

FIAP

NBA



# MBA Data Science

# MBA Data Science: Working with R



# Perfil Profissional

## Acadêmico

- MBA em Engenharia Financeira – POLI / USP.
- Pós Graduação em Análise de dados e Data Mining - FIA.
- Graduado em Ciência da Computação e Estatística.

Professor de Modelagem Estatística, Data Mining e Machine Learning dos Cursos - MBA Big Data, Data Science e Business Intelligence da Faculdade de Informática e Administração Paulista - FIAP com foco em linguagem de programação R e Python.

Professor do curso MBA Esalq/USP – Gestão de Vendas.



Prof. Edmar Caldas

## Profissional

- CEO e consultor de negócios da Inteligência Analítica com foco em consultoria: Credit Scoring, Previsão de Vendas, Fraudes entre outras.
- Certificações: IBM SPSS Modeler e SPSS Statistics.

## Objetivo da disciplina

Desenvolver competências e habilidades analíticas, além de ampliar a capacidade de observar e analisar os dados com apoio computacional da linguagem R.



# Conteúdo da disciplina



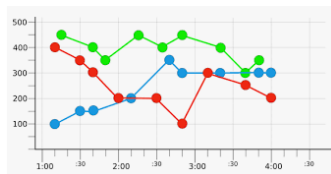
## Introdução

R, Rstudio, sintaxe.



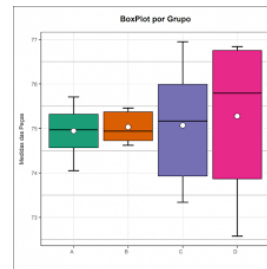
## Tipos de dados

Funções, Packages e I/O



## Dados Temporais

Dados temporais, normalização e padronização dos dados



## Gráficos

Ggplot, plotly e esquisse.

## • Introdução da disciplina

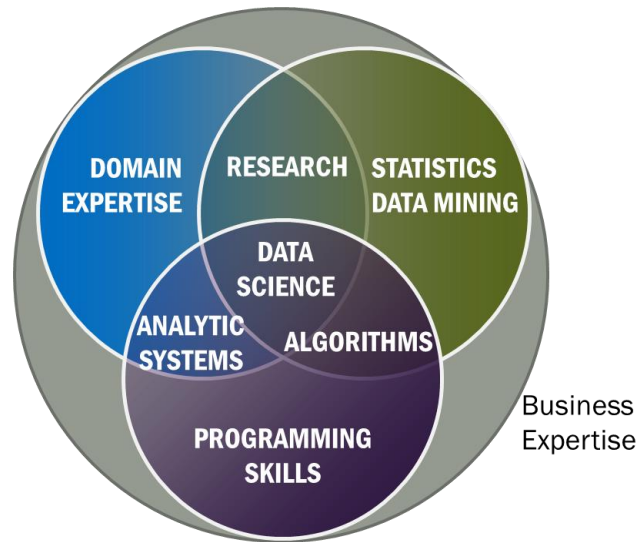


## O que é Data Science?

**Data Science (Ciência de Dados)** é a extração de conhecimento diretamente dos dados, através de um processo científico de descoberta, hipótese e análise.

Um cientista de dados é um profissional que tem conhecimentos suficientes para gerir esse processo, o que inclui conhecer as necessidades de negócio, expertise analítica no domínio de dados, habilidades matemáticas e conhecimento de programação.

**Big Data** refere-se a um grande volume de dados, cuja gestão requer escalabilidade em recursos, considerando requisitos de velocidade ou variedade de tipos de dados.





## • Software R

```
dens <- density(data, n = npts)
dx <- dens$x
dy <- dens$y
if(add == TRUE)
  plot(0., 0, main,
       ylab)
if(orientati == ysc(0))
  dx2 <- (dx - min(dx)) / (max(dx) - min(dx))
  x[1.]
  dy2 <- (dy - min(dy)) / (max(dy) - min(dy))
  y[1.]
seqbelow <- rep(y[1.], length(dx))
if(Fill == T)
  confshade(dx2, seqbelow, dy2)
```



## Características da Plataforma R - Open source

Padrão para pesquisa estatística, sendo hoje bastante ensinada nas universidades.

Foi desenvolvido por Ross Ihaka e Robert Gentleman no departamento de estatística da universidade Auckland nova zelândia. Com esforço colaborativo de várias pessoas em vários locais do mundo. Nome R provêm dos criadores.

A capacidade do R são estendidas através dos pacotes criados pela comunidade.

R é atualmente a base de soluções da IBM, Microsoft, Oracle.

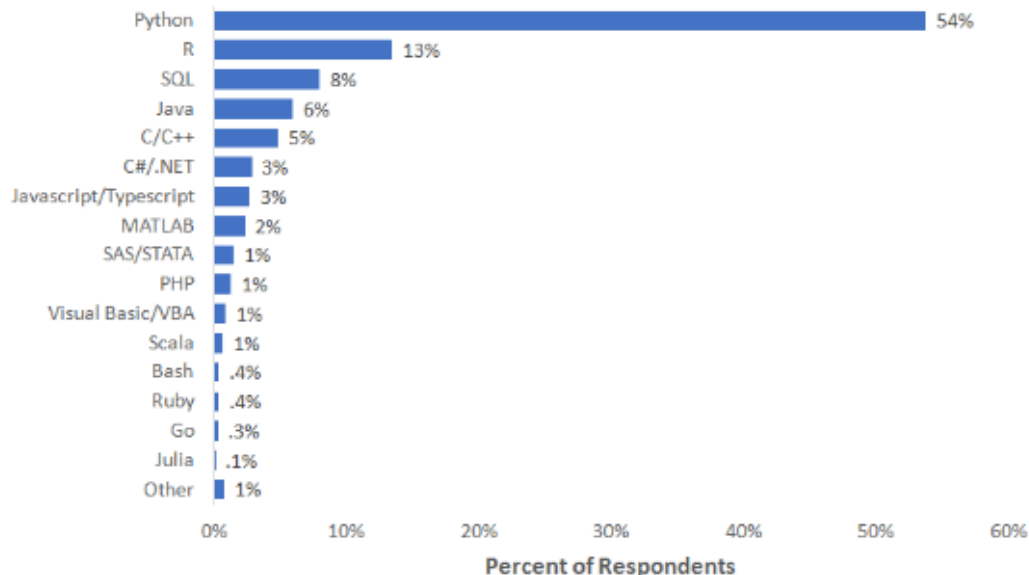


## • Evolução do R em Data Science



# Kaggle – Competições de Data Science

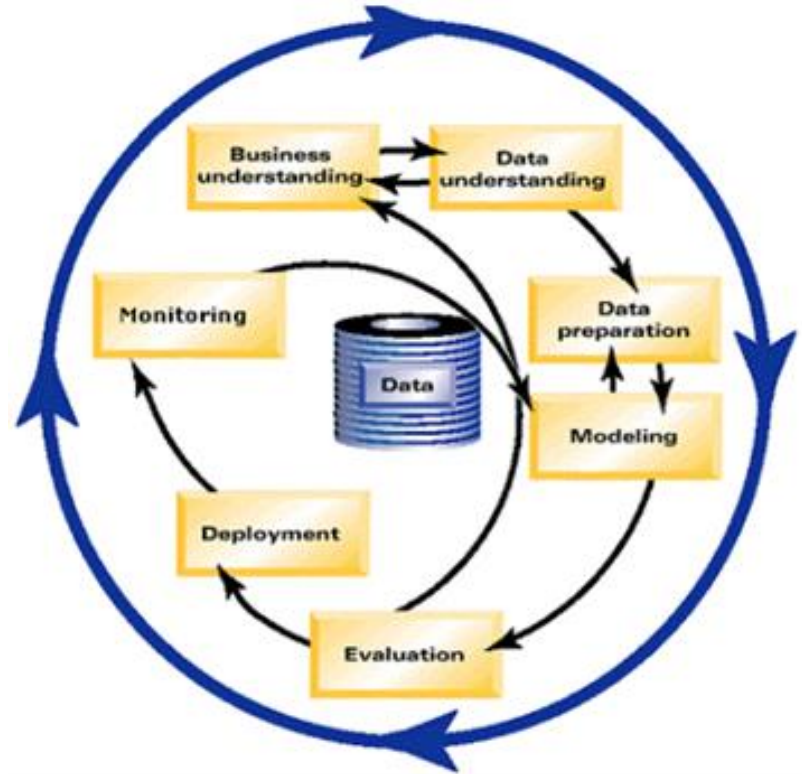
What specific programming language do you use most often?



Note: Data are from the 2018 Kaggle ML and Data Science Survey. You can learn more about the study here: <http://www.kaggle.com/kaggle/kaggle-survey-2018>.

A total of 23859 respondents completed the survey; the percentages in the graph are based on a total of 15222 respondents who provided an answer to this question.

## • Processo para Análise de Dados – Crisp - DM



●      ●      •      +      ●      □



# Aplicações de Machine Learning com uso R

## Análise Preditiva na Educação: Evasão de alunos



# Aplicações de Machine Learning com uso R

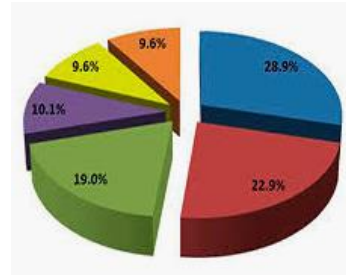
## Análise Preditiva em Finanças: Credit Score





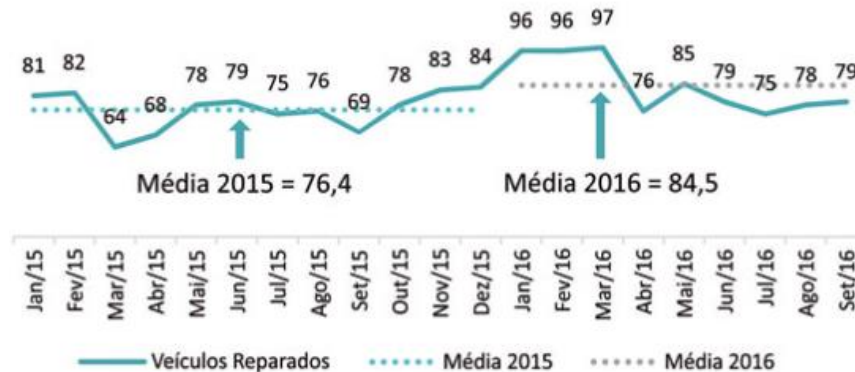
# Aplicações de Machine Learning com uso R

## Análise Preditiva em Marketing: Segmentação



## Aplicações de Machine Learning com uso R

### Análise Preditiva na Indústria: Previsão de vendas



# Aplicações de Machine Learning com uso R

**Análise Preditiva  
em Seguros:  
Renovação de  
contrato**



## • Aplicações de Machine Learning com uso R

### Análise Preditiva na Saúde: Fraude



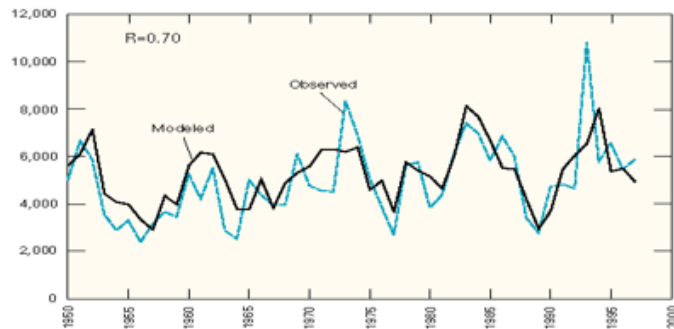
# Aplicações de Machine Learning com uso R

- Um Modelo estatístico é uma representação simplificada da realidade.

