KNN, Credit Scoring and Database

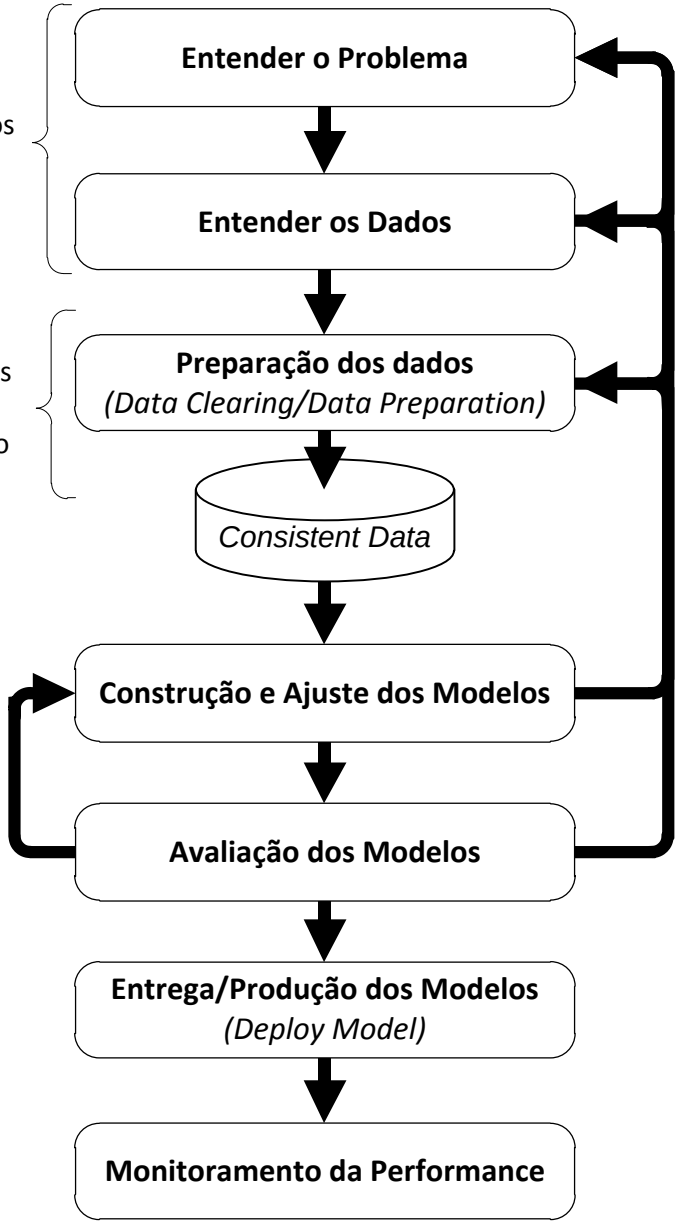
Objetivos:

- Apresentar o a interfacer de comunicação com banco de dados DBI
- Apresentar o banco de dados SQLite
- Utilizar o algoritmo: K-Nearest Neighbour (KNN) para classificar o crédito de clientes com de um banco nas categorias:
  - "Bom": Baixa probabilidade de default
  - "Ruim": Baixa probabilidade de default
- Avaliar a performance do modelo utilizando *confusion matrix*

Packages: `DBI,SQLite,class`

Base de dados SQL: `database/GermanCredit.sqlite.db`

Decision Tree and Credit Scoring



- Feito uma única vez
- Utilizar aplicativos interativos
- Normalmente não há necessidade de automatizar

Automatizar a execução no caso de Projetos ou Modelos em que este procedimento será recorrente ou o modelo esteja sujeito a alteração

- Brain Storn, experiência própria e conversar com as partes interessadas
- Definir objetivo (OUTPUT):

Classificação do crédito como em duas categorias: Bom ou Ruim

- Análise Exploratória dos dados
- Regressões, tabelas de contingência, histogramas
- Medidas de resumos das variáveis
- Classificação das variáveis

- Padronização dos dados
- Categorização dos dados
- Operações aritméticas
- Inclusão de variáveis/exclusão

Modelos de Classificação:
- **K-Nearest Neighbour (KNN)**
- Redes Bayesianas
- Regressões Logit

- Confusion Matrix
- ROC *Receiver Operating Characteristic Curve*
- *Cross-Folder validadtion*
- Erro Quadrado Médio
- Etc..

