Pré-Processamento de Dados

ESTAT0109 – Mineração de Dados em Estatística

Prof. Dr. Sadraque E. F. Lucena

sadraquelucena@academico.ufs.br

http://sadraquelucena.github.io/mineracao

Objetivo da Aula

- Dominar o pipeline de pré-processamento de dados

Por que Pré-Processar os Dados

- A qualidade de um modelo depende completamente da qualidade dos dados que utilizamos para construí-lo.
- Facilmente encontramos em bases de dados:
  - **Ruído (*noisy*):** Valores errados ou impossíveis (ex: um paciente com `IDADE` = 200 anos).
  - **Inconsistência (*inconsistency*):** A mesma informação registrada de formas diferentes (ex: `MUNICIPIO` = “Aracaju”, “Aracajú”).
  - **Heterogeneidade (*heterogeneity*):** Dados de múltiplas fontes, com formatos e chaves diferentes (ex: o código o município do Censo Escolar e do SIM usando padrões diferentes – 6 ou 7 dígitos).
  - **Dandos faltantes (*Missing*):** O famoso `NA`.

Por que Pré-Processar os Dados

- O objetivo do pré-processamento de dados é **transformar dados brutos e "sujos" em um conjunto de dados limpo, coeso e de alta qualidade** que seja apropriado para a mineração.

- Antes de aplicar qualquer técnica de limpeza, transformação ou integração, precisamos fazer um diagnóstico. A primeira e mais fundamental etapa desse diagnóstico é entender a natureza dos nossos dados.
- Precisamos identificar os tipos de atributos (ou variáveis) que temos.

Tipos de Dados

Tipos de Dados

Todo conjunto de dados é composto por duas partes fundamentais, assim como uma planilha:

- **Objetos de Dados (as Linhas):** Representam a entidade que estamos observando.
  - Também chamados de: *amostras*, *instâncias*, *casos* ou *tuplas* (em bancos de dados).
  - Exemplos: um *paciente*, um *cliente*, um *município*, um *domicílio*.
- **Atributos (as Colunas):** Representam a característica ou propriedade que descreve o objeto.
  - Também chamados de: *dimensões*, *features* (em *Machine Learning*) ou *variáveis* (nosso termo preferido em Estatística).
  - Exemplos: `IDADE`, `SEXO`, `POPULACAO_2022`, `IDH_M`.
- Classificar os **tipos de atributos**, define quais técnicas estatísticas e de mineração podemos usar. Vejamos as classificações.

Tipos de Atributos

Nominal

- A ordem não importa.
- Exemplos: `ocupacao`(estatístico, médico, professor…), `COD_MUNICIPIO_IBGE` (ex: “2800308” = Aracaju).
- Um atributo nominal pode usar números (como `COD_MUN_IBGE` ou `ID_cliente`), mas operações matemáticas são sem sentido.
  - Não podemos calcular a “média” dos códigos de município.
  - A única medida de tendência central válida é a Moda (o valor mais frequente).

Tipos de Atributos

Binário (booleano)

- É um atributo nominal com apenas dois estados).
- Exemplos: `OBITO` (1=Sim, 0=Não), `FUMANTE` (1=Sim, 0=Não).
- **Subtipos Importantes:**
  - **Simétrico:** Ambos os estados têm o mesmo “peso”. Ex: `SEXO` (M/F).
  - **Assimétrico:** Um estado é mais “raro” ou “importante”. Ex: `TESTE_COVID` (Positivo=1, Negativo=0). Por convenção, o “1” é o resultado de maior interesse.

Tipos de Atributos

Ordinal

- Os valores têm uma **ordem ou ranking significativo**, mas a **magnitude (distância) entre eles é desconhecida**.
- Sabemos que Grande > Médio > Pequeno, mas não o quanto “Grande” é maior que “Médio”.
- Exemplos: `ESCOLARIDADE` (Analfabeto < Fundamental < Médio < Superior), `SATISFACAO_ATENDIMENTO`: (Muito Ruim, Ruim, Neutro, Bom, Muito Bom).
- Estatísticas Válidas: **Moda** e **Mediana**. A Média ainda não faz sentido.

