pstatus - status de coabitação dos pais (binário: 'T' - morando juntos ou 'A' - separados)

# 7 Medu - escolaridade da mãe (numérico: 0 - nenhum, 1 - ensino fundamental (4ª série), 2 - 5ª a 9ª série, 3 - ensino médio ou 4 - ensino superior)

# 8 Fedu - educação do pai (numérico: 0 - nenhum, 1 - ensino fundamental (4a série), 2 - 5a a 9a série, 3 - educação secundária ou 4 - educação superior)

# 9 Mjob - trabalho da mãe (nominal: 'professor', 'serviços de saúde', 'serviços civis' (por exemplo, administrativos ou policiais), 'at\_home' ou 'outro')

# 10 Fjob - trabalho do pai (nominal: 'professor', 'serviços de saúde', 'serviços civis' (por exemplo, administrativos ou policiais), 'at\_home' ou 'outro')

# 11 motivo - motivo para escolher esta escola (nominal: próximo a 'casa', 'reputação' da escola, 'preferência do curso' ou 'outro')

# 12 guardião - guardião do aluno (nominal: 'mãe', 'pai' ou 'outro')

# 13 tempo de viagem - tempo de viagem entre casa e escola (numérico: 1 - <15 min., 2 - 15 a 30 min., 3 - 30 min a 1 hora ou 4 -> 1 hora)

# 14 tempo de estudo - tempo semanal de estudo (numérico: 1 - <2 horas, 2 - 2 a 5 horas, 3 - 5 a 10 horas ou 4 -> 10 horas)

# 15 falhas - número de falhas de classe passada (numérico: n se 1 <= n <3, senão 4)

# 16 escolas - suporte educacional extra (binário: sim ou não)

# 17 famsup - suporte educacional familiar (binário: sim ou não)

# 18 pagas - aulas extra pagas na disciplina (matemática ou português) (binário: sim ou não)

# 19 atividades - atividades extracurriculares (binárias: sim ou não)

# 20 creche - escola maternal frequentada (binária: sim ou não)

# 21 superior - quer cursar o ensino superior (binário: sim ou não)

# 22 internet - acesso à Internet em casa (binário: sim ou não)

# 23 romântico - com um relacionamento romântico (binário: sim ou não)

# 24 famrel - qualidade dos relacionamentos familiares (numérico: de 1 - muito ruim a 5 - excelente)

# 25 tempo livre - tempo livre depois da escola (numérico: de 1 - muito baixo a 5 - muito alto)

# 26 atividades - sair com os amigos (numérico: de 1 - muito baixo a 5 - muito alto)

# 27 Dalc - consumo diário de álcool no trabalho (numérico: de 1 - muito baixo a 5 - muito alto)

# 28 Walc - consumo de álcool no fim de semana (numérico: de 1 - muito baixo a 5 - muito alto)

# 29 estado de saúde atual (numérico: de 1 - muito ruim a 5 - muito bom)

# 30 faltas - número de faltas escolares (numérico: de 0 a 93)

# estas notas estão relacionadas com a disciplina, matemática ou português:

# 31 N1 - primeira série do período (numérica: de 0 a 20)

# 31 N2 - segunda série do período (numérica: de 0 a 20)

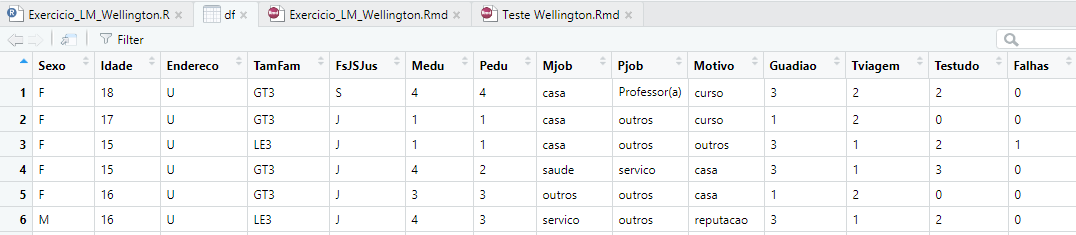
# 32 NF - nota final (numérica: de 0 a 20, meta de saída)