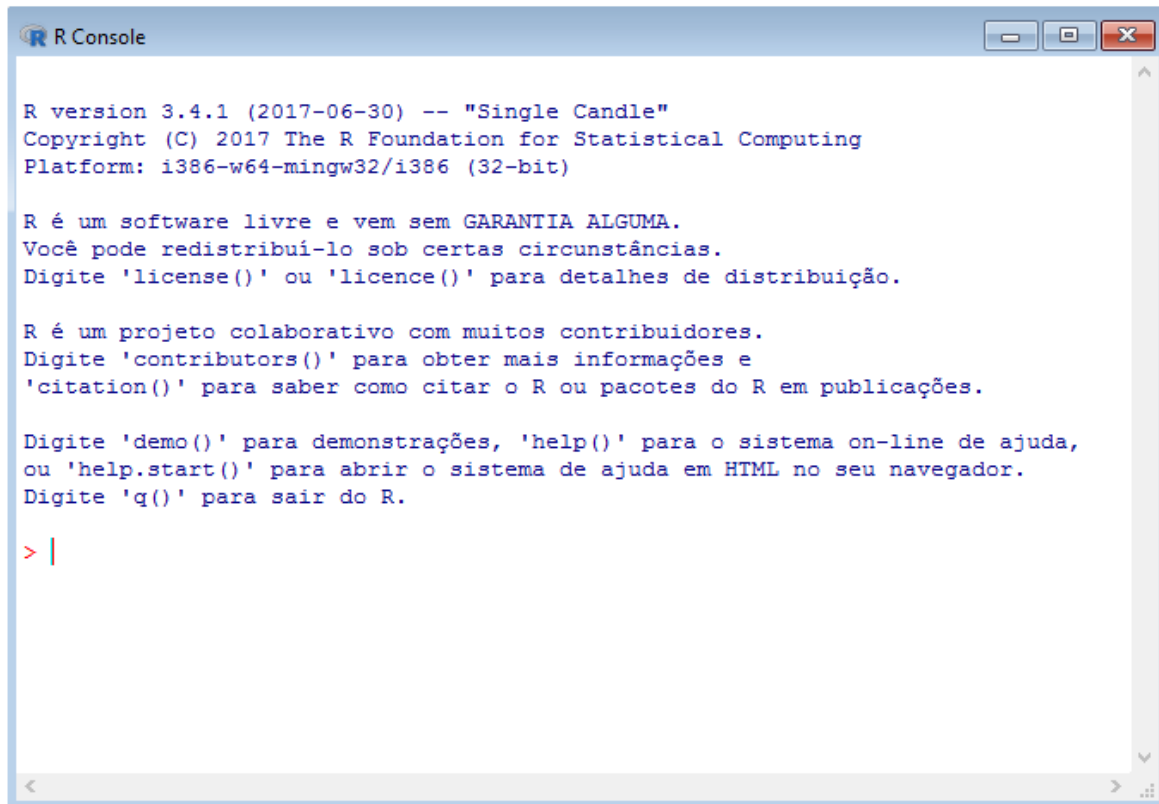


**George Box**  
1919-2013

“Todos os modelos são errados, mas alguns são úteis”



- **R Environment – Ambiente R raiz.**



```
R Console

R version 3.4.1 (2017-06-30) -- "Single Candle"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R é um software livre e vem sem GARANTIA ALGUMA.
Você pode redistribuí-lo sob certas circunstâncias.
Digite 'license()' ou 'licence()' para detalhes de distribuição.

R é um projeto colaborativo com muitos contribuidores.
Digite 'contributors()' para obter mais informações e
'citation()' para saber como citar o R ou pacotes do R em publicações.

Digite 'demo()' para demonstrações, 'help()' para o sistema on-line de ajuda,
ou 'help.start()' para abrir o sistema de ajuda em HTML no seu navegador.
Digite 'q()' para sair do R.

> |
```

The screenshot displays the RStudio desktop environment. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Tools, and Help. Below the menu is a toolbar with icons for file operations and running code. The main editor window shows a script named 'new-twitter.r' with the following R code:

```

1 #install the necessary packages
2 install.packages("ROAuth")
3 install.packages("twitter")
4 install.packages("wordcloud")
5 install.packages("tm")
6
7 library("ROAuth")
8 library("twitter")
9 library("wordcloud")
10 library("tm")
11
12 #necessary step for windows
13 download.file(url="http://curl.haxx.se/ca/cacert.pem", destfile="cacert.pem")
14
15 #to get your consumerkey and consumerSecret see the twitter documentation for instructions
16 cred <- OAuthFactory$new(consumerKey="ZqAvGdgZonpDYcmT9xMhOpjp",
17   consumerSecret="doabHL2uFn9Gyh847nwVr8ZMnTQ1XjN6A1Q89aXrCKH1oFGJo",
18   requestURL="https://api.twitter.com/oauth/request_token",
19   accessURL="https://api.twitter.com/oauth/access_token",
20   authURL="https://api.twitter.com/oauth/authorize")
21
22 #necessary step for windows
23 cred$handshake(cainfo="cacert.pem")
24 #save for later use for windows
25 save(cred, file="twitter_authentication.Rds")
26
11

```

The console window at the bottom shows the execution of the following commands and their output:

```

> string(x)
Error: could not find function "string"
> summary(x)
      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.3300  0.2121  0.2405  0.2932  0.5469  0.7968
> x<-rnorm(100)
> plot(x)
> summary(x)
      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-2.60200 -0.54400  0.12300  0.08437  0.77800  2.29800
>

```

The Environment pane on the right shows the 'Global Environment' with a 'Data' section listing several objects:

Object	Class	Dimensions	Summary
A	int	[1:10, 1:10]	0 0 1 1 1 0 1 0
ap.d	0 obs. of 1 variables		
ap.m	num[0, 0]		
b	699 obs. of 12 variables		
birthwt1	189 obs. of 10 variables		
c.dat	216 obs. of 8 variables		
clim	205 obs. of 3 variables		

The Plots pane on the right shows a scatter plot of 'x' (y-axis) versus 'Index' (x-axis). The x-axis ranges from 0 to 100, and the y-axis ranges from -2 to 2. The plot contains numerous data points scattered across the area.



# Help

The screenshot displays the R Help system interface. On the left, the R console shows the following commands and output:

```
help.start() # Ajuda geral.  
help("mean") # Ajuda sobre a função média.  
help.search("mean")  
help[  
  help {utils}  
  help.request {utils}  
  help.search {utils}  
  help.start {utils}
```

A yellow tooltip box is visible over the `help` command, containing the following text:

```
help(topic, package = NULL, lib.loc = NULL, verbose =  
getOption("verbose"), try.all.packages =  
getOption("help.try.all.packages"), help_type =  
getOption("help_type"))  
help is the primary interface to the help systems.  
Press F1 for additional help
```

On the right, the help search results for "mean" are displayed in a table:

Package	Obs.	Variables
Apertu_1	5 obs.	3 variables
Apertu_2	5 obs.	3 variables
Apertu_A	5 obs.	4 variables
Apertu_B	5 obs.	3 variables
Apertu_C	5 obs.	2 variables
Banco	512 obs.	10 variables
chave	5 obs.	6 variables
chave2	5 obs.	6 variables
masculino	278 obs.	10 variables
selecao	1 obs.	1 variable

Below the table, the R Help system interface is shown, including the "Statistical Data Analysis" logo and the "Manuals" section with links to various R documentation resources.

**Manuals**

- [An Introduction to R](#)
- [Writing R Extensions](#)
- [R Data Import/Export](#)
- [The R Language Definition](#)
- [R Installation and Administration](#)
- [R Internals](#)

**Reference**

- [Packages](#)
- [Search Engine & Keywords](#)



- **Tipo de Dados**

- Caracter (**character**): "a", "swc", "a1b", "11"
- Numérico (**numeric**): 2, 0.5
- Inteiro (**integer**): 1, 3, 2
- Lógico (**logical**): TRUE, FALSE
- Complexo (**complex**): 1+4i

## • Tipo de Estrutura

- **Vetor atômico (atomic vector)**

- `vet_num <- c(1, 2.5, 4.5)`

- **Lista (list)**

- `lista <- list(1:3, "a", c(TRUE, FALSE, TRUE), c(2.3, 5.9))`

- **Matriz (matrix)**

- `matriz <- matrix(1:6, ncol = 3, nrow = 2)`

- **Data frame**

- `df <- data.frame(x = 1:3, y = c("a", "b", "c"))`

Vamos brincar !!!!!!!!!!!!!!!!!!!!!