Tipos de Atributos

Quantitativo (Numérico)

São quantidades mensuráveis (números inteiros ou reais) onde as operações matemáticas (média, desvio padrão) fazem sentido. Há dois tipos:

1. **Intervalar (Interval-Scaled):** Tem ordem e as “distâncias” (intervalos) são iguais. O zero é apenas um ponto na escala, não significa ausência. Exemplos:
   - **Temperatura (°C / °F):** 0°C não é “ausência de temperatura”. Não podemos dizer que 20°C é “o dobro do calor” de 10°C.
   - **Datas (Ano):** O “Ano 0” não foi o começo do tempo.

- **Operações:** Média, Mediana, Moda, Diferenças (20°C – 10°C = 10°C).

Tipos de Atributos

Quantitativo (Numérico)

2. **Racional (Ratio-Scaled):** Tem ordem, distâncias iguais **E POSSUI UM “ZERO VERDADEIRO”**. O Zero (0) significa a ausência total da medida.
   - `VL_TOTAL_INTERNACAO` (R$ 0,00 é ausência de custo).
   - `IDADE`, `PESO`, `ALTURA`.
   - `RENDA_MENSAL`, `N_DE_FILHOS`.

- **Operações:** Todas! Média, Mediana, Moda, Diferenças E Rácios (R$ 100 é exatamente o dobro de R$ 50).

A Visão do *Machine Learning*

Muitas vezes, os algoritmos e softwares (como R e Python) simplificam a classificação em dois grandes grupos, que não são exatamente iguais aos anteriores, mas os englobam.

1. **Atributo DISCRETO:** Possui um conjunto de valores finito ou infinito contável (geralmente inteiros). Engloba:
   - Nominal (ex: `RACA_COR`)
   - Binário (ex: `OBITO`)
   - Ordinal (ex: `ESCOLARIDADE`)
   - Numéricos Inteiros (ex: `N_DE_FILHOS`, `DIAS_DE_INTERNACAO`).

A Visão do *Machine Learning*

2. **Atributo CONTÍNUO:** Possui um número infinito de valores “não contáveis” dentro de um intervalo (números reais, *floating-point*). Engloba atributos numéricos (Intervalares ou Racionais) que são medidos com casas decimais:
   - `PESO` (ex: 75,32 kg)
   - `TAXA_MORTALIDADE_INFANTIL` (ex: 12.4 por 1000)
   - `VL_MEDICAMENTO` (ex: R$ 150,75)

Pré-Processamento de Dados

Pré-Processamento de Dados

- Dados de entrada incorretos ou de baixa qualidade resultam inevitavelmente em saídas incorretas ou de baixa qualidade.
- Se não for tomado o devido cuidado em lidar adequadamente com questões de qualidade de dados antes de treinar um modelo, a saída do modelo será não confiável, enganadora ou simplesmente incorreta.

Objetivo: Arrumar os dados para iniciar uma análise de boa qualidade. Questões que temos que lidar:

1. Tratamento de Inconsistências
2. Valores Ausentes (*Missing Values*)
3. Ruído (*Noise*)
4. Integração de Dados (*Data Integration*)
5. Transformação de Dados (*Data Transformation*)
6. Redução de Dados (*Data Reduction*)

Vejamos detalhes de cada fase.

1. Tratamento de Inconsistências

1. Tratamento de Inconsistências

- Ocorre quando o mesmo dado é registrado de formas diferentes (sintaxe) ou quando um valor viola uma regra lógica (semântica).
- **Com identificar?**
  - **Para categorias:** Fazer uma tabela de frequência para avaliar há registros com sintaxe diferente (ex: `municipio` com respostas “Aracaju” e “Aracajú”).
  - **Para numéricos:** Obter mínimo, máximo e Boxplots par aver se há valores fora so possível (ex: `idade_da_mae` = 5 anos).
- **O que fazer:**
  - **Padronização de categorias:** Agrupar sinônimos ou erros de digitação em um único rótulo padrão (ex: “Aracajú”, “AJU”, “Aracaju” → “Aracaju”).
  - **Validação de regras:** Transformar valores inválidos em `NA`s.