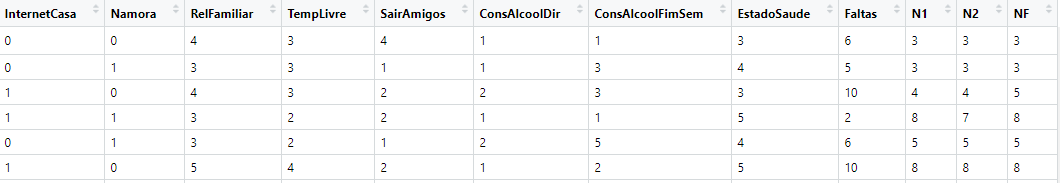
# Vamos prever a nota final (grade) dos alunos

**# Carregando o dataset**

**df <- read.csv2('estudantesB.csv')**

**View(df)**



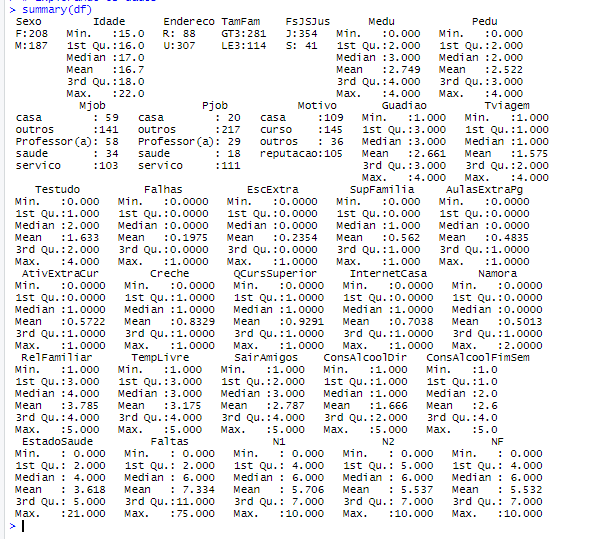


**# Explorando os dados**

**summary(df)**

**str(df)**

**any(is.na(df))**



**# Instalando os pacotes**

**install.packages("ggplot2")**

**install.packages("ggthemes")**

**install.packages("dplyr")**

**library(ggplot2)**

**library(ggthemes)**

**library(dplyr)**

**# Obtendo apenas as colunas numéricas**

**# Perguntar se é uma variável numérica**

**colunas\_numericas <- sapply(df, is.numeric)**

**colunas\_numericas**

**# Filtrando as colunas numéricas para correlação**

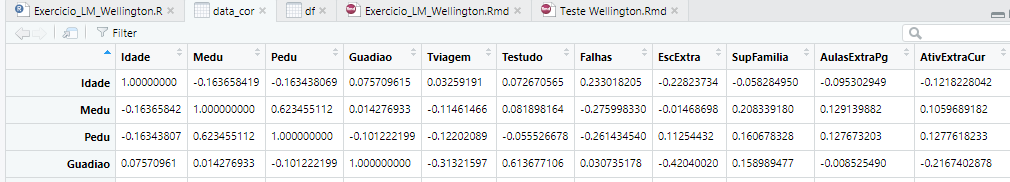
**# função cor() df [todas linhas, colunas numericas]**