## Operadores Aritméticos

```
> 1+1
[1] 2

> 5*5-4
[1] 21

> 16/4*(4)
[1] 16

> 36+2*(25+(18-(5-2)*3))
[1] 104

> 2^4+2*(4^2+8-1)
[1] 62

> ((5^2-6*2^2)*3+(13-7)^2/3)/5
[1] 3
```

## Operadores Relacionais

```
> 1 <= 1
[1] TRUE

> 1 == 0.999 # SINAL DE IGUAL
[1] FALSE

> 1 != 1 # SINAL DE DIFERENTE
[1] FALSE

> a <- c(2,6,9)
> b <- c(3,7,8)
> a > b
[1] FALSE FALSE TRUE

> a == b
[1] FALSE FALSE FALSE

> a < b
[1] TRUE TRUE FALSE

> ("string teste" == "strings testes")
[1] FALSE
```

## Outros Operadores

```
> ((sqrt(16)/2)*3^2)/2*(9-2^3)
[1] 9

> -(-2)^3+(-1)^0-sqrt(25-3^2)-5^3/25
[1] 0

> #Produtos Notáveis

> a <-1
> b <-3

> a^2 + 2*a*b + b^2
[1] 16
```

- A função "substr" extrai um fragmento de caracteres. O primeiro argumento é a string em si, o segundo é a posição de índice do início do processo e a última é a posição de índice final.
- Função Paste é usado para concatenar string.

```
> "string teste"  
[1] "string teste"
```

```
> ("string teste" == "strings testes")  
[1] FALSE
```

```
> paste ("Edmar","Caldas", sep=" ")  
[1] "Edmar_Caldas"
```

```
>  
> paste ("Edmar","Caldas", sep=" ")  
[1] "Edmar Caldas"
```

```
>  
> paste ("Edmar","Caldas")  
[1] "Edmar Caldas"
```

```
> a <-substr("Data science",6,12)  
> a  
[1] "Science"
```

```
> aa <- "Data science"  
> a <-substr(aa,6,12)  
> a  
[1] "Science"
```

```
> b <- 17784899812  
> bb <-substr(b,9,9)  
> bb  
[1] "8"
```

# Funções diversas

```
> x <- c(1,2,3,4,5,6)
```

```
> mean(x)
```

```
[1] 3.5
```

```
> sum(x)
```

```
[1] 21
```

```
> min(x)
```

```
[1] 1
```

```
> max(x)
```

```
[1] 6
```

```
> prod(x)
```

```
[1] 720
```

```
> range(x)
```

```
[1] 1 6
```

```
> median(x)
```

```
[1] 3.5
```

```
> summary(x)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.00	2.25	3.50	3.50	4.75	6.00

```
> sample(1:100,10)
```

```
[1] 97 84 57 12 11 49 76 56 14 7
```

```
> sexo <-c(1,2,1,2)
```

```
> sexo
```

```
[1] 1 2 1 2
```

```
> is.numeric(sexo)
```

```
[1] TRUE
```

```
> sexo <-factor(sexo, levels = c(1,2), labels = c("Masculino","Feminino"))
```

```
> sexo
```

```
[1] Masculino Feminino Masculino Feminino
```

```
Levels: Masculino Feminino
```

```
> is.numeric(sexo)
```

```
[1] FALSE
```

```
> is.factor(sexo)
```

```
[1] TRUE
```

```
> sexo<-as.numeric(sexo)
```

```
> sexo
```

```
[1] 1 2 1 2
```

```
> is.numeric(sexo)
```

```
[1] TRUE
```

- `rbind()`: Combina os argumentos como linhas de uma matriz
- `cbind()`: Combina os argumentos como colunas de uma matriz

```
> A <-matrix(data =1:12, ncol =4)
> A
      [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
>
> A1 <-A[2,4]
> A1
[1] 11
>
> #Soma 10 ao vetor A1
> 10 + A1
[1] 21
>
> B <- matrix(data= 13:24, ncol =4)
> B
      [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

```
> BB <-B[1:3,1]
> BB
[1] 13 14 15
>
> #Juntar matrizes linhas e colunas
> cbind(A,B)
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    1    4    7   10   13   16   19   22
[2,]    2    5    8   11   14   17   20   23
[3,]    3    6    9   12   15   18   21   24
> rbind(A,B)
      [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
[4,]   13   16   19   22
[5,]   14   17   20   23
[6,]   15   18   21   24
```

```
> m<-matrix(data=1:12, nrow = 3)
> m
      [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
>
>
> m[3,] # seleciona 3 linha
[1] 3 6 9 12
>
>
> m[,3] # seleciona 3 coluna
[1] 7 8 9
>
> soma <- (m[1]+m[12])
> soma
[1] 13
```

# Listas

É uma coleção ordenada de objetos conhecidos como componentes da lista. Esses componentes não necessitam ser do mesmo tipo ou tamanho.

`rbind()`: Combina os argumentos como linhas de uma matriz

`cbind()`: Combina os argumentos como colunas de uma matriz

```
> estudantes <- list(cbind(nome=c("Edmar","Pedro"),
+                           idade=c(30,8),
+                           escolaridade=c("MBA","Primário")))
> estudantes
[[1]]
  nome   idade escolaridade
[1,] "Edmar" "30"    "MBA"
[2,] "Pedro" "8"     "Primário"
```



É uma estrutura utilizada para armazenar elementos em forma de tabela, organizados em linhas e colunas. Portanto tem duas dimensões. Na sua forma, um **data frame**, é muito semelhante a uma matriz, mas suas colunas tem sempre nomes, e podem conter dados de tipos diferentes. Um **data frame** pode ser visto como uma tabela de uma base de dados relacional, em que cada linha corresponde às propriedades (campos) a serem armazenadas para cada registro da tabela.

```
> Nome <-c("João","José","Junior")
> Idade <-c(45,35,25)
> Salario <-c(1000,2000,3000)
> Cadastral <-data.frame(Nome,Idade,salario)
> Cadastral
```

	Nome	Idade	Salario
1	João	45	1000
2	José	35	2000
3	Junior	25	3000

```
> Cadastral$filhos <-c(2,1,0)
```

```
> Cadastral$Aumento_Salarial <-c((Salario*0.06)+Salario)
```

```
> Cadastral
```

	Nome	Idade	salario	filhos	Aumento_Salarial
1	João	45	1000	2	1060
2	José	35	2000	1	2120
3	Junior	25	3000	0	3180

```
> Cadastral1 <-Cadastral[, -3]
```

```
> Cadastral1
```

	Nome	Idade	Aumento_Salarial
1	João	45	1060
2	José	35	2120
3	Junior	25	3180

```
> names(Cadastral)
```

```
[1] "Nome" "Idade" "salario"
```

```
> levels(Cadastral$Nome)
```

```
[1] "João" "José" "Junior"
```

# Package ODBC

O objetivo do pacote odbc é fornecer uma interface compatível com DBI (database interface) para drivers Open Database Connectivity (ODBC). Isso permite uma conexão eficiente e fácil de configurar a qualquer banco de dados com drivers ODBC disponíveis, incluindo SQL Server, Oracle, MySQL, PostgreSQL, SQLite e outros.

```
library(odbc)
```

```
df_con <-dbConnect(odbc::odbc(),  
  .connection_string = "Driver={SQL Server Native Client 11.0};  
  Server=DESKTOP-o6v4dv1;Database=TesteSPSS;Trusted_Connection=yes;")
```

```
df <-dbGetQuery(df_con,"Select * from Banco$")  
head(df)
```

	id	datanasc	sexo	estudo	catemp	salário	salarin	temp_ser	cartao_credito	Emprestimos
1	474	1968-11-05	Feminino	12	C	29400	NA	63	5880	2940
2	473	1937-11-25	Feminino	12	C	21450	12750	63	4290	2145
3	472	1966-02-21	Masculino	15	C	39150	15750	63	7830	3915
4	471	1966-08-03	Masculino	15	C	26400	15750	64	5280	2640
5	470	1964-01-22	Masculino	12	C	26250	15750	64	5250	2625
6	469	1964-06-01	Feminino	15	C	25200	13950	64	5040	2520

```
# Lista todas as tabelas  
dbListTables(df_con)  
[1] "Banco$"  
[2] "Consolidado_final$"  
[3] "iris"
```

```
# Lista toda as tabelas que começa com B  
dbListTables(df_con, table_name = "B%")  
[1] "Banco$" "backup_devices"
```

```
# Lista nome dos campos  
dbListFields(df_con, "Banco$")  
"id" "datanasc" "sexo"  
"salário" "salarin" "temp_ser"  
"estudo" "catemp"  
"cartao_credito" "Emprestimos"
```

## Package summarytools

O summarytools fornece um conjunto coerente de funções centradas na exploração de dados e relatórios simples.

```
View(Banco)
```

```
f_abs <-table(Banco$sexo)
f_abs
```

```
Feminino Masculino
233      278
```

```
f_rel <-prop.table(table(Banco$sexo))
f_rel
```

```
Feminino Masculino
0.4559687 0.5440313
```

```
tabela <-round(t(rbind(f_abs,f_rel)),digits=2)
tabela
```

```
      f_abs f_rel
Feminino 233  0.46
Masculino 278  0.54
```

## Package summarytools

Função	Descrição
<code>freq()</code>	Tabelas de frequência apresentando contagens, proporções, bem como informações de dados ausentes
<code>ctable()</code>	Tabulações cruzadas (frequências conjuntas) entre pares de variáveis discretas / categóricas, apresentando somas marginais, bem como linhas, colunas ou proporções totais
<code>descr()</code>	Estatísticas descritivas (univariadas) para dados numéricos, apresentando medidas comuns de tendência central e dispersão
<code>dfSummary()</code>	Extensive Data Frame Summaries apresenta informações específicas para todas as variáveis em um data frame: estatísticas univariadas e / ou distribuições de frequência, gráficos de barras ou histogramas, bem como contagens e proporções de dados missing. Muito útil para detectar anomalias e identificar tendências rapidamente

# Package summarytools - Função Freq()

```
library(summarytools)
```

```
freq(Banco$sexo)
```

Frequencies

Banco\$sexo

Type: Character

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
Feminino	233	45.60	45.60	45.60	45.60
Masculino	278	54.40	100.00	54.40	100.00
<NA>	0			0.00	100.00
Total	511	100.00	100.00	100.00	100.00

```
freq(Banco$catemp, order = "freq")
```

Frequencies

Banco\$catemp

Type: Character

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
C	390	76.32	76.32	76.32	76.32
A	94	18.40	94.72	18.40	94.72
B	27	5.28	100.00	5.28	100.00
<NA>	0			0.00	100.00
Total	511	100.00	100.00	100.00	100.00

## Package summarytools - Função Freq()

```
freq(Banco$catemp, order = "freq", round.digits = 1)
```

Frequencies

Banco\$catemp

Type: Character

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
C	390	76.3	76.3	76.3	76.3
A	94	18.4	94.7	18.4	94.7
B	27	5.3	100.0	5.3	100.0
<NA>	0			0.0	100.0
Total	511	100.0	100.0	100.0	100.0

```
freq(Banco$catemp, totals=FALSE, cumul=FALSE, headings = FALSE)
```

	Freq	% Valid	% Total
A	94	18.40	18.40
B	27	5.28	5.28
C	390	76.32	76.32
<NA>	0		0.00

# Package summarytools - Função Freq()

```
freq(Banco$catemp, report.nas = F)
```

Frequencies

Banco\$catemp

Type: Character

	Freq	%	% Cum.
A	94	18.40	18.40
B	27	5.28	23.68
C	390	76.32	100.00
Total	511	100.00	100.00

# Package summarytools - Função Freq()

```
with(Banco, by(catemp, sexo, freq))
```

```
sexo: Feminino
```

```
Frequencies
```

```
Banco$catemp
```

```
Type: Character
```

```
Group: sexo = Feminino
```

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
A	12	5.15	5.15	5.15	5.15
B	1	0.43	5.58	0.43	5.58
C	220	94.42	100.00	94.42	100.00
<NA>	0			0.00	100.00
Total	233	100.00	100.00	100.00	100.00

```
sexo: Masculino
```

```
Frequencies
```

```
Banco$catemp
```

```
Type: Character
```

```
Group: sexo = Masculino
```

	Freq	% Valid	% Valid Cum.	% Total	% Total Cum.
A	82	29.50	29.50	29.50	29.50
B	26	9.35	38.85	9.35	38.85
C	170	61.15	100.00	61.15	100.00
<NA>	0			0.00	100.00
Total	278	100.00	100.00	100.00	100.00



## Package summarytools - Função ctable()

```
ctable(x=Banco$catemp, y=Banco$sexo, prop = 'r', round.digits = 2, justify = 'center')
Cross-Tabulation, Row Proportions
catemp * sexo
Data Frame: Banco
```

	sexo	Feminino	Masculino	Total
catemp				
A		12 (12.77%)	82 (87.23%)	94 (100.00%)
B		1 ( 3.70%)	26 (96.30%)	27 (100.00%)
C		220 (56.41%)	170 (43.59%)	390 (100.00%)
Total		233 (45.60%)	278 (54.40%)	511 (100.00%)

## Package Gmodels: Tabela cruzada

- É uma técnica que permite entender o relacionamento entre duas variáveis categóricas;
- É utilizado praticamente em todas as áreas de pesquisa;
- É uma tabela de frequência de contagens;
- Permite comparações entre grupo;
- Pode ser usado de uma forma descritiva;
- Também pode ser usado para tirar conclusões sobre a população com base nos dados da amostra.