2. Valores Ausentes (*Missing Values*, **NA**s)

2. Valores Ausentes (*Missing Values*, NAs)

Um `NA` pode ocorrer por 2 fatores principais:

a. **Erro Aleatório:**

- O digitador esqueceu; o paciente/cliente não quis informar.
- *Ação:* **Imputação**.

b. **Erro Estrutural / "Não Aplicável":**

- *Ex (DATASUS):* `DATA_OBITO` está `NA` (porque o paciente está vivo).
- *Ex (CadÚnico):* `NOME_ESCOLA_FILHO` está `NA` (porque a família não tem filhos).
- *Ação:* **Não imputar!** O `NA` aqui é informação. Talvez criar uma categoria "Não Aplicável".

Moral: Sempre leia o **Dicionário de Dados**!

2. Valores Ausentes (*Missing Values*)

Ao lidarmos com NAs, algumas estratégias costumam ser utilizadas:

a. Remover a linha inteira se ela tiver algum NA;

b. Substituir por uma constante;

c. Substituir por uma medida de tendência central;

d. Substituir pelo valor mais provável.

2. Valores Ausentes (*Missing Values*)

a. Remover a linha inteira se ela tiver algum `NA`

- Remover a linha inteira apenas se o `NA` ocorreu por acaso e se a perda for menor que 5% dos dados.
- **Problema:**
  - **Perda de Informação:** Se o `NA` estava só em `RACA_COR`, jogamos fora `IDADE`, `SEXO`, `MUNICIPIO` e o desfecho (a *label*).
  - **Viés de Seleção (PERIGO!):** E se os dados *não* estiverem faltando ao acaso?
    - *Exemplo:* Se só os mais ricos não responderam a renda e você **remove os mais ricos da sua análise**, o seu modelo se torna enviesado.

2. Valores Ausentes (*Missing Values*)

b. Substituir por uma constante

- `NA` em `RENDA` → `0`
- `NA` em `RACA_COR` → `"Desconhecido"` ou `"99"`
- **Problema:**
  - O algoritmo pode erroneamente achar que “os `NAs` formam um conceito interessante”.
  - O modelo aprende que “Renda = 0” é um forte preditor, quando na verdade ele só significa “dado faltante”.
  - Isso distorce a distribuição dos dados (ex: cria um pico falso no “0”).

2. Valores Ausentes (*Missing Values*)

c. Substituir por uma medida de tendência central

Substitui o `NA` pela medida “do meio” da distribuição daquele atributo. * **A Regra de Ouro:** * **Média:** Usar se a distribuição for **simétrica** (ex: `IDADE`, se for normal). * **Mediana:** Usar se a distribuição for **assimétrica** (ex: `RENDA`). * **Vantagem:** É rápido e preserva a média/mediana geral. * **Desvantagem:** Ignora as relações entre variáveis e **“achata” a variância** (subestima a variabilidade real).

2. Valores Ausentes (*Missing Values*)

d. Substituir pelo valor mais provável

Esta é a abordagem moderna e preferida na maioria dos casos. Trata o valor ausente como um problema de predição.

- **Conceito:** Usamos os outros atributos (`X1`, `X2`, `X3`) para *prever* o valor faltante (`Y_na`).
- **Como?**
  - **Regressão:** Para prever `RENDA` (numérico) usando `IDADE` e `ESCOLARIDADE`.
  - **Árvore de Decisão / k-NN:** Para prever `RACA_COR` (categórico) usando `MUNICIPIO` e `RENDA`.
  - *No R:* Pacotes como `recipes` (Tidymodels) ou `mice` fazem isso.
- **Vantagem:** Usa a maior parte da informação dos dados presentes e preserva as relações entre os atributos.

3. Ruído (*Noise*)

3. Ruído (*Noise*)

Ruído é um erro aleatório ou variância em uma variável medida.

- Não é um `NA`, mas um valor que parece “deslocado”.
- **Exemplo (SIH/DATASUS):**
  - Uma internação por apendicite (`VL_TOTAL`) com custo de R$ 1,50.
  - Uma internação com custo de R$ 5.000.000,00 (enquanto a média é R$ 2.000,00).
- **Objetivo:** “Suavizar” (*smooth*) esses dados para remover a variação aleatória sem perder o sinal verdadeiro.

3. Ruído (*Noise*)

Técnica 1: *Binning* (Agrupamento ou Discretização)

Binning é uma técnica de *suavização local* (olha a "vizinhança").