**data\_cor <- cor(df[,colunas\_numericas])**

**data\_cor**

**View(data\_cor)**

**head(data\_cor)**



**# Pacotes para visualizar a análise de correlação**

**install.packages('corrgram')**

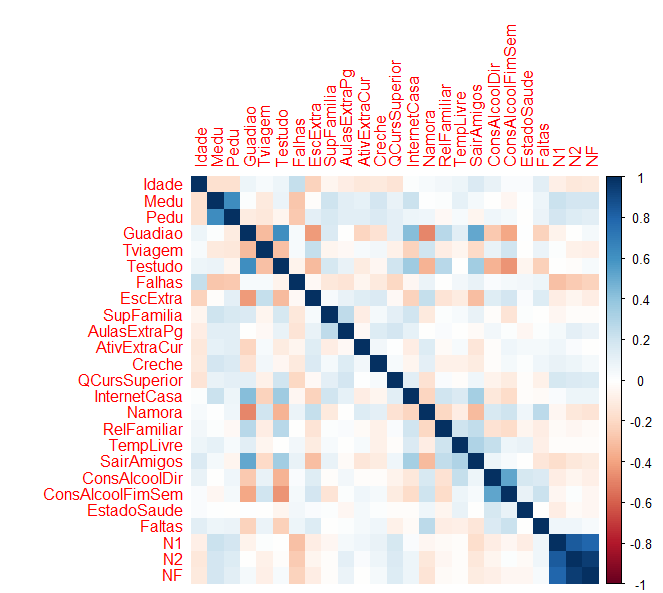
**install.packages('corrplot')**

**library(corrplot)**

**library(corrgram)**

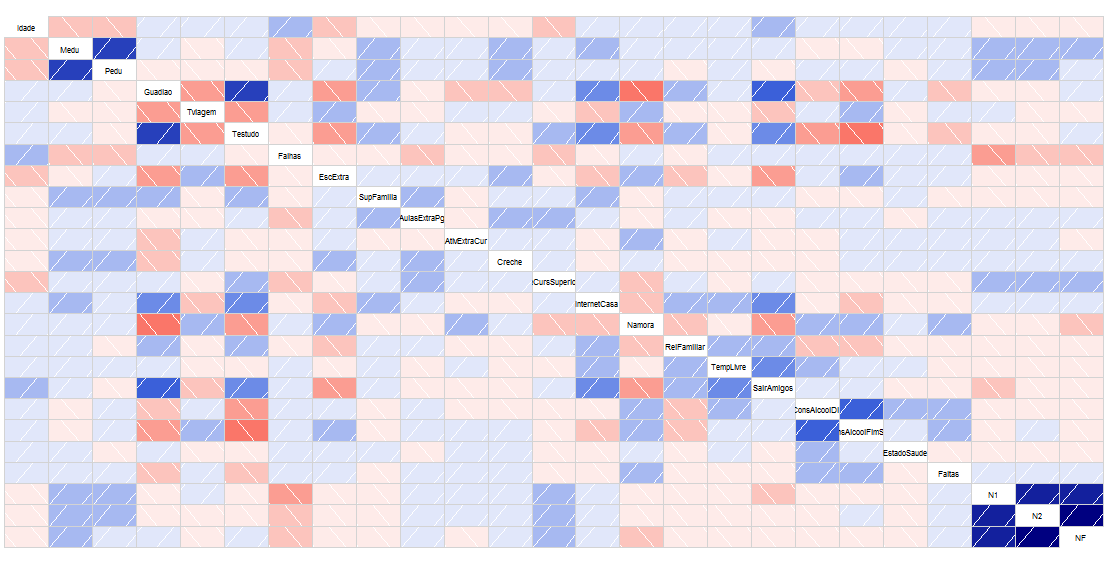
**# Criando um corrplot**

**corrplot(data\_cor, method = 'color')**



**# Criando um corrgram**

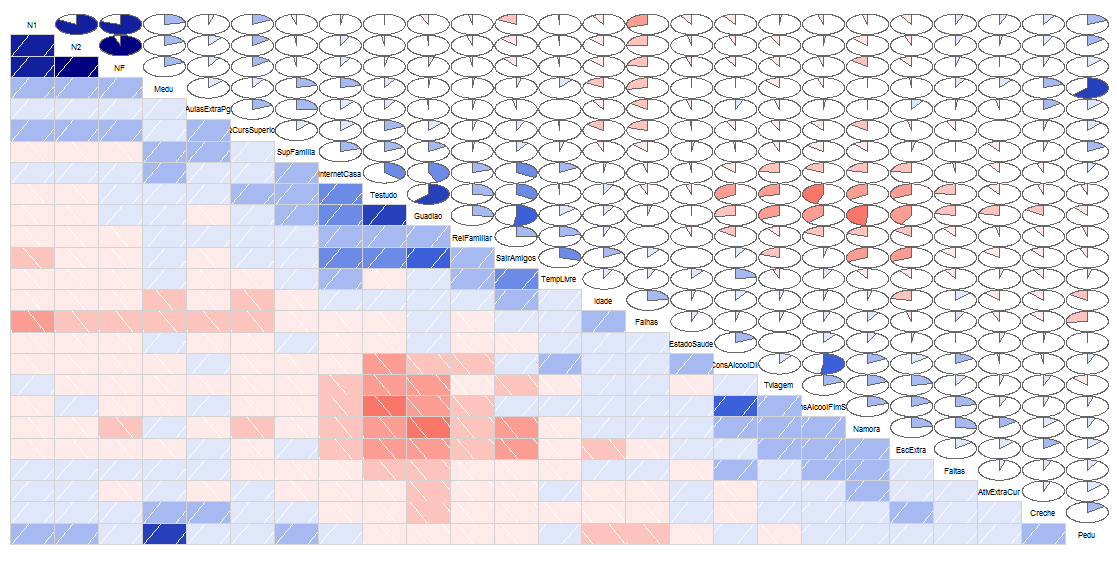
**corrgram(df)**



**# Criando um corrgram com mais parâmetros**

**corrgram(df, order = TRUE, lower.panel = panel.shade,**

**upper.panel = panel.pie, text.panel = panel.txt)**



**# Criando um histograma**

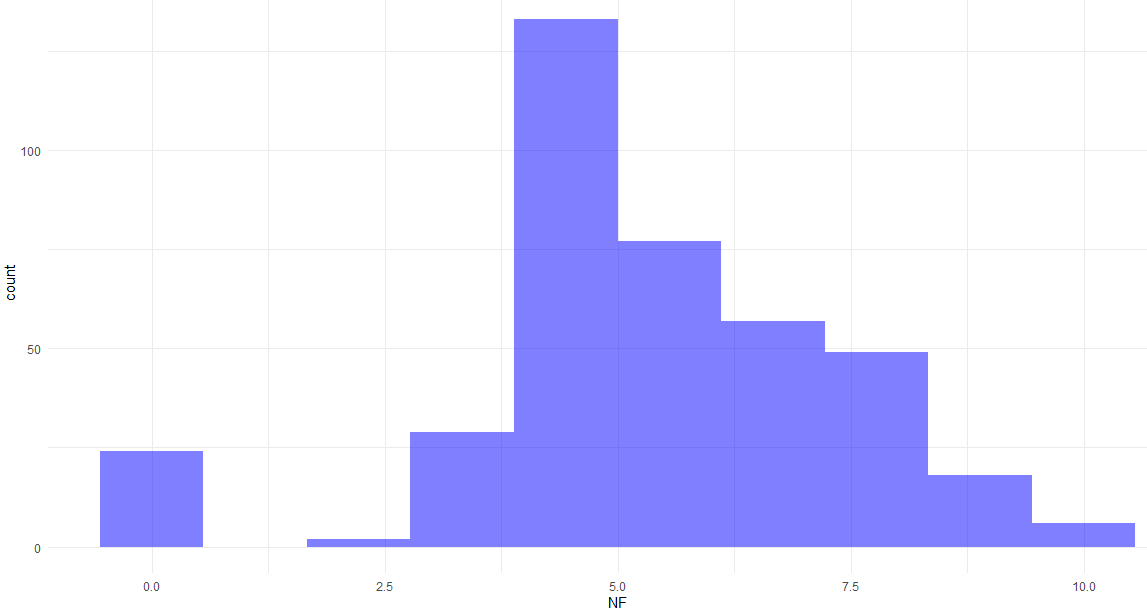
**# Variável NF - Variável target**

**ggplot(df, aes(x = NF)) +**

**geom\_histogram(bins = 10,**

**alpha = 0.5, fill = 'blue') +**

**theme\_minimal()**



**# Treinando e Interpretando o Modelo**

# Import Library

# caTools faz um split entre dados de treino e dados de teste

# de forma aleatória

install.packages("caTools")

library(caTools)

**# Criando as amostras de forma randômica**

**# reploduzir uma massa de dados igual do vídeo**

set.seed(101)