Função Table: Usa os fatores de classificação para criar uma tabela de contagens.

```
> table(sexo)
```

sexo	
Feminino	234
Masculino	278

```
> table(sexo,catemp)
```

	catemp		
sexo	A	B	C
Feminino	12	1	221
Masculino	82	26	170

# Package Gmodels

- Package gmodels: É uma ferramenta de programação para montagem de modelos.
- Função CrossTable: É uma função para plotagem/ montagem de uma tabulação cruzada com percentuais.

```
installed.packages("gmodels")
library(gmodels)
```

```
CrossTable(sexo,catemp)
CrossTable(Banco$sexo,Banco$catemp, chisq = TRUE)
```

sexo	catemp			
	A	B	C	Row Total
Feminino	12	1	221	234
	22.313	10.421	10.013	
	0.051	0.004	0.944	0.457
	0.128	0.037	0.565	
	0.023	0.002	0.432	
Masculino	82	26	170	278
	18.781	8.772	8.428	
	0.295	0.094	0.612	0.543
	0.872	0.963	0.435	
	0.160	0.051	0.332	
Column Total	94	27	391	512
	0.184	0.053	0.764	

Statistics for All Table Factors

Pearson's Chi-squared test

Chi^2 = 78.72816    d.f. = 2    p = 8.024121e-18

# Package dplyr



## Hadley Wickham

Estatístico

Hadley Wickham é um estatístico da Nova Zelândia que atualmente é cientista-chefe do RStudio e professor-adjunto de estatística da Universidade de Auckland, da Universidade de Stanford e da Rice University.

Fornecer funções simples que correspondem às tarefas mais comuns de manipulação de dados, para ajudá-lo a traduzir seus pensamentos em código. O dplyr também suporta bancos de dados através do pacote dbplyr, consulte `install.packages("dbplyr")`.

Dplyr visa fornecer uma função para cada verbo básico de manipulação de dados:

- `filter()` para selecionar casos com base em seus valores.
- `arrange()` para reordenar os casos.
- `select()` e `rename()` selecionar variáveis com base em seus nomes.
- `mutate()` e `transmute()` adicionar novas variáveis que são funções de variáveis existentes.
- `summarise()` para condensar vários valores para um único valor.
- `sample_n()` e `sample_frac()` para tirar amostras aleatórias.

## Package dplyr – Summarise: Compras

```
library(dplyr)
library(rfm)
library(lubridate)

agregar <- summarise(group_by(Compras, id),
                      ticket_medio = mean(Valor_Compra),
                      data_max      = max(DT_Compra),
                      count         = n())
```

agregar

A tibble: 50 x 4

	id	ticket_medio	data_max	count
	<dbl>	<dbl>	<dtm>	<int>
1	1	1572.	1997-01-01 00:00:00	4
2	2	1399.	1996-02-02 00:00:00	3
3	3	1473.	1997-02-12 00:00:00	3
4	4	1379.	1997-01-03 00:00:00	5
5	5	1444.	1996-10-07 00:00:00	4
6	6	1430	1996-11-20 00:00:00	3
7	7	1741	1997-01-04 00:00:00	1
8	8	1345.	1997-01-01 00:00:00	4
9	9	1266	1997-01-03 00:00:00	3
10	10	1375	1996-03-03 00:00:00	2

... with 40 more rows

# Package dplyr – Summarise: Compras

```
# lubridate
agregar$data_max <- as_date(agregar$data_max)
agregar$data_atual <- as_date(Sys.Date())

names(agregar)
[1] "id"          "ticket_medio" "data_max"     "count"       "data_atual"

# rfm
df_rfm <- rfm_table_order(agregar, id, data_max, ticket_medio, agregar$data_atual)
df_rfm
# A tibble: 50 x 9
  customer_id date_most_recent recency_days transaction_count amount recency_score frequency_score monetary_score rfm_score
  <dbl> <date> <dbl> <dbl> <dbl> <int> <int> <int> <dbl>
1 1 1997-01-01 9258 1 1572. 4 1 5 415
2 2 1996-02-02 9592 1 1399. 1 1 2 112
3 3 1997-02-12 9216 1 1473. 5 1 4 514
4 4 1997-01-03 9256 1 1379. 5 1 2 512
5 5 1996-10-07 9344 1 1444. 2 1 4 214
6 6 1996-11-20 9300 1 1430 3 1 3 313
7 7 1997-01-04 9255 1 1741 5 1 5 515
8 8 1997-01-01 9258 1 1345. 4 1 2 412
9 9 1997-01-03 9256 1 1266 5 1 1 511
10 10 1996-03-03 9562 1 1375 1 1 2 112
# ... with 40 more rows
```

# Package dplyr – Summarise: Compras

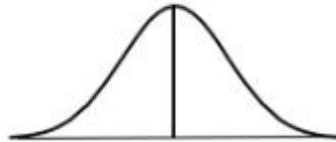
```
select
# A tibble: 25 x 9
  customer_id date_most_recent recency_days transaction_count amount recency_score frequency_score monetary_score rfm_score
  <dbl> <date> <dbl> <dbl> <dbl> <int> <int> <int> <dbl>
1 1 1997-01-01 9258 1 1572. 4 1 5 415
2 3 1997-02-12 9216 1 1473. 5 1 4 514
3 4 1997-01-03 9256 1 1379. 5 1 2 512
4 7 1997-01-04 9255 1 1741 5 1 5 515
5 8 1997-01-01 9258 1 1345. 4 1 2 412
6 9 1997-01-03 9256 1 1266 5 1 1 511
7 11 1997-01-01 9258 1 1770 4 1 5 415
8 13 1997-01-01 9258 1 1741 4 1 5 415
9 14 1997-01-03 9256 1 1397. 5 1 2 512
10 19 1997-01-04 9255 1 1433. 5 1 3 513
# ... with 15 more rows
min(select$rfm_score)
[1] 412
max(select$rfm_score)
[1] 515
```

## Função scale: Padronizar variáveis

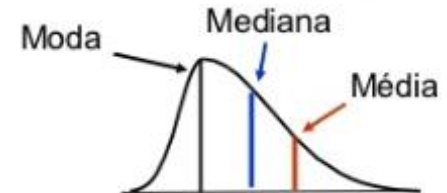
Em estatística, padronização é o processo de colocar **variáveis** diferentes na mesma escala. Esse processo permite comparar pontuações entre diferentes tipos de **variáveis**. Normalmente, para **padronizar variáveis**, você calcula a média e o desvio padrão para uma **variável**.

$$z = \frac{x - \mu}{\sigma}$$

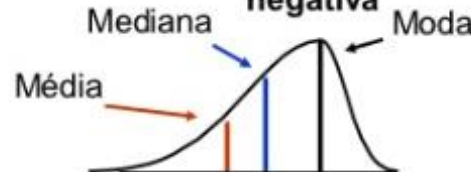
**Distribuição Simétrica**  
Média = Mediana = Moda



**Assimetria à direita ou positiva**



**Assimetria à esquerda ou negativa**





## Função scale: Padronizar variáveis

```
options(scipen = 100)
options(digits=3)

hist(Banco$salário)

media <-mean(Banco$salário)
sd     <-sd  (Banco$salário)
z_salario <-(Banco$salário - media) / sd
Banco$z_salario <- z_salario

mean(Banco$z_salario)
[1] 0.0000000000000000032
sd(Banco$z_salario)
[1] 1

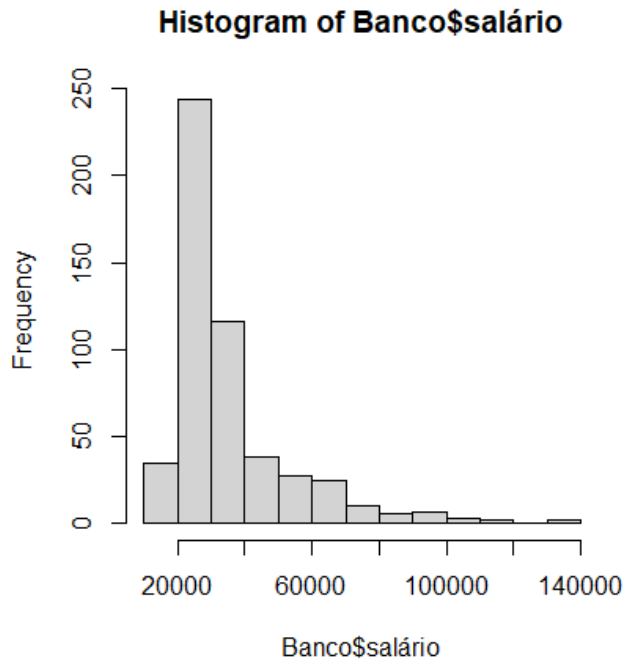
hist(Banco$z_salario)
```

Função scale – padroniza para média 0 e desvio padrão 1.

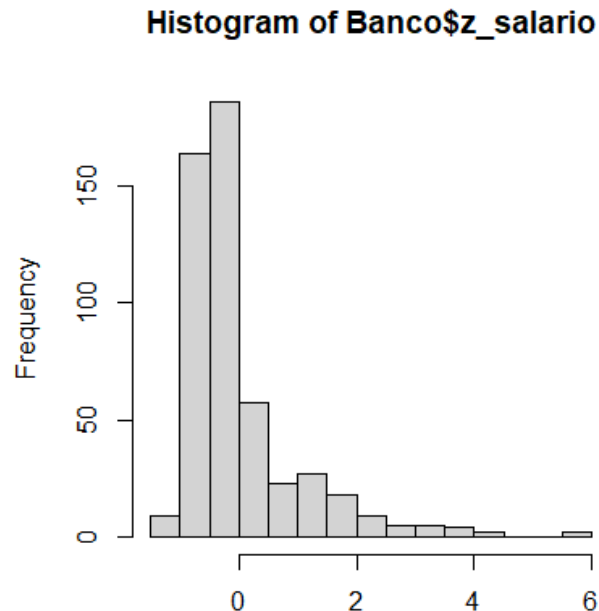
```
Banco$zz_salario <-scale(Banco$salário)
```

# Função scale: Padronizar variáveis

```
hist(Banco$salário)
```



```
hist(Banco$z_salario)
```



## Função Apply

A família Apply representa um conjunto de funções básicas do R que permite realizar operações sobre os dados contidos nas várias estruturas disponíveis (vetor, data frame, listas). O valor 1 na função se refere a linha e, o 2 a coluna. Consulte `??base::apply`.