O Processo (Ex: `VALOR_INTERNACAO`):

1. **Ordenar os dados:** `[4, 8, 15, 21, 21, 24, 25, 28, 34]`
2. **Particionar em "Bins" (Baldes):**
   - *Ex: Bins de frequência igual (tamanho 3)*
   - **Bin 1:** `[4, 8, 15]`
   - **Bin 2:** `[21, 21, 24]`
   - **Bin 3:** `[25, 28, 34]`
3. **Substituir (Suavizar):** Aplicar uma regra ao bin.

3. Ruído (*Noise*)

Tipos de Suavização por *Binning*

Usando o exemplo (Bin 1: `[4, 8, 15]`):

1. Suavização pela MÉDIA: * **O que faz:** Substitui todos os valores pela média do bin. * *Ex:* Média(4, 8, 15) = 9 * *Resultado:* `[9, 9, 9]`

2. Suavização pela MEDIANA: (Muito recomendado!) * O que faz:** Substitui todos pela mediana do bin (robusto a *outliers*!). * *Ex:* Mediana(4, 8, 15) = 8 * *Resultado:* `[8, 8, 8]`

3. Suavização pelos LIMITES: * **O que faz:** Substitui cada valor pelo limite (min/max) mais próximo. * *Ex:* `[4, 8, 15]` -> `[4, 4, 15]` (8 está mais perto de 4 do que de 15)

3. Ruído (*Noise*)

Técnica 2 e 3: Regressão e Análise de *Outliers*

O *Binning* não é a única forma de suavizar dados.

Regressão: Ajusta os dados a uma função (ex: uma linha de regressão linear). * **Como suaviza?** O "ruído" é a variação aleatória (o erro, ϵ) ao redor da linha. O valor "suavizado" é o valor *predito* pela linha.

Análise de *Outliers* (via Clustering): Agrupa dados similares (clusters). * **Como suaviza?** Valores que caem fora dos clusters podem ser considerados *outliers* (ruído).

4. Integração de Dados (*Data Integration*)

4. Integração de Dados (*Data Integration*)

É o processo de combinar dados de múltiplas fontes.

O Desafio: Os dados *nunca* vêm de uma única fonte limpa. * Queremos cruzar **Taxas de Mortalidade (SIM/DATASUS)**… * … com **Indicadores Socioeconômicos (Censo/IBGE)**… * … com **Dados de Escolaridade (CadÚnico)**… * … para **cada Município de Sergipe**.

Temos quatro grandes desafios ao fazer isso. Vejamos.

4. Integração de Dados (*Data Integration*)

Desafio 1: O Problema da Identificação da Entidade

Como o computador sabe que ‘Aracaju’ é ‘Aracaju’?

- **Definição:** Como parear entidades do mundo real (pacientes, municípios, clientes) que estão em bases diferentes?
- **Ex:** `id_cliente` (Base A) vs. `numero_client` (Base B).

Exemplo: * **Base IBGE (Censo):** O código de Aracaju é `CD_MUN_IBGE` = **“2800308”** (7 dígitos). * **Base DATASUS (SIH):** O código de Aracaju é `CD_MUN_DATASUS` = **“280030”** (6 dígitos). * **Solução:** Não dá para juntar direto! Precisamos transformar `CD_MUN_IBGE` para criar uma chave compatível com `CD_MUN_DATASUS`.

4. Integração de Dados (*Data Integration*)

Desafio 2: Conflito de Valores

Ok, conseguimos fazer o `join()` pelo código do município. Agora o problema é outro: **os valores não “falam” a mesma língua**.

Causas (do Texto):

1. **Diferença de Escala/Unidade:** (O mais comum!)
    - *Base A (IBGE):* `RENDA` em “Salários Mínimos”.
    - *Base B (CadÚnico):* `RENDA_PER_CAPITA` em “Reais (R$)”.
    - *Base C (World Bank):* `GDP` em “Dólares (USD)”.
2. **Diferença de Abstração:**
    - *Base A (SIH):* `VL_TOTAL_INTERNACAO` (nível do paciente).
    - *Base B (CNES):* `ORCAMENTO_ANUAL_HOSPITAL` (nível da unidade).
3. **Diferença de Semântica:**
    - *Base A:* `INDICE_ESCOLARIDADE` (População > 18 anos).
    - *Base B:* `INDICE_ESCOLARIDADE` (População > 25 anos).