**?sample.split**

**# gerando um objeto ( amostra )**

**# sample.slit => gera dados de treino e teste**

**# 70% para treino e 30% para teste**

amostra <- sample.split(df$Idade, SplitRatio = 0.70)

**# \*\*\*\*\* Treinamos nosso modelo nos dados de treino \*\*\*\*\***

**# \*\*\*\*\* Fazemos as predições nos dados de teste \*\*\*\*\***

**# Criando dados de treino - 70% dos dados**

treino = subset(df, amostra == TRUE)

**# Criando dados de teste - 30% dos dados**

teste = subset(df, amostra == FALSE)

**# Gerando o Modelo (Usando todos os atributos)**

**# Posso gerar vários modelos**

modelo\_v1 <- lm(NF ~ ., treino) **# Todas variaveis**

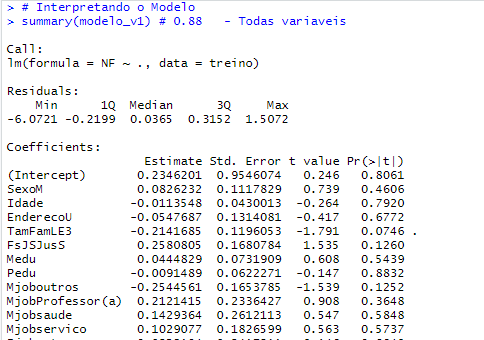
modelo\_v2 <- lm(NF ~ N2 + N1, treino) **# nota N1 e N2**

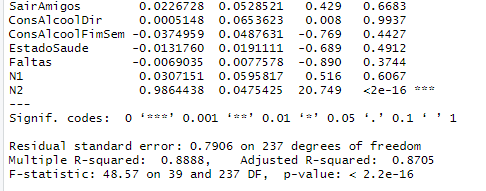
modelo\_v3 <- lm(NF ~ Faltas, treino)  **# Faltas**

modelo\_v4 <- lm(NF ~ Medu, treino) **# Grau de educação da Mãe**

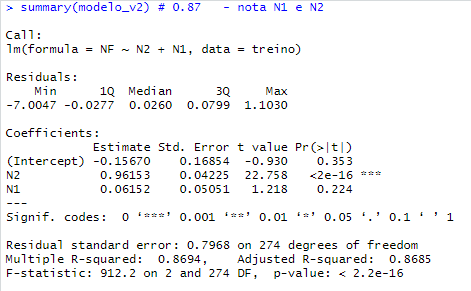
**# Interpretando o Modelo**

summary(modelo\_v1) # 0.88 - Todas variáveis

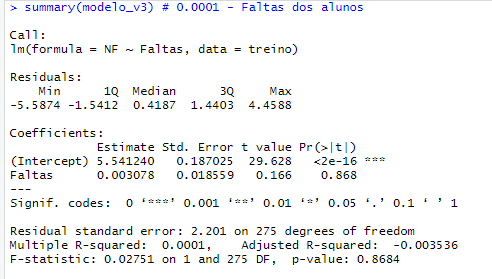




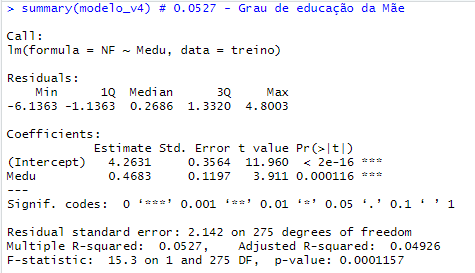
summary(modelo\_v2) # 0.87 - nota N1 e N2



summary(modelo\_v3) # 0.0001 - Faltas dos alunos



summary(modelo\_v4) # 0.0527 - Grau de educação da Mãe



**# Visualizando o Modelo e Fazendo Previsões**

**# Obtendo os resíduos (taxas de erro)**

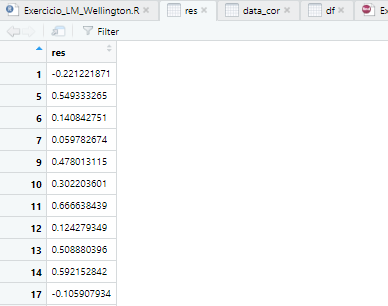
res <- residuals(modelo\_v1)