Podemos usar diretamente a função `rowSums`, que em geral é um pouco mais rápido do que o `apply`

```
> Banco$soma<-apply(Banco[,9:10],1,sum)
>
> Banco$soma1 <-c(cartao_credito+Emprestimos)
>
> Banco$soma2 <-rowSums(Banco[,9:10])
>
> Banco
```

```
# A tibble: 512 x 13
```

	id	datanasc	sexo	estudo	catemp	salário	salarin	temp_ser	cartao_credito	Emprestimos	soma	soma1	soma2
	<dbl>	<dtm>	<chr>	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	474.	1968-11-05 00:00:00	Feminino	12.	C	29400.	NA	63.	5880.	2940.	8820.	8820.	8820.
2	473.	1937-11-25 00:00:00	Feminino	12.	C	21450.	12750.	63.	4290.	2145.	6435.	6435.	6435.
3	472.	1966-02-21 00:00:00	Masculino	15.	C	39150.	15750.	63.	7830.	3915.	11745.	11745.	11745.
4	471.	1966-08-03 00:00:00	Masculino	15.	C	26400.	15750.	64.	5280.	2640.	7920.	7920.	7920.
5	470.	1964-01-22 00:00:00	Masculino	12.	C	26250.	15750.	64.	5250.	2625.	7875.	7875.	7875.
6	469.	1964-06-01 00:00:00	Feminino	15.	C	25200.	13950.	64.	5040.	2520.	7560.	7560.	7560.
7	468.	1965-11-28 00:00:00	Feminino	16.	A	55750.	19980.	64.	11150.	5575.	16725.	16725.	16725.
8	467.	1967-08-18 00:00:00	Feminino	16.	C	32850.	19500.	64.	6570.	3285.	9855.	9855.	9855.
9	466.	1948-06-15 00:00:00	Feminino	12.	C	23400.	13500.	64.	4680.	2340.	7020.	7020.	7020.
10	465.	1962-07-20 00:00:00	Masculino	12.	C	33900.	16500.	64.	6780.	3390.	10170.	10170.	10170.

```
# ... with 502 more rows
```

## Função na.rm

Função na.rm: Calcula a média considerando apenas os dados existentes, ignora os dados missing/faltantes. Se estiver valor missing vai retornar NA. NA é caracterizado como observação perdida, faltante (missing value). NA é o NULL SQL.

```
> mean(salarin)
[1] NA
> mean(salarin, na.rm = TRUE)
[1] 17179.56
> round(mean(salarin, na.rm= TRUE))
[1] 17180
```

A opção by pode ser usado como um relatório, a primeira variável é usada para calcular a estatística e a segunda para criar subgrupos.

```
> by(Banco$salário, Banco$sexo, mean)
Banco$sexo: Feminino
[1] 26160.92
-----
Banco$sexo: Masculino
[1] 42434.73
```

# Programação Estrutural

```
# Exemplo com For
```

```
Banco$media <- (cartao_credito + Emprestimos) / 2
```

```
Banco
```

```
Banco$classe <- NA
```

```
Banco
```

```
attach(Banco)
```

```
media <- as.numeric(media)
```

```
min(media)
```

```
max(media)
```

```
for (i in 1:nrow(Banco)){
```

```
  if(Banco[i,"media"] >= 10000){
```

```
    Banco[i, "classe"] <- "classe A"
```

```
  } else if (Banco[i,"media"] < 10000 & Banco[i,"media"] >= 5000){
```

```
    Banco[i,"classe"] <- "classe B"
```

```
  } else{
```

```
    Banco[i,"classe"] <- "classe C"
```

```
  }
```

```
}
```

```
Banco
```

```
table(Banco$classe)
```

```
# Exemplo com ifelse
```

```
Banco$resultado <- ifelse (estudo > 10, "Doutorado", "Mestrado")
```

# Programação Estrutural

id	datanasc	sexo	estudo	catemp	salário	salarin	temp_ser	cartao_credito	Emprestimos	media	classe	resultado
474	1968-11-05	Feminino	12	C	29400	NA	63	5880	2940.0	4410.00	classe C	Doutorado
473	1937-11-25	Feminino	12	C	21450	12750	63	4290	2145.0	3217.50	classe C	Doutorado
472	1966-02-21	Masculino	15	C	39150	15750	63	7830	3915.0	5872.50	classe B	Doutorado
471	1966-08-03	Masculino	15	C	26400	15750	64	5280	2640.0	3960.00	classe C	Doutorado
470	1964-01-22	Masculino	12	C	26250	15750	64	5250	2625.0	3937.50	classe C	Doutorado
469	1964-06-01	Feminino	15	C	25200	13950	64	5040	2520.0	3780.00	classe C	Doutorado
468	1965-11-28	Feminino	16	A	55750	19980	64	11150	5575.0	8362.50	classe B	Doutorado
467	1967-08-18	Feminino	16	C	32850	19500	64	6570	3285.0	4927.50	classe C	Doutorado
466	1948-06-15	Feminino	12	C	23400	13500	64	4680	2340.0	3510.00	classe C	Doutorado
465	1962-07-20	Masculino	12	C	33900	16500	64	6780	3390.0	5085.00	classe B	Doutorado
464	1962-03-20	Masculino	19	A	47550	33000	64	9510	4755.0	7132.50	classe B	Doutorado
463	1934-10-15	Feminino	15	C	20700	14250	65	4140	2070.0	3105.00	classe C	Doutorado
462	1963-10-18	Feminino	16	A	34410	19500	65	6882	3441.0	5161.50	classe B	Doutorado
461	1943-11-08	Feminino	8	C	21600	13500	65	4320	2160.0	3240.00	classe C	Mestrado
460	1969-08-12	Feminino	12	C	22500	12750	65	4500	2250.0	3375.00	classe C	Doutorado
459	1971-02-10	Feminino	12	C	21750	11250	65	4350	2175.0	3262.50	classe C	Doutorado
458	1965-07-06	Masculino	19	A	61875	28740	65	12375	6187.5	9281.25	classe B	Doutorado
457	1968-05-27	Masculino	15	C	31650	14250	65	6330	3165.0	4747.50	classe C	Doutorado
456	1959-10-17	Masculino	19	A	75000	42510	65	15000	7500.0	11250.00	classe A	Doutorado
455	1964-01-17	Masculino	16	A	43650	19500	65	8730	4365.0	6547.50	classe B	Doutorado
454	1965-07-28	Masculino	19	A	90625	31250	65	18125	9062.5	13593.75	classe A	Doutorado

## Package SQLDF

R + SQL = Package sqldf - Fornece uma maneira fácil de executar as seleções de SQL em dados no R. Lê um arquivo em R filtrando-o com uma declaração sql. Somente a parte filtrada é processada pelo R. Portanto até arquivos muito maiores podem ser acomodados.

## Data Analytics with R and SQL Server



# Package SQLDF

```
install.packages("sqldf")
```

```
library(sqldf)
```

```
> masculino <-sqldf("Select * from Banco where sexo = 'Masculino'")
```

```
>
```

```
> head(masculino)
```

	id		datanasc	sexo	estudo	catemp	salário	salarin	temp_ser	cartao_credito	Emprestimos
1	472	1966-02-20	22:00:00	Masculino	15	C	39150	15750	63	7830	3915.0
2	471	1966-08-02	21:00:00	Masculino	15	C	26400	15750	64	5280	2640.0
3	470	1964-01-21	22:00:00	Masculino	12	C	26250	15750	64	5250	2625.0
4	465	1962-07-19	21:00:00	Masculino	12	C	33900	16500	64	6780	3390.0
5	464	1962-03-19	21:00:00	Masculino	19	A	47550	33000	64	9510	4755.0
6	458	1965-07-05	21:00:00	Masculino	19	A	61875	28740	65	12375	6187.5

```
> tail(masculino)
```

	id		datanasc	sexo	estudo	catemp	salário	salarin	temp_ser	cartao_credito	Emprestimos
273	26	1966-11-07	22:00:00	Masculino	15	C	31050	12600	96	6210	3105
274	22	1940-09-23	21:00:00	Masculino	12	C	21750	12750	97	4350	2175
275	19	1962-08-18	21:00:00	Masculino	12	C	42300	14250	97	8460	4230
276	18	1956-03-19	21:00:00	Masculino	16	A	103750	27510	97	20750	10375
277	17	1962-07-17	21:00:00	Masculino	15	C	46000	14250	97	9200	4600
278	16	1964-11-16	21:00:00	Masculino	12	C	40800	15000	97	8160	4080

```
> selecao <-sqldf("Select avg(salário) from Banco")
```

```
> selecao
```

	avg(salário)
1	34997.09



# Package SQLDF

```
count <- sqldf("select count(id) from Banco")
```

```
count
count(id)
1      511
```

```
group_by <- sqldf("select count(id),catemp from Banco group by catemp")
```

```
group_by
count(id) catemp
1      94      A
2      27      B
3     390      C
```

```
table(Banco$catemp)
```

```
  A  B  C
94 27 390
```

