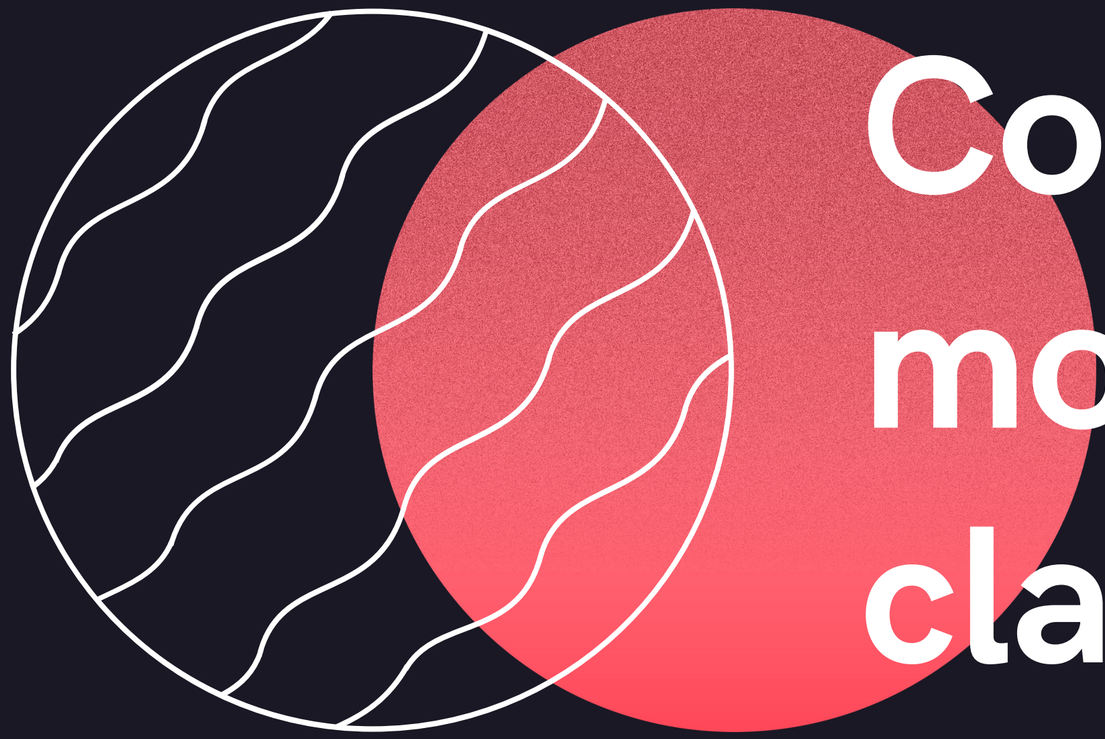


Comparação de modelos de classificação

Giovana Martinelli e Jonatas Amorim

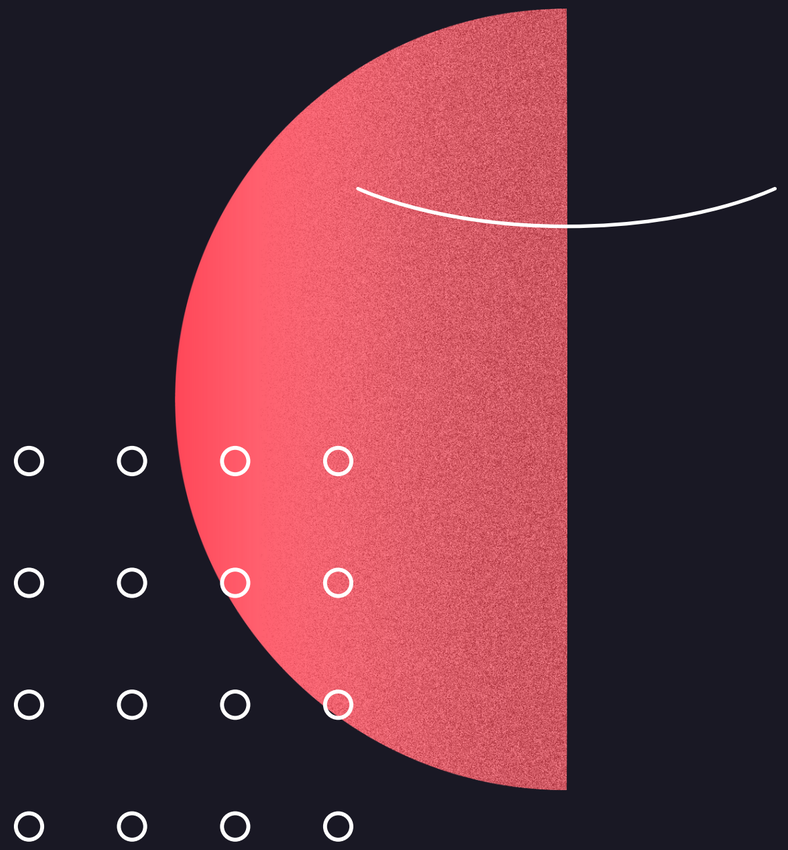
Modelos de classificação supervisionada

Existem diversas metodologias já desenvolvidas e também em desenvolvimento de modelos de classificação supervisionada.