**# Convertendo o objeto para um dataframe**

res <- as.data.frame(res)

head(res)

View(res)



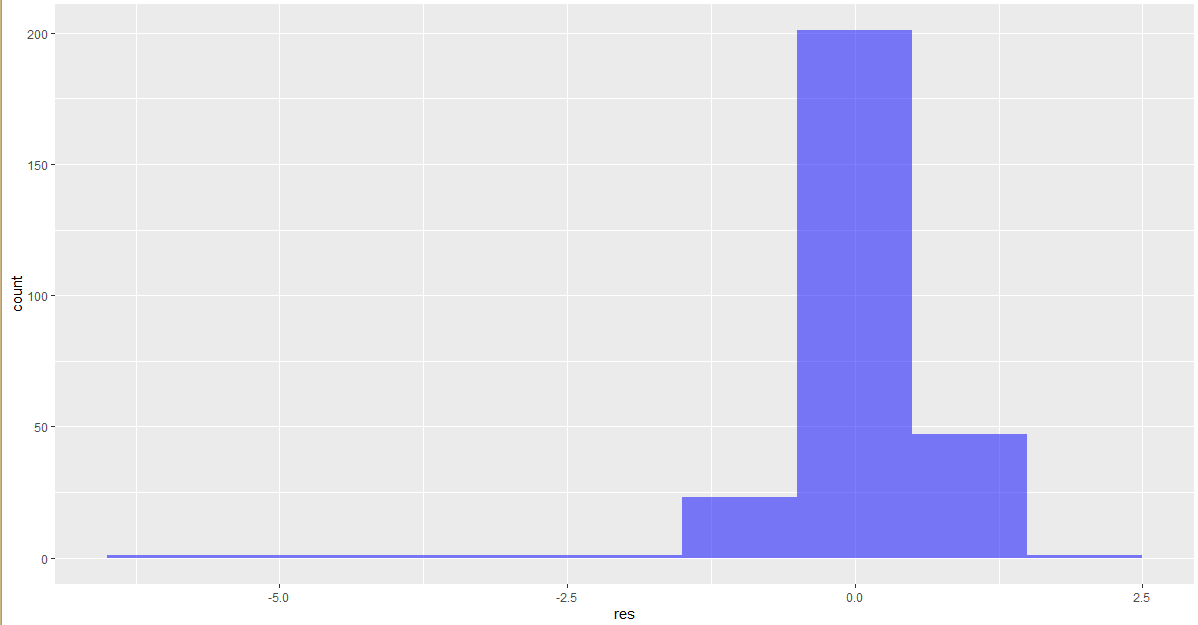
**# Histograma dos resíduos - taxa de erro**