KNN, Credit Scoring and Database

Base de Dados: *German Credit Database*

Referência:

- Asuncion, A. & Newman, D.J. (2007). UCI Machine Learning Repository http://www.ics.uci.edu/~mlearn/MLRepository.html . Irvine, CA: University of California, School of Information and Computer Science. Source of German Credit Data.
- Gayler, R (2008) Credit Risks Analystics Occasional newsletter. Retrieved from http://r.gayler.googlepages.com/CRAON01.pdf
- Guide to Credit Scoring in R – CRAN https://www.google.com.br/search?q=Guide+to+Credit+Scoring+in+R&ie=utf-8&oe=utf-8&client=firefox-b-ab&gfe_rd=cr&ei=jVFYV8z6J--p8weckLsQ
- Analysis of German Credit Data - The Pennsylvania State University - STAT 897D - Applied Data Mining and Statistical Learning

KNN, Credit Scoring and Database

Base de dados:

Informações pessoais e financeiras, como por exemplo

- Informações Pessoais
  - Nível de Escolaridade
  - Idade
  - Estado Civil, etc...
- Informações do Empréstimo
  - Duração
  - Montante, etc...
- Informações Financeiras
  - Montante de poupança
  - Patrimônio, etc...

KNN, Credit Scoring and Database

Base de dados: Tabela não normalizada organizada em colunas

Tabela: *DataDictionary*

Id	ColName	ColType
1	AccountBalance	discretized
2	DurationOfCreditMonth	numerical
3	PaymentStatusOfPreviousCredit	qualitative
4	Purpose	qualitative
5	CreditAmount	numerical
6	ValueSavingsOrStocks	discretized
7	LengthOfCurrentEmployment	discretized
8	InstalmentPerCent	numerical
9	SexMaritalStatus	qualitative
10	Guarantors	qualitative
11	DurationInCurrentAddress	numerical
12	MostValuableAvailableAsset	qualitativa
13	Age	numerical
14	ConcurrentCredits	qualitative
15	TypeOfApartment	qualitative
16	NoOfCreditsAtThisBank	numerical
17	Occupation	qualitative
18	NoOfDependents	numerical
19	Telephone	qualitative
20	ForeignWorker	qualitative
21	Creditability	qualitative

Tabela: *CreditData*

Id	Creditability	AccountBalance	DurationOfCreditMonth	PaymentStatusOfPrev
1	1	1	18	4
2	1	1	9	4
3	1	2	12	2
4	1	1	12	4
5	1	1	12	4
6	1	1	10	4
7	1	1	8	4
8	1	1	6	4
9	1	4	18	4
10	1	2	24	2
11	1	1	11	4
12	1	1	30	4
13	1	1	6	4
14	1	2	48	3
15	1	1	18	2
16	1	1	6	2
17	1	1	11	4
18	1	2	18	2
19	1	2	36	4
20	1	4	11	4
21	1	1	6	4

KNN, Credit Scoring and Database

Objetivos:

- Apresentar o a interfacer de comunicação com banco de dados DBI
- Apresentar o banco de dados SQLite
- Utilizar o algoritmo: k-Nearest Neighbour (KNN) para fazer a classificação de Crédito
- Avaliar a performance do modelo utilizando *confusion matrix*

Packages: `DBI,SQLite,class`

Arquivos: R/databases/KNN-NotasDisciplinas.v0.1.txt

KNN, Credit Scoring and Database

Exercícios

1. Conectar a Base de dados :

```
database/GermanCredit.sqlite.db
```

Utilizar o package DBI e o driver para o banco de dados SQLite (`RSQLite::SQLite()`) para se conectar a base de dados e ler integralmente o conteúdo da tabela: `DataDictionary`