A melhor metodologia é aquela que se ajusta melhor aos seus dados e que é capaz de trazer as melhores respostas de acordo com o seu objetivo.

03

Regressão Logística



Segundo alguns pesquisadores (Hosmer, Lemeshow, Cox e Snell), a Regressão Logística é uma metodologia que teve suas primeiras aplicações em 1967 em um estudo do risco de doença coronária.

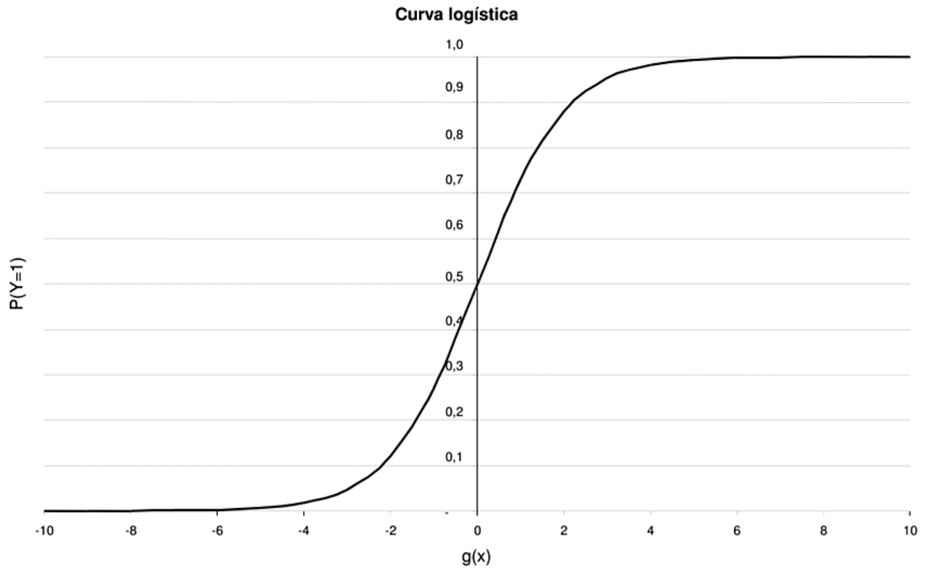
Essa metologia é uma das mais conhecidas nos problemas de classificação e é aplicada até hoje em várias áreas de estudo.

A regressão logística consiste em uma equação matemática que busca estimar a probabilidade de ocorrência de um determinado eveto a partir de uma ou mais variáveis chamadas variáveis independentes.

Regressão Logística

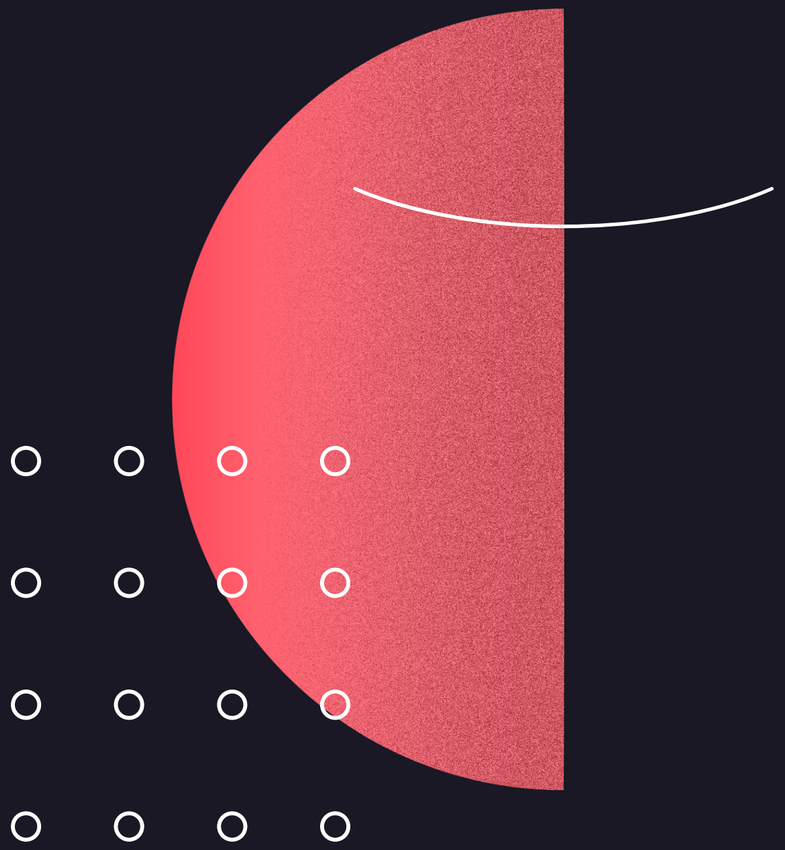
O modelo de regressão logística formado por p variáveis independentes pode ser escrito na seguinte forma:

$$P(Y=1) = \frac{1}{1+e^{-g(X)}} \quad \text{onde} \quad g(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$



05

Árvore de Decisão



Modelos baseados em árvores foram desenvolvidos na década de 1980 por Leo Breiman e associados e são bastante utilizados tanto para classicação quanto para regressão.

Na árvore de decisão, o processo de classificação é modelado com o uso de um conjunto de regras baseadas em uma ou mais variáveis independentes. As decisões são organizadas em uma estrutura parecida com uma árvore.

O critério de partição divide os dados em duas ou mais partes (nós) através do cálculo do ganho de informação. O ganho de informação, leva em consideração a impureza dos nós resultados da divisão. Os critérios de impureza mais comuns são a entropia e o índice Gini.

Árvore de Decisão

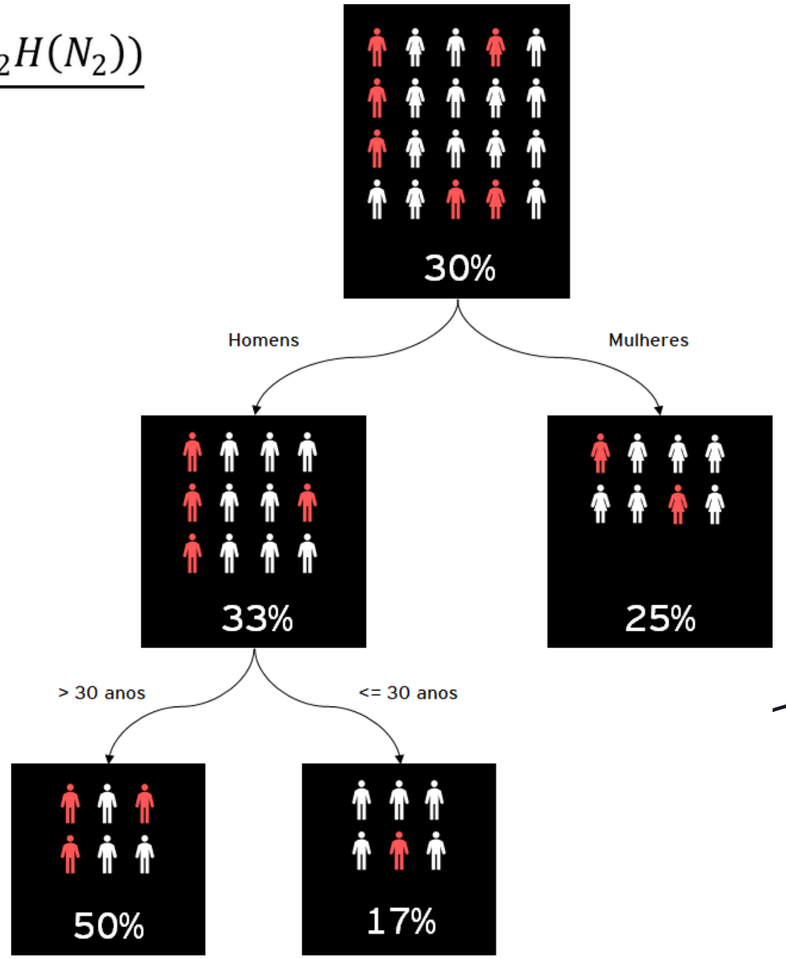
$$Ganho(N, N_1, N_2) = H(N) - \frac{(n_1 H(N_1) + n_2 H(N_2))}{n}$$

Onde H(N) é a impureza do nó N.