# Package Ggplot

```

attach(Banco)

# Gráfico Simples
barplot(table(Banco$sexo))

#####

# Gráfico Colorido
barplot(prop.table(table(Banco$sexo))=100,
        col=c("blue", "red"))
        title("Tabela de Frequência", xlab = "Sexo", ylab="%")

prop.table(table(sexo))

#####

# Gráfico package Plotly - Gráficos interativos com qualidade em publicação.
install.packages("plotly")
library(plotly)

barras <-plot_ly(x=Banco$sexo, y=Banco$salário, type = "bar")
barras

#####

# Gráfico Package ggplot.
install.packages("ggplot2")
library(ggplot2)

grafico_barras <-ggplot(Banco, aes(x=categ, fill=sexo))+
  geom_bar(position = 'dodge')+
  xlab("Grupos definido por sexo")+
  ylab("Numeros de clientes")+
  ggtitle("Gráfico de Barra")

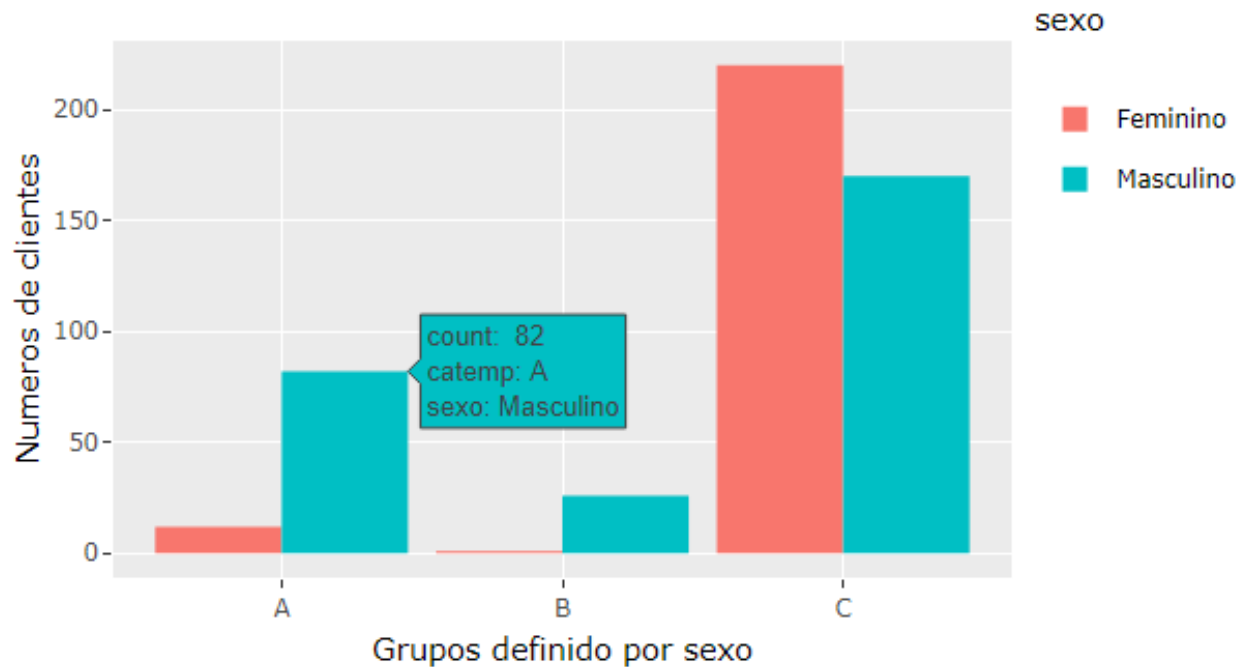
grafico_barras

# Gráfico Package ggplot com plotly.
grafico_barras1 <-ggplotly(grafico_barras)
grafico_barras1

```

# Package Ggplot

Gráfico de Barra



# Package Ggplot

```
#####

#Gráfico de Pizza

#####

# Gráfico simples
pizza <- pie(table(Banco$sexo))

# Gráfico 3D
install.packages("plotrix")
library(plotrix)

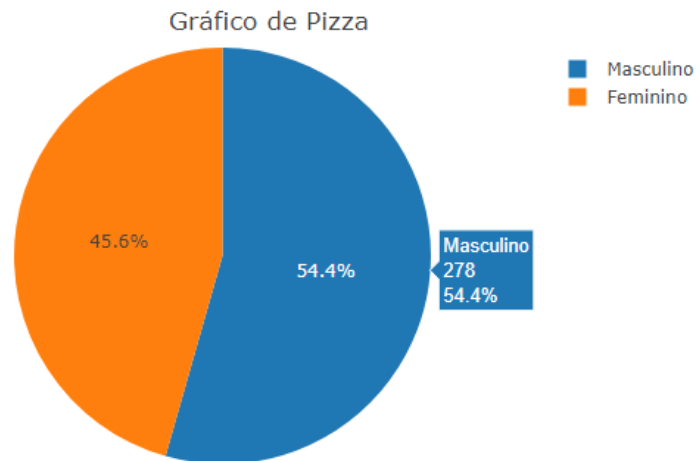
pie <- pie3D(table(Banco$sexo))

# Gráfico colorido e interativo

grafico_pie <- table(Banco$sexo)
grafico_pie1 <- as.data.frame(grafico_pie)
grafico_pie1

grafico_pie2 <- plot_ly(grafico_pie1,
  labels = ~Var1,
  values = ~Freq,
  type = 'pie') %>%
  layout(title = "Gráfico de Pizza")

grafico_pie2
```



# Package Ggplot

```
#####
```

```
# Gráfico de Histograma
```

```
#####
```

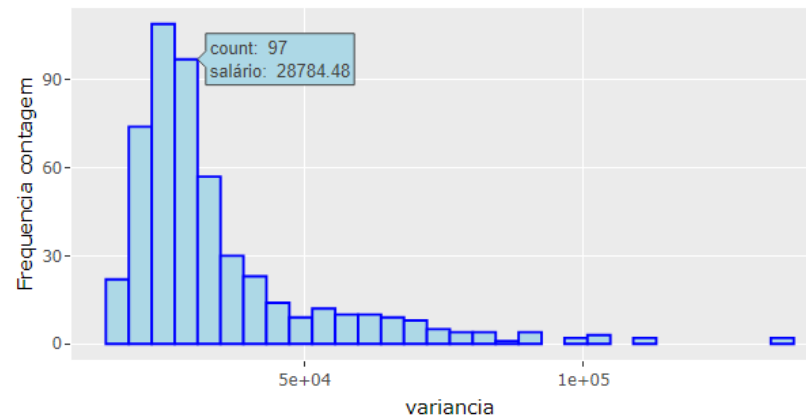
```
# Gráfico simples
hist(Banco$salário)
```

```
# Gráfico colorido pacote ggplot
histograma <-ggplot(Banco, aes(x=salário))+
  geom_histogram(color="blue", fill="lightblue", bins = 30)+
  xlab("variancia")+
  ylab("Frequencia contagem")+
  ggtitle("Gráfico_Histogrma")+
  theme()
```

```
histograma
```

```
# Grafico ggplot com plotly
ggplotly(histograma)
```

Gráfico\_Histogrma



# Package Ggplot

```
#####

# Gráfico de dispersão

#####

#Gráfico simples
plot(Banco$salário~Banco$estudo)

#Gráfico ggplot
dispersao <-ggplot(Banco, aes(x=estudo, y=salário, color=sexo))+
geom_point()

dispersao
ggplotly(dispersao)

#Gráfico ggplot com função face_wrap
dispersao1 <-ggplot(Banco, aes(x=estudo, y=salário))+
  geom_point()+facet_wrap(~sexo)

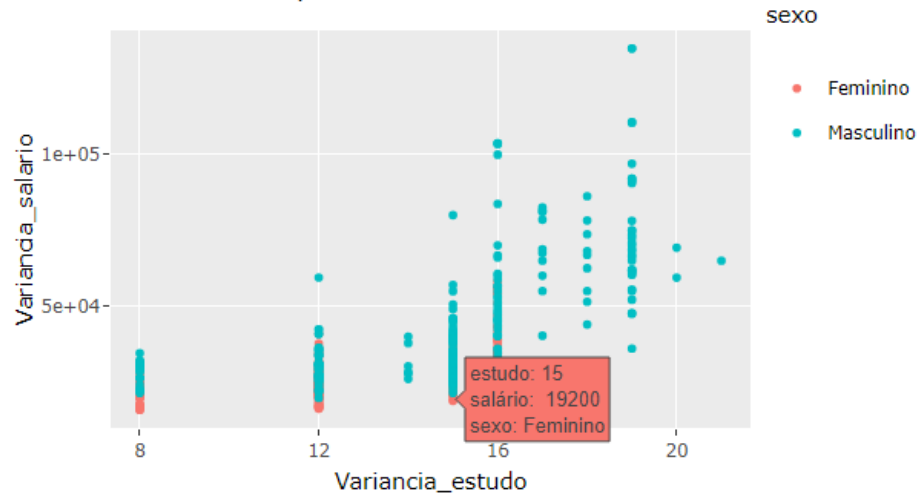
dispersao1

#Gráfico ggplot com legenda
dispersao2 <-ggplot(Banco, aes(x=estudo, y=salário, color=sexo))+
  geom_point(size=1)+
  xlab("variancia_estudo")+
  ylab("variancia_salario")+
  ggtitle("Gráfico de dispersao")

dispersao2
ggplotly(dispersao2)

#Gráfico ggplot com plotly
plot_ly(x=Banco$estudo, y=Banco$salário, color=Banco$sexo)
```

Gráfico de dispersao



# Package Ggplot

```
#####
```

```
# Gráfico BoxPlot
```

```
#####
```

```
# Gráfico Simples
```

```
boxplot(Banco$cartao_credito)
```

```
# Gráfico colorido e separado por categoria de emprego
```

```
boxplot(Banco$cartao_credito~Banco$catemp,  
        main="Cartao Credito por categoria emprego",  
        xlab="Catemp", ylab="Cartao Credito",  
        col=c("blue","red","yellow"))
```

```
# Gráfico colorido e separado por categoria de emprego sem outliers.
```

```
boxplot(Banco$cartao_credito~Banco$catemp,  
        main="Cartao Credito por categoria emprego",  
        xlab="Catemp", ylab="Cartao Credito",  
        col=c("blue","red","yellow"), outline=FALSE, horizontal = FALSE)
```

```
# Gráfico ggplot colorido por categoria de emprego
```

```
boxplot<-ggplot(Banco, aes(x=cartao_credito , y=salário, fill=catemp))+  
  geom_boxplot()+  
  xlab("cartao de credito")+  
  ylab("Cartao Credito")+  
  ggtitle("Cartao Credito e salario por categoria emprego")
```

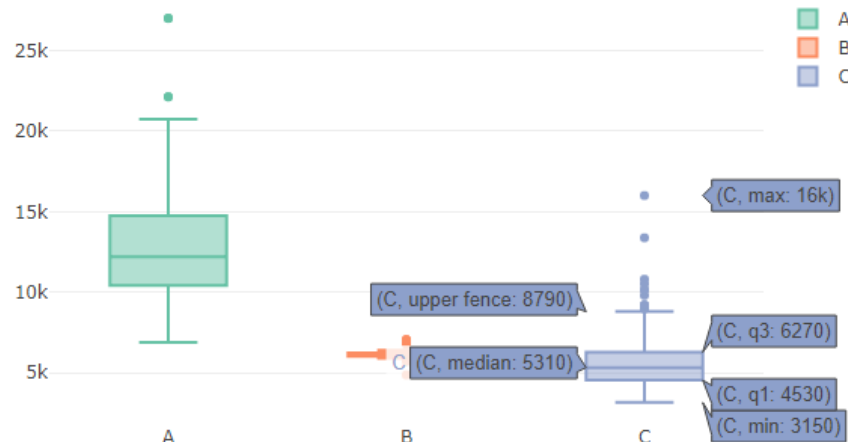
```
boxplot
```

```
# Gráfico BoxPlot interativo
```

```
plot_ly(x=Banco$catemp, y=Banco$cartao_credito,  
        main="Gráfico interativo",  
        xlab="Catemp", ylab="Cartao Credito",  
        color=(catemp), outline = TRUE, type='box')
```

```
# Gráfico interativo simples.
```

```
plot_ly (Banco, x=cartao_credito, type='box')
```