4. Integração de Dados (*Data Integration*)

Desafio 3: Redundância (“Informação Repetida”)

Um atributo que pode ser “derivado” de outros.

- **Exemplo:**
  - Você baixa uma tabela que tem as colunas:
  - `POP_TOTAL`
  - `POP_URBANA`
  - `POP_RURAL`
- **Problema:** `POP_TOTAL` é redundante (é `POP_URBANA + POP_RURAL`).
- **Por que é ruim?**
  - Aumenta a dimensionalidade (Maldição da Dimensionalidade).
  - Viola premissas de alguns modelos (ex: Multicolinearidade em Regressão).
  - Dá peso duplicado a uma mesma informação em algoritmos de distância (K-means).

4. Integração de Dados (*Data Integration*)

Como Detectar Redundância? Com Estatística!

Para Atributos NUMÉRICOS (ex: `POP_TOTAL` vs `POP_URBANA`): * **Coeficiente de Correlação (Pearson)** * `cor(dados$pop_total, dados$pop_urbana)` * Se r for muito alto (ex: > 0.9), há forte suspeita de redundância. * **Covariância**

Para Atributos NOMINAIS (Categóricos): * **Teste χ^2 (Qui-Quadrado)** * `chisq.test(table(dados$var1, dados$var2))` * Mede a independência. Se p-valor for baixo (ex: < 0.05), as variáveis são *dependentes*, o que pode indicar redundância (ex: `COD_MUNICIPIO` e `NOME_MUNICIPIO` são 100% dependentes).

4. Integração de Dados (*Data Integration*)

Desafio 4: Duplicação de linhas

A *mesma entidade* (linha) aparece mais de uma vez.

- **Exemplo:** O mesmo cliente (`"João da Silva"`) aparece duas vezes na tabela de compras, uma com endereço “Rua A” e outra com “Rua B” (pois ele se mudou e a base não foi atualizada corretamente).
- **Exemplo:**
  - **Record Linkage (Ligação de Registros)**
  - A paciente `MARIA JOSE DA SILVA` (do CadÚnico) é a *mesma* paciente `M J SILVA` (do SINAN/Dengue)?
- **Problema:** Gera inconsistências e superestima contagens.
- **Solução:** Requer técnicas avançadas (ex: *fuzzy matching*) para encontrar duplicatas “prováveis” e consolidá-las. (Isso dá um ótimo TCC com o Prof. Sadraque!)

5. Transformação de Dados (*Data Transformation*)

5. Transformação de Dados (*Data Transformation*)

- Frequentemente é preciso modificar a estrutura ou características dos dados para formas “apropriadas para a mineração”.
- Algumas técnicas usadas são:

  a. Normalização Z-score;

  b. Discretização Min-Max;

  c. Transformação logarítmica;

  d. Discretização;

  e. Codificação de variáveis *dummy*.

5. Transformação de Dados (*Data Transformation*)

a. Normalização Z-score

- Conhecida como *normalização z-score* ou *normalização de média zero*.
- Esta abordagem resulta em valores com média 0 e variância 1.
- A variável normalizada v' é dada por

$$v' = \frac{v - \overline{v}}{\sigma_v},$$

em que v é a variável original, $\overline{v}$ é a média da variável v e σ_v é o desvio padrão da variável v.

Quem Precisa Disso?

- **K-Means** (Clusterização), **K-Nearest Neighbors (K-NN)** (Classificação), **SVM** (Classificação), **Redes Neurais**, **PCA** (Redução de Dimensionalidade).

Basicamente, **qualquer algoritmo baseado em distância**!

5. Transformação de Dados (*Data Transformation*)

b. Normalização Min-Max

- Mapeia linearmente os valores para um novo intervalo, geralmente **[0, 1]**, usando:

$$v'_i = \frac{v_i - \min_A}{\max_A - \min_A} \times (\text{novo\_max} - \text{novo\_min}) + \text{novo\_min}$$

  (Para [0, 1], os dois últimos termos desaparecem).

Exemplo (RENDA):

- $v = 700, \min = 200, \max = 2000$
- $v' = (700 - 200)/(2000 - 200) = 500/1800 = 0.277$
- **Vantagem:** Preserva as relações lineares.
- **Desvantagem: Extremamente sensível a outliers!** Um único valor de Renda de R$ 50.000 (um erro) “espremeria” todos os outros dados perto de 0.