ggplot(res, aes(res)) +

geom\_histogram(fill = 'blue',

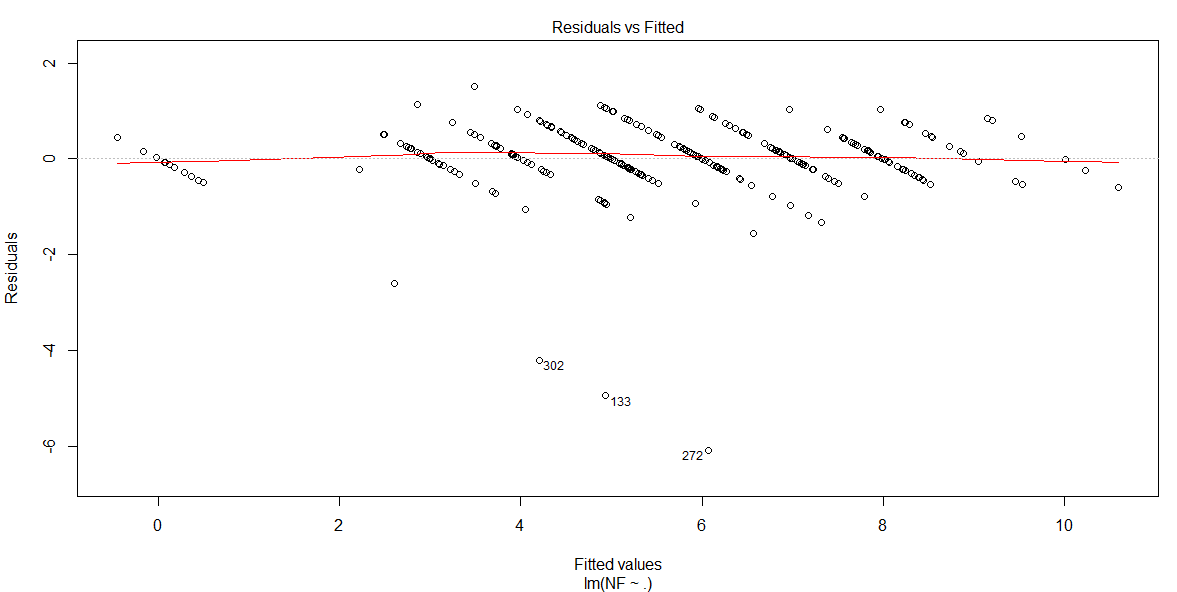
alpha = 0.5,

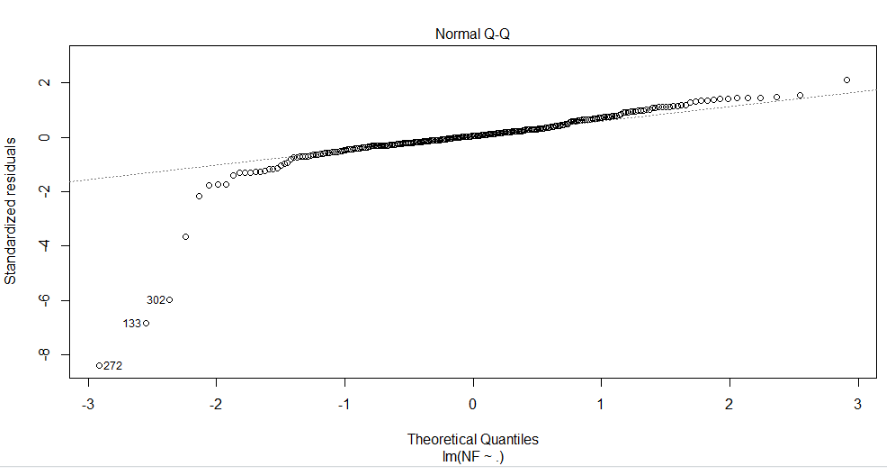
binwidth = 1)