$$Entropia(N) = -\sum P(c|N)\log(P(c|N))$$

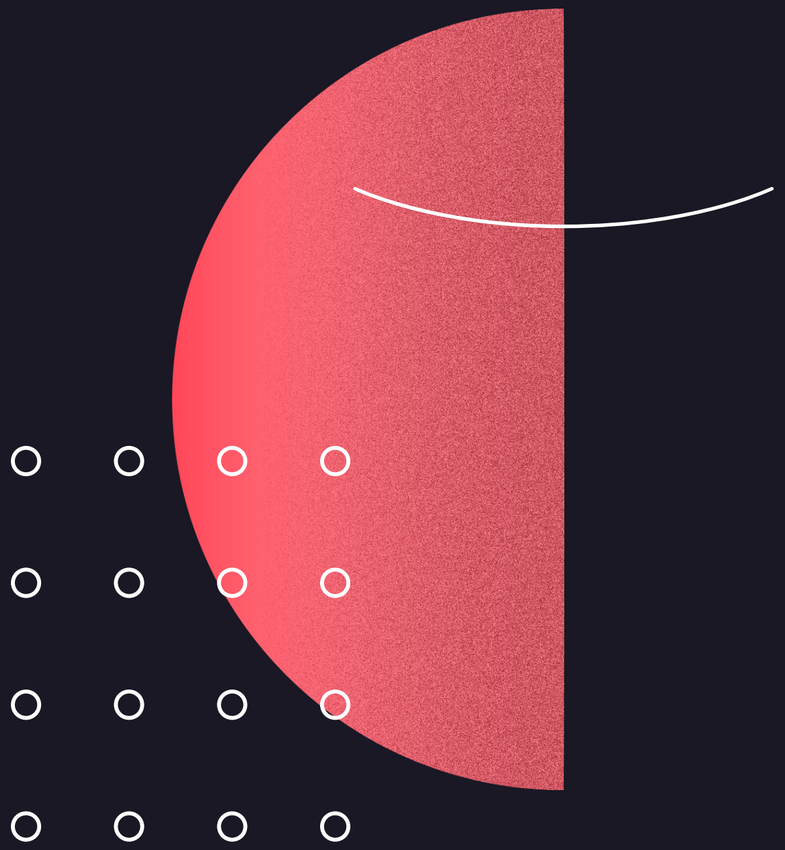
$$Gini(N) = -\sum P(c|N)(1 - (P(c|N))$$

Onde P(c|N) é a probabilidade de um ponto do nó N pertencer à classe c.



07

Floresta Aleatória



Os modelos de floresta aleatória são baseados em árvores de decisão e são muito utilizados tanto para classificação quanto para regressão. Estes algorítmos são conhecidos como aprendizado conjunto, pois são uma combinação de vários modelos.

Seu enfoque é baseado em um conjunto de B árvores com a utilização de diferentes conjuntos de p variáveis independentes na construção de cada uma das árvores.

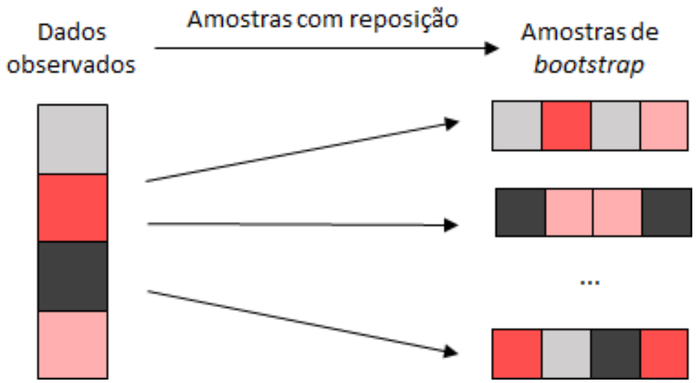
Utilizando uma analogia, a utilização desta técnica para a tomada de decisão corresponde à síntese da opinião de diversos indivíduos com diferentes fontes de informação sobre o problema em questão.

Se criarmos várias árvores e treinarmos todas elas no mesmo dataset, suas predições serão idênticas. Para contornar este problema existe o processos de *bootstrapping*.

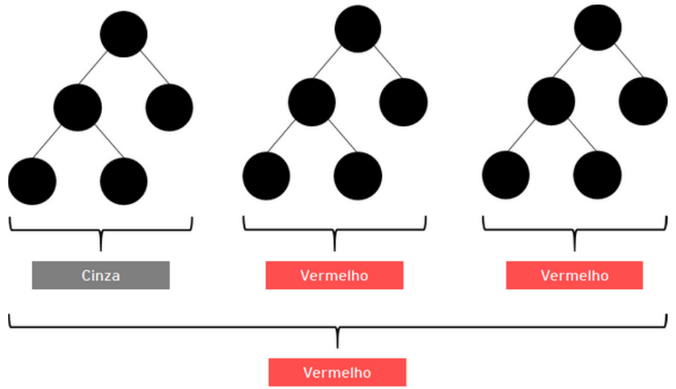
Floresta Aleatória

Além do fato das florestas aleatórias serem treinadas em diferentes conjuntos de dados, também é feita uma seleção aleatória das variáveis que estarão contidas nesse conjunto de dados.