# Package Ggplot

```
# Gráfico Simples
plot(cartao_credito, type='line')

# Gráfico colorido
plot(cartao_credito, type='line')

# Gráfico colorido interativo
grafico_linha <- plot_ly(Banco, y= ~cartao_credito, type='scatter', mode='lines')
grafico_linha

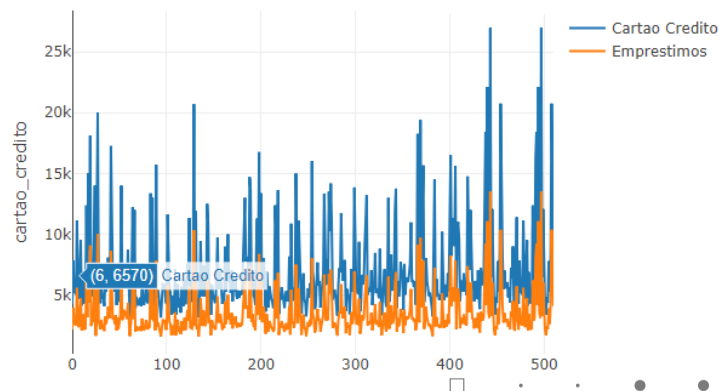
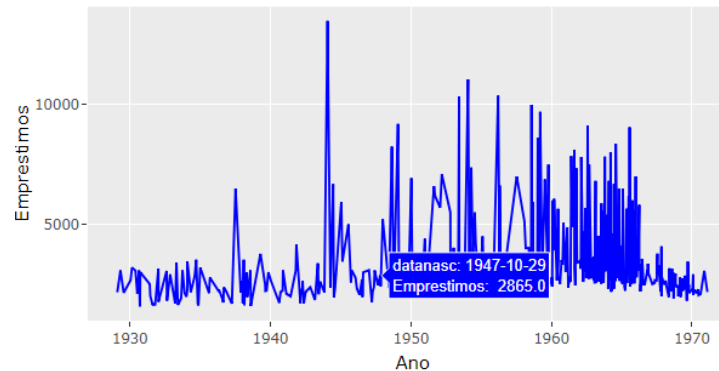
#Gráfico duplo interativo
duplo <-plot_ly(Banco, y= ~cartao_credito)%>%
  add_trace(y=~cartao_credito, type='scatter', name='Cartao Credito', mode='lines')%>%
  add_trace(y=~Emprestimos, type='scatter', name='Emprestimos', mode='lines')

duplo

#Gráfico Ggplot
linha <-ggplot(Banco, aes(x=datanasc, y=Emprestimos))+
  geom_line(col="blue")+
  xlab("Ano")+
  ylab("Emprestimos")+
  ggtitle("Emprestimos por ano")

linha
ggplotly(linha)
```

Emprestimos por ano



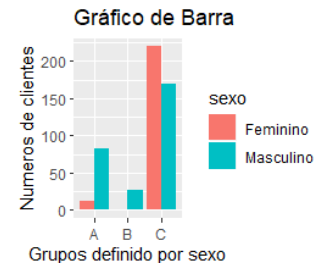
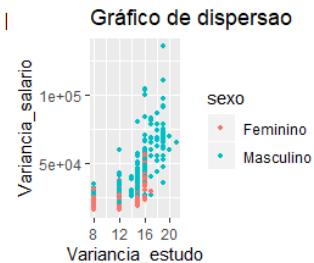
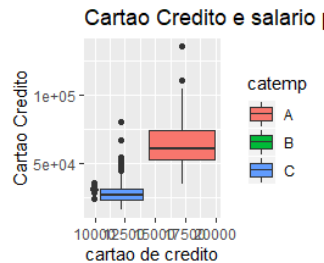
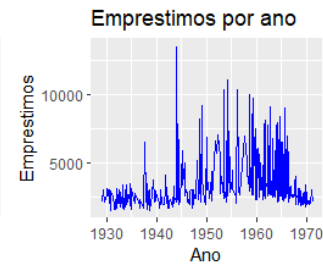
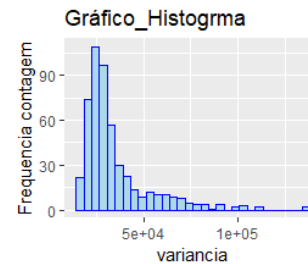
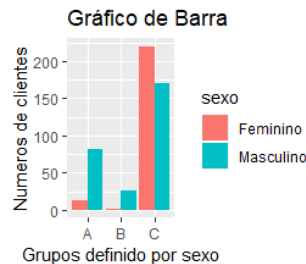


# Package GridExtra

```
#####
# Gráfico na mesma janela
#####

install.packages("GGally")
install.packages("gridExtra")
library(GGally)
library(gridExtra)

#Gráfico na mesma janela
grid.arrange(
  grafico_barras,
  histograma,
  linha,
  boxplot,
  dispersao2,
  grafico_barras,
  ncol=3, nrow=2)
```



# Package Esquisse

```
#####
```

```
# Pacote esquisse
```

```
#####
```

```
install.packages("esquisse")
```

```
library(esquisse)
```

```
install.packages("breakDown")
```

```
library(breakDown)
```

```
janela <- esquisser(Banco)
```

```
Banco <- Banco %>%
```

```
  filter(datanasc >= "1937-08-11 04:48:00" & datanasc <= "1971-02-10 00:00:00")
```

```
ggplot(Banco) +
```

```
  aes(x = estudo, y = salário, fill = sexo, colour = catemp, size = Ano) +
```

```
  geom_point() +
```

```
  scale_fill_viridis_d(option = "plasma") +
```

```
  scale_color_viridis_d(option = "plasma") +
```

```
  labs(x = "Estudo", y = "Salario", title = "Grafico Esquisse", fill = "Sexo", color = "Catemp") +
```

```
  theme_minimal()
```

# Package Esquisse



# Package Forecasting

## AirPassengers

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1949	112	118	132	129	121	135	148	148	136	119	104	118
1950	115	126	141	135	125	149	170	170	158	133	114	140
1951	145	150	178	163	172	178	199	199	184	162	146	166
1952	171	180	193	181	183	218	230	242	209	191	172	194
1953	196	196	236	235	229	243	264	272	237	211	180	201
1954	204	188	235	227	234	264	302	293	259	229	203	229
1955	242	233	267	269	270	315	364	347	312	274	237	278
1956	284	277	317	313	318	374	413	405	355	306	271	306
1957	315	301	356	348	355	422	465	467	404	347	305	336
1958	340	318	362	348	363	435	491	505	404	359	310	337
1959	360	342	406	396	420	472	548	559	463	407	362	405
1960	417	391	419	461	472	535	622	606	508	461	390	432

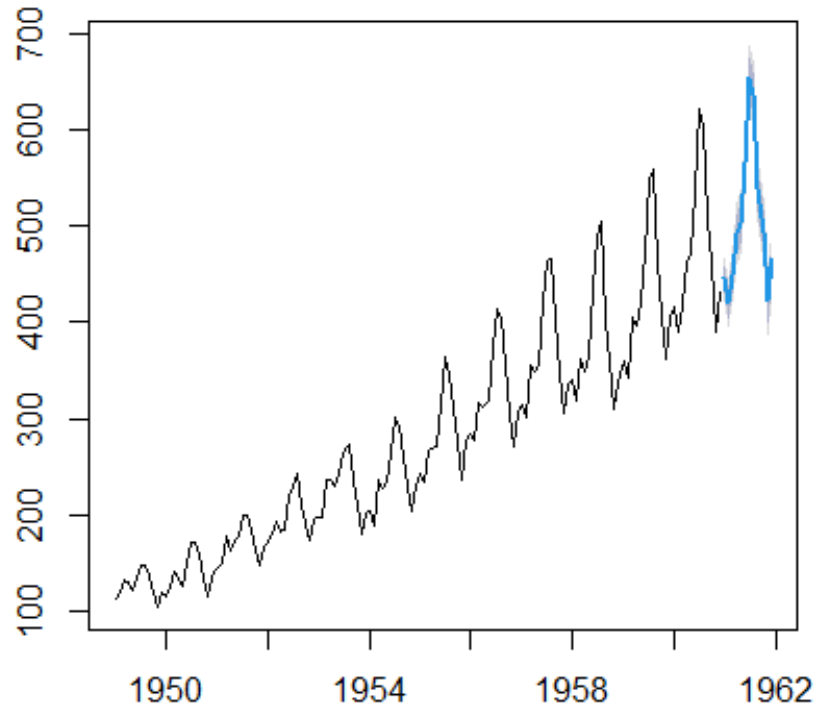
# Package Forecasting

```
library(forecast)
arima <- auto.arima(AirPassengers)
previsao <- forecast(arima, h=12)
previsao
```

	Point	Forecast	Lo 80	Hi 80	Lo 95	Hi 95
Jan 1961	446	431	460	423	468	
Feb 1961	420	403	438	394	447	
Mar 1961	449	430	469	419	479	
Apr 1961	492	471	513	460	524	
May 1961	503	482	525	470	537	
Jun 1961	567	544	589	532	601	
Jul 1961	654	631	677	619	690	
Aug 1961	639	615	662	602	675	
Sep 1961	541	517	565	504	578	
Oct 1961	494	470	518	457	531	
Nov 1961	423	399	448	386	461	
Dec 1961	466	441	490	428	503	

# Package Forecasting

Forecasts from  $\text{ARIMA}(2,1,1)(0,1,0)[12]$



# Package Caret

O caret é um pacote da linguagem R desenvolvido para **treinamento de modelos de classificação e regressão**. Dificilmente você irá trabalhar com machine learning no R sem utilizar este pacote em algum momento. Ele não possui os algoritmos de machine learning em suas funções, mas faz uso de outros pacotes para essa finalidade, permitindo assim que sejam **utilizados muitos algoritmos a partir da mesma função**, apenas alterando seus parâmetros.

Função train(): criando modelo

Parâmetro method: definindo algoritmo

Validação cruzada: Kfold

Parâmetro trControl: utilizando validação cruzada

Parâmetro tuneGrid: Configurando Algoritmo

Entre outras.