5. Transformação de Dados (*Data Transformation*)

Atenção!!!

NUNCA ajuste seus parâmetros de normalização (min/max ou z-score) usando os dados de **TREINO E TESTE** juntos!

Isto é um *vazamento de dados* (*data leakage*). Você estaria "contando" ao seu modelo de treino sobre a distribuição do futuro (teste).

O Processo Correto (Pipeline):

1. Divida os dados: **Treino** e **Teste**.
2. Calcule os parâmetros (ex: `mean()`, `sd()`) **APENAS** no conjunto de **Treino**.
3. Aplique esses mesmos parâmetros em **ambos** (Treino e Teste).
4. No R (Tidymodels), o `recipe()` + `step_normalize()` faz isso automaticamente!

5. Transformação de Dados (*Data Transformation*)

c. Transformação logarítmica

- As transformações anteriores são boas quando os dados são simétricos.
- A transformação logarítmica é mais adequada para distribuições assimétricas e dados com valores que variam amplamente em magnitude.
- A transformação é

$$v' = \log(v).$$

5. Transformação de Dados (*Data Transformation*)

d. Discretização

- A *discretização* consiste em transformar variáveis contínuas em categóricas.
- Alguns algoritmos exigem que a variável independente seja binária ou tenha um número limitado de valores distintos.
- Esse processo pode ser feito usando a suavização com médias de intervalo ou suavização com limites de intervalo.
- Outra forma comum é a *dicotomização*.
  - Exemplo: Os valores $\{4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34\}$ seriam dicotomizados usado 20 como valore de corte, ficando $\{0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1\}$.

5. Transformação de Dados (*Data Transformation*)

e. Codificação de variáveis *dummy*

- Uma variável *dummy* é uma dicotomização de uma variável contínua.
- Ela é muito usada em algoritmos que exigem que os atributos independentes sejam numericos (como regressão ou k-NN) e como uma forma de representar dados ausentes.
- Suponha que temos a variável abaixo:

Escolaridade	**Código**
Enino Fundamental	1
Ensino Médio	2
Ensino Superior	3

5. Transformação de Dados (*Data Transformation*)

e. Codificação de variáveis *dummy*

- Usando uma variável dummy completa, temos:

Escolaridade	Ensino Médio	Ensino Superior
Enino Fundamental	0	0
Ensino Médio	1	0
Ensino Superior	0	1

- Agora, ao invés de uma variável, temos 2 variáveis *dummies*.
- Em geral, o número de variáveis *dummies* criadas é $n - 1$, em que n é o número de categorias da variável original.
- Em geral, a categoria que não virou *dummy* é porque ela é, de alguma forma, menos importante para o estudo.

6. Redução de Dados (*Data Reduction*)

6. Redução de Dados (*Data Reduction*)

O Objetivo: Reduzir o número de colunas (k) de d para k (onde $k < d$), preservando o máximo de "sinal" e removendo "ruído".

Três Estratégias Principais:

1. **Projeção Linear (PCA - Principal Components Analysis):**
   - **O que faz: Combina** atributos correlacionados para criar *novos* eixos (Componentes Principais) que capturam o máximo de variância dos dados.
   - **Resultado:** Um *novo* dataset (ex: $k = 5$) onde cada coluna é uma combinação linear das originais (ex: $d = 50$).

6. Redução de Dados (*Data Reduction*)

2. **Seleção de Atributos (Attribute Subset Selection):**
   - **O que faz: Remove** atributos irrelevantes ou redundantes. *Não cria* novas colunas.
   - **Como:** Métodos *greedy* (Heurísticos) como *Forward Selection* (adiciona o melhor) ou *Backward Elimination* (remove o pior).
   - **Resultado:** Um *subconjunto* do dataset original (ex: $k = 10$ das $d = 50$ colunas originais).
3. **Mapeamento Não Linear (ex: t-SNE, Kernel PCA):**
   - **O que faz:** (Para quando o PCA falha). Mapeia os dados de alta dimensão para baixa dimensão (ex: $k = 2$ ou $k = 3$) **preservando a estrutura de vizinhança** (proximidade).
   - **Resultado:** Essencial para *visualizar* clusters complexos que são “emaranhados” nos dados originais.

Agora vamos fazer no R...

Fim