Na construção de um novo nó, em vez de escolher a melhor variável dentre as p disponíveis no conjunto de treinamento, o algoritmo seleciona a melhor delas dentre um conjunto de m < p selecionadas ao acaso. Usualmente escolhe-se $m \approx \sqrt{p}$.



Bootstrapping



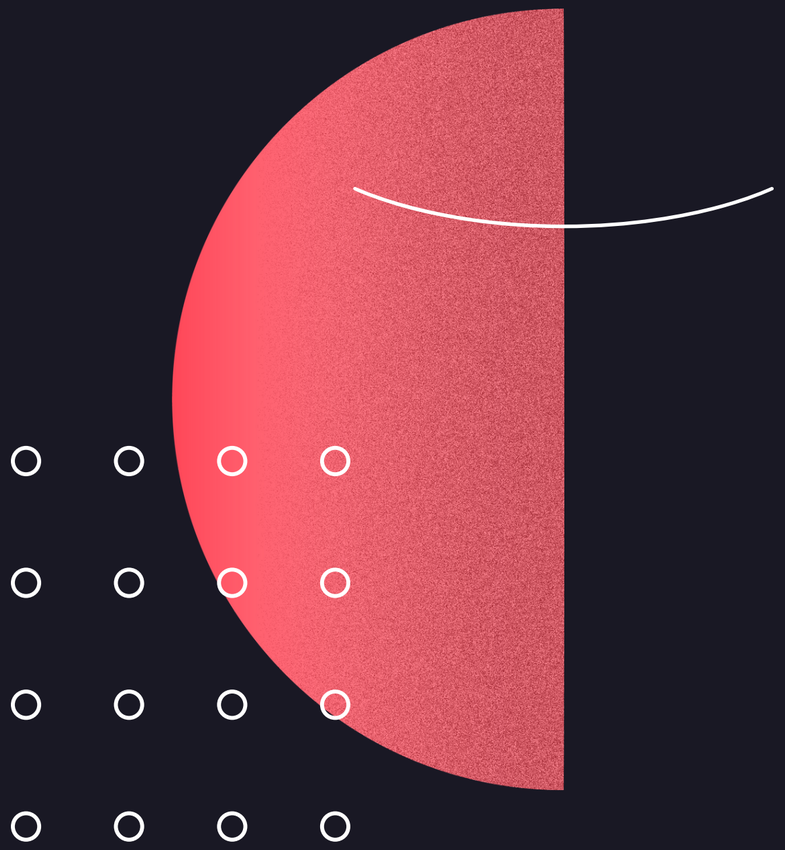
Bagging

Floresta Aleatória

Bagging

Random Forest

KNN



O método *K-Nearest Neighbors* (KNN) foi descrito inicialmente no início dos anos 50. Trata-se de um algorítmo que requer alto esforço computacional quando aplicado em grandes quantidades de dados, e por esse motivo ganhou popularidade apenas a partir dos anos 60.

É uma das técnicas de regressão / classificação mais simples, pois não há nenhum modelo a ser ajustado (como nas técnicas anteriores).

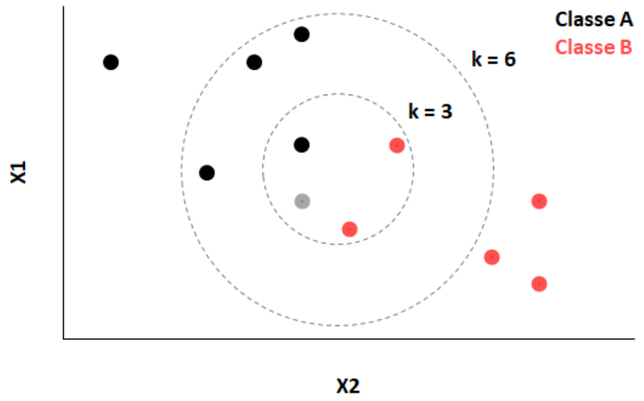
Isso não significa que o uso de KNN seja um procedimento automático. Os resultados da previsão dependem de como os recursos são dimensionados, como a similaridade é medida e quão grande K é definido. Além disso, todas as variáveis devem estar no formato numérico.