Comando da interface DBI:

`conn = dbConnect(drv,…)`: Inicia a conexão com um banco de dados genérico

`dbDisconnect(conn)`: Fecha a conexão com o banco de dado

`dbReadTable(conn, name)`: Lê integralmente o conteúdo da tabela

KNN, Credit Scoring and Database

Exercício 01 - Solução

```
library(DBI);
library(RSQLite);

## Arquivo com o conteúdo da base de dados SQLite
pathfile = "../database/GermanCredit.sqlite.db"

## DBI: Abre a conexão
conn = dbConnect(RSQLite::SQLite(),pathfile);

## <executar queries e comandos com a conexão aberta> ###

## Grava a tabela o dicionário de dados
dbReadTable(conn,"DataDictionay");


## DBI: Fecha a Conexão
dbDisconnect(conn);
```

KNN, Credit Scoring and Database

Exercícios

2. Selecionar da Tabela de Diconário apenas as colunas que sejam do tipo:

`numerical,dicretized`

Utilizar o comando:

`rs = dbGetQuery(conn,query):` executa a query e retorna um *resultset* no formato *data.frame*.

3. Selecionar da Tabela de Dados de Crédito todos os registros da coluna `Creditability` e das colunas da coluna que sejam do tipo: `numerical, dicretized.`

Exercício 02 - Solução

```
library(DBI);
library(RSQLite);

## Arquivo com o conteúdo da base de dados SQLite
pathfile = "../database/GermanCredit.sqlite.db"

## DBI: Abre a conexão
conn = dbConnect(RSQLite::SQLite(),pathfile);

## Query
query = "select * from DataDictionay where ColType in ('discretized','numerical')";

## Grava a tabela o dicionário de dados
rs = dbGetQuery(conn,query);

## Imprime o resultado
print(rs);

## DBI: Fecha a Conexão
dbDisconnect(conn);
```

Exercício 03 - Solução

```
library(DBI);
library(RSQLite);

## Arquivo com o conteúdo da base de dados SQLite
pathfile = "../database/GermanCredit.sqlite.db"

## DBI: Abre a conexão
conn = dbConnect(RSQLite::SQLite(),pathfile);

## Query que seleciona colunas discretizadas e numéricas
query = "select * from DataDictionay where ColType in ('discretized','numerical')";

## Colunas dicretizadas e numéricas
rs = dbGetQuery(conn,query);

## Query que seleciona colunas de dados
query = sprintf("select Creditability, %s from CreditData",paste(rs[["ColName"]],collapse=(","

## Colunas dicretizadas e numéricas
rs = dbGetQuery(conn,query);

## Imprime o resultado
print(rs);

## DBI: Fecha a Conexão
dbDisconnect(conn);

## Base de dados
data = rs
```

KNN, Credit Scoring and Database

Exercícios

4. *Data preparation:*

- Isolar a coluna `Creditability` que será utilizada com o classes para classificação e normalizar as demais colunas para valore entre 0 e 1 utizando a fórmula:

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

- Transformar a coluna `Creditability` em categorias: 0 = “bad” 1 = “good”
- Separa aleatóriamente a base de dados em base de treinamento `data.train` com 60% da amostra e a base de testes `data.test` com 40% da amostra
- Para utilizar a função `knn(.)` isolar as colunas de classes correspondentes a base de treinamento e de testes

KNN, Credit Scoring and Database

Exercícios

5. *Classificação:*

- Fazer a classificação do crédito utilizando o algorírmo KNN implementado na package `class`:

```
classes.predict= knn(train=data.train,
                     test=data.test,
                     cl=classes.train,
                     k=3);
```

- Utilizar o comando:

```
CrossTable(x = classes.test, y = classes.predict, prop.chisq=FALSE)
```

Do package `gmodels` para avaliar a performance