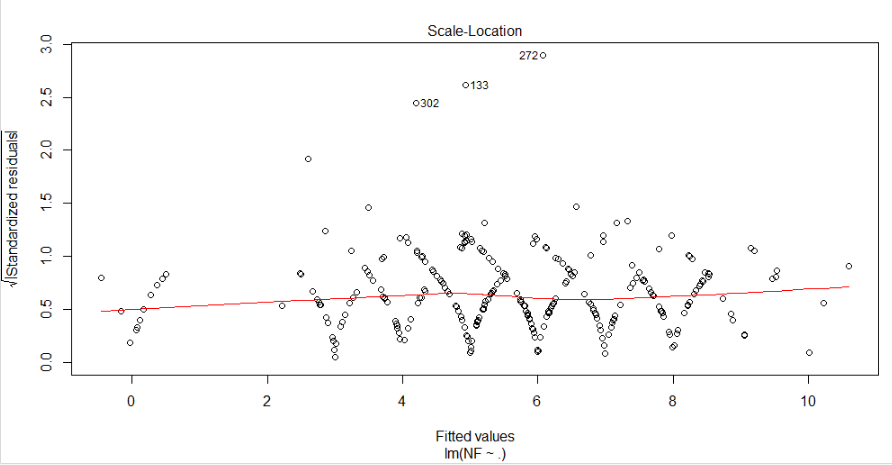


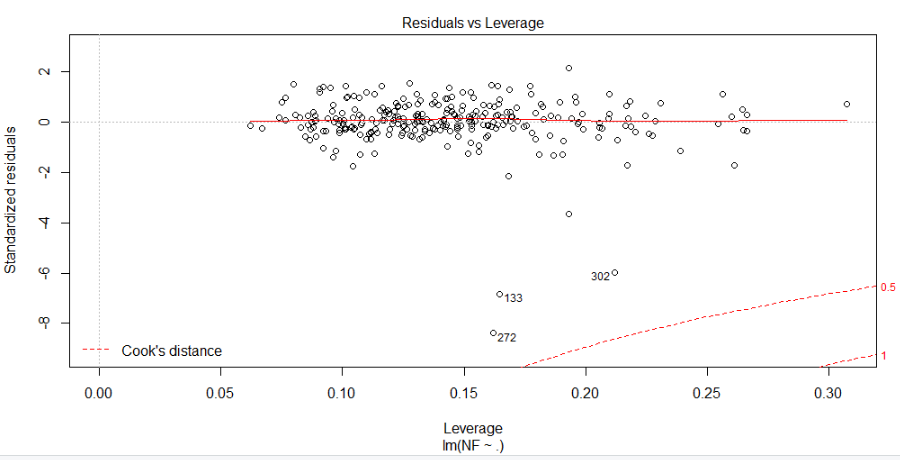
**# Plot do Modelo**

plot(modelo\_v1)









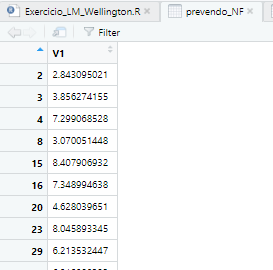
**# Fazendo as predições**

modelo\_v1 <- lm(NF ~ ., treino) # recriando o modelo (Não precisava)

prevendo\_NF <- predict(modelo\_v1, teste)

prevendo\_NF

View(prevendo\_NF)



**# Visualizando os valores previstos e observados**

**# Gerar um data.frame (resultados, Valores previsto e real)**

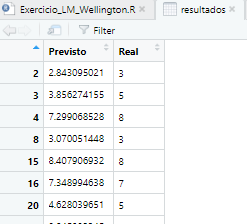
resultados <- cbind(prevendo\_NF, teste$NF) # objeto resultados

colnames(resultados) <- c('Previsto','Real') # nome das colunas

resultados <- as.data.frame(resultados) # transformando em data.frame

resultados

View(resultados)



min(resultados)



**# Função para Tratar valores negativos**

trata\_zero <- function(x){

if (x < 0){

return(0)

}else{

return(x)

}

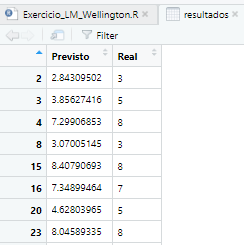
}

**# Aplicando a função para tratar valores negativos em nossa previsão**

resultados$Previsto <- sapply(resultados$Previsto, trata\_zero)

resultados$Previsto

View(resultados)



**# Calculando o erro médio**

**# Quão distantes seus valores previstos estão dos valores observados**

**# MSE**

**# MSE - o erro quadrático médio ( MSE ) ou o desvio quadrático médio ( MSD )**

**# de um estimador (de um procedimento para estimar uma quantidade não observada)**

**# mede a média dos quadrados dos erros - ou seja, a diferença quadrática**

**# média entre a estimativa valores e o valor real.**

**# MSE é uma função de risco , correspondente ao valor esperado da perda de erro**

**# ao quadrado.**

mse <- mean((resultados$Real - resultados$Previsto)^2)

print(mse)



**# RMSE**

rmse <- mse^0.5

rmse



# Calculando R Squared

SSE = sum((resultados$Previsto - resultados$Real)^2)

SST = sum((mean(df$NF) - resultados$Real)^2)

# R-Squared

# Ajuda a avaliar o nível de precisão do nosso modelo. Quanto maior, melhor, sendo 1 o valor ideal.

R2 = 1 - (SSE/SST)

R2