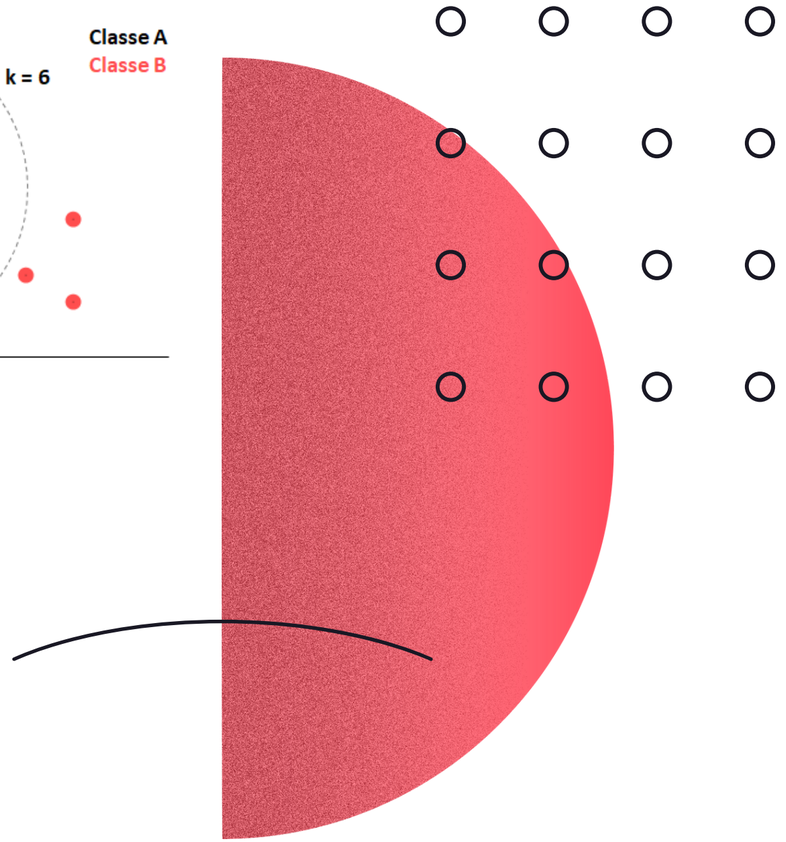
KNN

A idéia por trás do KNN é bem simples:

Para cada registro a ser classificado, o algoritmo:

1. Calcula as distâncias para todos os vizinhos;
2. Encontrar os K vizinhos mais próximos;
3. Classifica o registro de acordo com os K vizinhos.





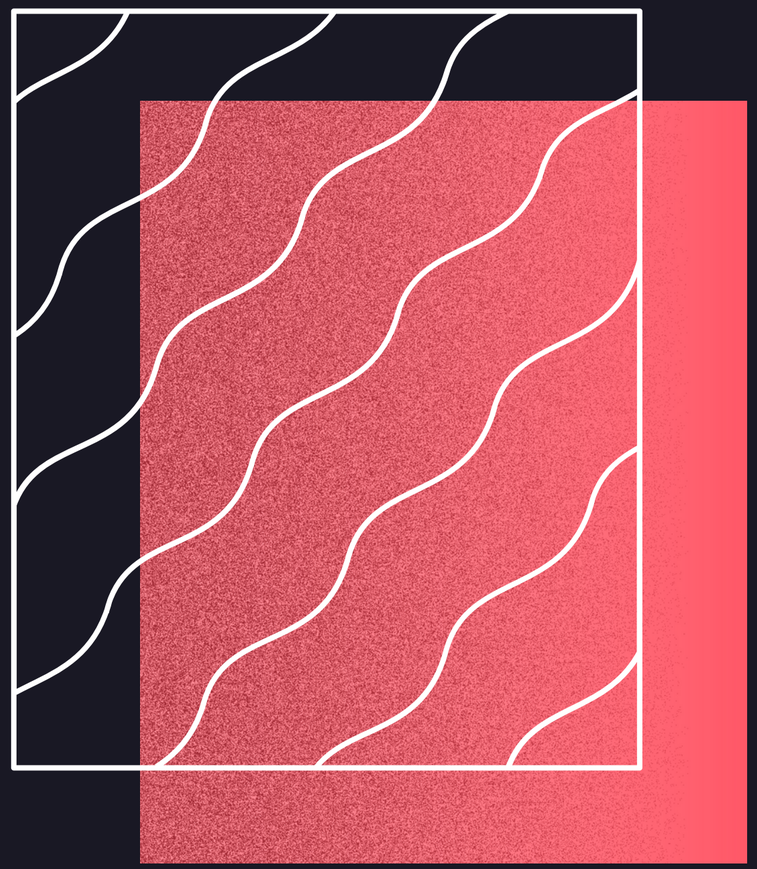
As distâncias podem ser calculadas a partir de várias metodologias:

1. Distância Euclidiana
2. Distância de Hamming
3. Distância Manhattan
4. Distância de Markowski

VANTAGENS E DESVANTAGENS

	Vantagens	Desvantagens
Regressão Logística	• Interpretabilidade das variáveis; • Aceita variáveis categóricas e numéricas; • Baixa probabilidade de *overfitting*.	• Nem sempre os dados se ajustarão bem à curva logística; • Requer um tratamento de dados mais cuidadoso (*missings* e *outliers*).
Árvore de Decisão	• Interpretabilidade das variáveis; • Modelo visual; • Aceita variáveis categóricas e numéricas; • Não necessita de um tratamento de dados mais cuidadoso, pois lida bem com *missings* e *outliers*.	• Grande probabilidade de *overfitting;* • Muitos parâmetros para ajuste.
Floresta Aleatória	• Controle de *overfitting*; • Aceita variáveis categóricas e numéricas; • Não necessita de um tratamento de dados mais cuidadoso, pois lida bem com *missings* e *outliers*.	• Interpretabilidade das variáveis; • Muitos parâmetros para ajuste.
KNN	• Algoritmo simples; • Baixa probabilidade de *overfitting*; • Poucos parâmetros para ajuste.	• Algoritmo lento se a quantidade de dados for grande; • Requer um tratamento de dados mais cuidadoso (*missings* e *outliers*); • Interpretabilidade das variáveis; • Aceita apenas variáveis numéricas.

MÉTRICAS



As métricas mais comuns para avaliação de modelos de classificação são:

Acurácia · Precisão · Revocação · F1-Score · KS

Todas estas métricas são calculadas com base na matriz de confusão:

		Real 0	Real 1
Predito	0	VN	FN
Predito	1	FP	VP

MÉTRICAS

		Real 0	Real 1
Predito	0	VN	FN
Predito	1	FP	VP

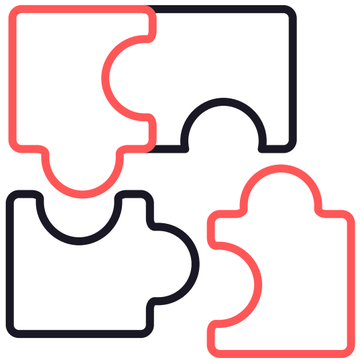
$$Acurácia = \frac{VP + VN}{Total}$$

$$Revocação = \frac{VP}{VP + FN}$$

$$Precisão = \frac{VP}{VP + FP}$$

$$F1 - Score = 2 \times \frac{Precisão \times Revocação}{Precisão + Revocação}$$

$$KS = \frac{VP}{VP + FN} + \frac{VN}{VN + FP} - 1$$



DADOS ESTUDADOS

Analisamos dois conjuntos de dados com o objetivo de criar 4 modelos para cada conjunto (um para cada uma das 4 metodologias) e comparar os resultados entre as metodologias.

Classificação de drogas

O primeiro conjunto de dados analisado foi sobre classificação do tipo de droga utilizada pelos pacientes.

Os dados foram extraídos do Kaggle e contam com algumas informações relacionadas às características dos pacientes, como idade, sexo, pressão sanguínea, colesterol, etc.

https://www.kaggle.com/prathamtripathi/drug-classification/discussion/188540

Ataque cardíaco

O segundo conjunto de dados analisado foi sobre pacientes que tiveram ou não ataque cardíaco.

Os dados foram extraídos do Kaggle e contam com algumas informações relacionadas às características dos pacientes, como idade, sexo, dor no peito, colesterol, etc.

https://www.kaggle.com/rishidamarla/heart-disease-prediction

CONSTRUÇÃO DOS MODELOS

As análises foram construídas no Python através dos seguintes passos:

1. Tratamento dos dados
2. Análise exploratória
3. Separação em dataset de treino e teste
4. Construção do modelo
    a. Regressão Logística
    b. Árvore de Decisão
    c. Floresta Aleatória
    d. KNN
5. Validação do modelo

Principais bibliotecas utilizadas:

- Pandas
- Statsmodels
- Pandas Profiling
- Scikit Learn

As análises foram construídas no Python através dos seguintes passos:

1. Tratamento dos dados
2. Análise exploratória
3. Separação em dataset de treino e teste
4. Construção do modelo
    a. Regressão Logística
    b. Árvore de Decisão
    c. Floresta Aleatória
    d. KNN
5. Validação do modelo

CONSTRUÇÃO DOS MODELOS

Os dados foram importados e tratados através da biblioteca Pandas e a análise exploratória foi realizada pela biblioteca Pandas Profiling.