LET'S BUILD A  
ML MODEL  
WITH CARET

# Package Caret

```
library(caret)
```

```
rnorm(20)
```

```
[1]  0.359 -1.729  0.846 -0.244 -0.246 -2.044  0.326 -1.220 -1.289  0.273 -0.225  1.032 -2.167 -0.403
[15] -0.901 -0.279  0.217 -0.815  0.156  0.165
```

```
rnorm(20)
```

```
[1] -2.6293  0.1926 -1.1032  0.4394 -1.7133  0.5693  0.0534 -1.0191  0.1507  0.3934  0.0761  1.1701
[13]  1.0895 -0.7272 -0.4658  0.5132  1.3565  0.2145  0.3409  0.5482
```

```
set.seed(123456); rnorm(20)
```

```
[1]  0.8337 -0.2760 -0.3550  0.0875  2.2523  0.8345  1.3124  2.5026  1.1682 -0.4262 -0.9961 -1.1139
[13] -0.0557  1.1744  1.0532  0.0576 -0.7350  0.9305  1.6682  0.5597
```

```
set.seed(123456); rnorm(20)
```

```
[1]  0.8337 -0.2760 -0.3550  0.0875  2.2523  0.8345  1.3124  2.5026  1.1682 -0.4262 -0.9961 -1.1139
[13] -0.0557  1.1744  1.0532  0.0576 -0.7350  0.9305  1.6682  0.5597
```



# Package Caret

```
treinamento <-createDataPartition(credito_scoring$ID, p=.7,list = FALSE, times = 1)
head(treinamento)
```

```
Resample1
[1,]      1
[2,]      3
[3,]      4
[4,]      5
[5,]      6
[6,]      8
```

```
amostra_treinamento <-credito_scoring[treinamento,]
head(amostra_treinamento)
```

```
# A tibble: 6 x 7
```

	ID	credito	Rank_credito	Cargo	Pagamento	Faixa_Etaria	Cartao_Credito	
	<dbl>	<dbl>	<chr>	<chr>	<chr>	<chr>	<chr>	
1	1	0	Bom	Gerente	Mensal	Adulto (25-35)	Sim	
2	3	1	Mal	Profissional	Semanal	Jovem (< 25)	Sim	
3	4	0	Bom	Gerente	Mensal	Adulto (25-35)	Nao	
4	5	0	Bom	Profissional	Qualificado	Mensal	Jovem (< 25)	Nao
5	6	0	Bom	Diretor	Mensal	Jovem (< 25)	Sim	
6	8	1	Mal	Gerente	Mensal	Jovem (< 25)	Nao	

# Package Caret

```

amostra_teste <- credito_scoring[-treinamento,]
head(amostra_teste)
# A tibble: 6 x 7
  ID credito Rank_credito Cargo      Pagamento Faixa_Etaria  Cartao_Credito
  <dbl>   <dbl> <chr>      <chr>      <chr>      <chr>      <chr>
1     2     1 Mal      Gerente     Semanal    Adulto (25-35) Nao
2     7     0 Bom      Gerente     Mensal     Experiente ( > 35) Nao
3    18     1 Mal      Gerente     Mensal     Jovem (< 25) Nao
4    19     1 Mal      Profissional Qualificado Semanal     Jovem (< 25) Sim
5    20     1 Mal      Autonomo     Semanal     Jovem (< 25) Nao
6    22     1 Mal      Profissional Qualificado Semanal     Jovem (< 25) Sim

dim(amostra_treinamento)
[1] 454  7
dim(amostra_teste)
[1] 192  7
dim(credito_scoring)
[1] 646  7

454+192
[1] 646

```

# Package Rpart

```
library(rpart)
library(rattle)
```

```
table(credito_scoring$Cargo)
```

	Autonomo	Diretor	Gerente
Profissional	76	78	316
Nao Qualificado	82	94	

```
mytree <- rpart(
+ Rank_credito ~ Pagamento + Faixa_Etaria + Cartao_Credito + Cargo,
+ data = credito_scoring,
+ method = "class",
+ parms = list(split = 'gini'),
+ minsplit = 2,
+ minbucket = 1 # numero minimo em cada nó.
+ )
```

```
mytree
```

## Package Rpart

mytree

n= 646

node), split, n, loss, yval, (yprob)

\* denotes terminal node

1) root 646 310 Mal (0.47988 0.52012)

2) Pagamento=Mensal 316 50 Bom (0.84177 0.15823)

4) Faixa\_Etaria=Adulto (25-35),Experiente ( > 35) 218 2 Bom (0.99083 0.00917) \*

5) Faixa\_Etaria=Jovem (< 25) 98 48 Bom (0.51020 0.48980)

10) Cargo=Diretor,Profissional Qualificado 16 0 Bom (1.00000 0.00000) \*

11) Cargo=Gerente 82 34 Mal (0.41463 0.58537) \*

3) Pagamento=Semanal 330 44 Mal (0.13333 0.86667)

6) Faixa\_Etaria=Experiente ( > 35) 14 0 Bom (1.00000 0.00000) \*

7) Faixa\_Etaria=Adulto (25-35),Jovem (< 25) 316 30 Mal (0.09494 0.90506) \*

## Package Rpart

```
mytree$variable.importance
```

Pagamento	Faixa_Etaria	Cargo	Cartao_Credito
162.0	137.3	122.7	10.3

```
credito_scoring$prob <- predict(mytree, newdata = credito_scoring, type = "prob")
credito_scoring$class <- predict(mytree, newdata = credito_scoring, type = "class")
```

```
acerto <- table(credito_scoring$Rank_credito, credito_scoring$class)
acerto
```

	Bom	Mal
Bom	246	64
Mal	2	334

```
table(credito_scoring$Rank_credito)
```

	Bom	Mal
Bom	310	336

```
acerto_geral <- (acerto[1]+acerto[4])/sum(acerto)
acerto_geral
```

```
[1] 0.898
```

## Package Rpart

```
confusionMatrix(factor(credito_scoring$Rank_credito),factor(credito_scoring$class))
```

Confusion Matrix and Statistics

```

              Reference
Prediction Bom Mal
Bom      246    64
Mal       2   334

```

```

Accuracy : 0.898
 95% CI : (0.872, 0.92)
No Information Rate : 0.616
P-Value [Acc > NIR] : < 0.00000000000000002

```

```
Kappa : 0.794
```

```
McNemar's Test P-Value : 0.00000000000000598
```

```

Sensitivity : 0.992
Specificity : 0.839
Pos Pred Value : 0.794
Neg Pred Value : 0.994
Prevalence : 0.384
Detection Rate : 0.381
Detection Prevalence : 0.480
Balanced Accuracy : 0.916

```

```
'Positive' Class : Bom
```

# Package Rpart

```
path.rpart(mytree,node=4)
```

```
node number: 4
```

```
root
```

```
Pagamento=Mensal
```

```
Faixa_Etaria=Adulto (25-35),Experiente ( > 35)
```

```
path.rpart(mytree,node=5)
```

```
node number: 5
```

```
root
```

```
Pagamento=Mensal
```

```
Faixa_Etaria=Jovem (< 25)
```

```
path.rpart(mytree,node=6)
```

```
node number: 6
```

```
root
```

```
Pagamento=Semanal
```

```
Faixa_Etaria=Experiente ( > 35)
```

```
path.rpart(mytree,node=7)
```

```
node number: 7
```

```
root
```

```
Pagamento=Semanal
```

```
Faixa_Etaria=Adulto (25-35),Jovem (< 25)
```

```
path.rpart(mytree,node=10)
```

```
node number: 10
```

```
root
```

```
Pagamento=Mensal
```

```
Faixa_Etaria=Jovem (< 25)
```

```
Cargo=Diretor,Profissional Qualificado
```

```
path.rpart(mytree,node=11)
```

```
node number: 11
```

```
root
```

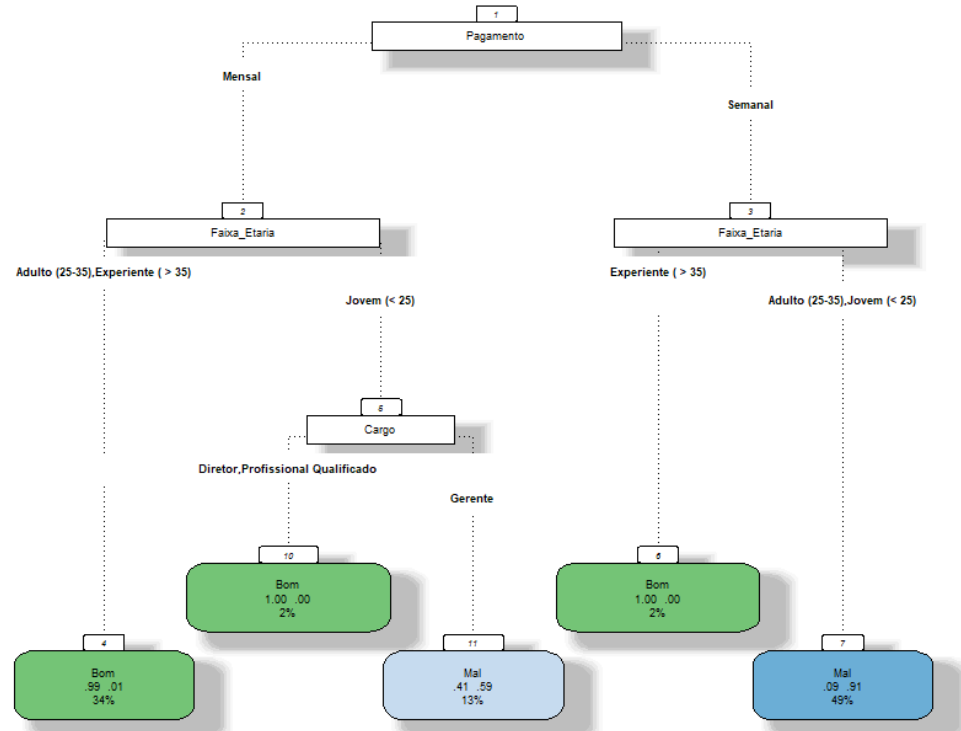
```
Pagamento=Mensal
```

```
Faixa_Etaria=Jovem (< 25)
```

```
Cargo=Gerente
```

# Package Rpart

`fancyRpartPlot(mytree, type=5)`





## Referências Bibliográficas

*Teetor, Paul. 2011. 25 Recipes for Getting Started with R. O'Reilly Media.*

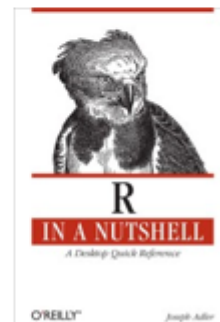
*Teetor, Paul. 2011. R Cookbook. O'Reilly Media.*

*Adler, Joseph. 2012. R in a Nutshell. O'Reilly Media.*

Tutorial sobre o R: <http://tryr.codeschool.com/>

Rmarkdown em <http://rmarkdown.rstudio.com/>

Lista de discussão [r-br-request@listas.c3sl.ufpr.br](mailto:r-br-request@listas.c3sl.ufpr.br)



# OBRIGADO



Copyright © 2022 | Professor (a) Edmar Caldas  
Todos os direitos reservados. Reprodução ou divulgação total ou parcial deste documento, é expressamente proibido sem consentimento formal, por escrito, do professor/autor.

FIAP