Os tratamentos realizados foram conversões de variáveis categóricas para numéricas.

```
import pandas as pd
import pandas_profiling

df = pd.read_csv('dataset.csv')
profile = pandas_profiling.ProfileReport(df)

df['HD'] = df['HD'].apply(lambda x: 1 if x == 'Yes' else 0)
```

As análises foram construídas no Python através dos seguintes passos:

1. Tratamento dos dados
2. Análise exploratória
3. Separação em dataset de treino e teste
4. Construção do modelo
    a. Regressão Logística
    b. Árvore de Decisão
    c. Floresta Aleatória
    d. KNN
5. Validação do modelo

CONSTRUÇÃO DOS MODELOS

Os dados foram separados em dataset de treino (70%) e teste (30%) através da biblioteca Scikit Learn.

```
from sklearn.model_selection import
train_test_split

X = df.drop(columns = ['HD'])
y = df['HD']

X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size = 0.3)
```

As análises foram construídas no Python através dos seguintes passos:

1. Tratamento dos dados
2. Análise exploratória
3. Separação em dataset de treino e teste
4. Construção do modelo
    a. Regressão Logística
    b. Árvore de Decisão
    c. Floresta Aleatória
    d. KNN
5. Validação do modelo

CONSTRUÇÃO DOS MODELOS

Os modelos de Regressão Logística foram construídos a partir da biblioteca Statsmodels.

```
import statsmodels.api as sm

X_sm = sm.add_constant(X_train)
clf= sm.Logit(y_train, X_sm).fit()
clf.summary()
```

De forma interativa, realizou-se a redução de variáveis afim de obter um modelo apenas com as variáveis significativas.

As análises foram construídas no Python através dos seguintes passos:

1. Tratamento dos dados
2. Análise exploratória
3. Separação em dataset de treino e teste
4. Construção do modelo
    a. Regressão Logística
    b. Árvore de Decisão
    c. Floresta Aleatória
    d. KNN
5. Validação do modelo

CONSTRUÇÃO DOS MODELOS

Os modelos de Árvore de Decisão foram construídos a partir da biblioteca Scikit Learn.

Foi construído um dicionário p_grid com diversas opções para alguns hiperparâmetros do modelo afim de realizar o Tuning do algorítimo e determinar quais os melhores hiperparâmetros.

```
from sklearn import tree

clf = tree.DecisionTreeClassifier()
grid = GridSearchCV(clf, p_grid, scoring =
'accuracy')
clf = grid.fit(X_train, y_train)
clf.best_params_
```

As análises foram construídas no Python através dos seguintes passos:

1. Tratamento dos dados
2. Análise exploratória
3. Separação em dataset de treino e teste
4. Construção do modelo
    a. Regressão Logística
    b. Árvore de Decisão
    c. Floresta Aleatória
    d. KNN
5. Validação do modelo

CONSTRUÇÃO DOS MODELOS

Os modelos de Floresta Aleatória foram construídos a partir da biblioteca Scikit Learn.

Foi construído um dicionário p_grid com diversas opções para alguns hiperparâmetros do modelo afim de realizar o Tuning do algorítimo e determinar quais os melhores hiperparâmetros.

```
from sklearn.ensemble import
RandomForestClassifier

clf = RandomForestClassifier()
grid = GridSearchCV(clf, p_grid, scoring =
'accuracy')
clf = grid.fit(X_train, y_train)
clf.best_params_
```

As análises foram construídas no Python através dos seguintes passos:

1. Tratamento dos dados
2. Análise exploratória
3. Separação em dataset de treino e teste
4. Construção do modelo
    a. Regressão Logística
    b. Árvore de Decisão
    c. Floresta Aleatória
    d. KNN
5. Validação do modelo

CONSTRUÇÃO DOS MODELOS

Os modelos de KNN foram construídos a partir da biblioteca Scikit Learn.

Foi construído um dicionário p_grid com diversas opções para o número de vizinhos a serem considerados no modelo afim de realizar o Tuning do algorítimo e determinar qual o melhor valor.

```
from sklearn.neighbors import
KNeighborsClassifier

clf = KNeighborsClassifier()
grid = GridSearchCV(clf, p_grid, scoring =
'accuracy')
clf = grid.fit(X_train, y_train)
clf.best_params_
```

CONSTRUÇÃO DOS MODELOS

As análises foram construídas no Python através dos seguintes passos:

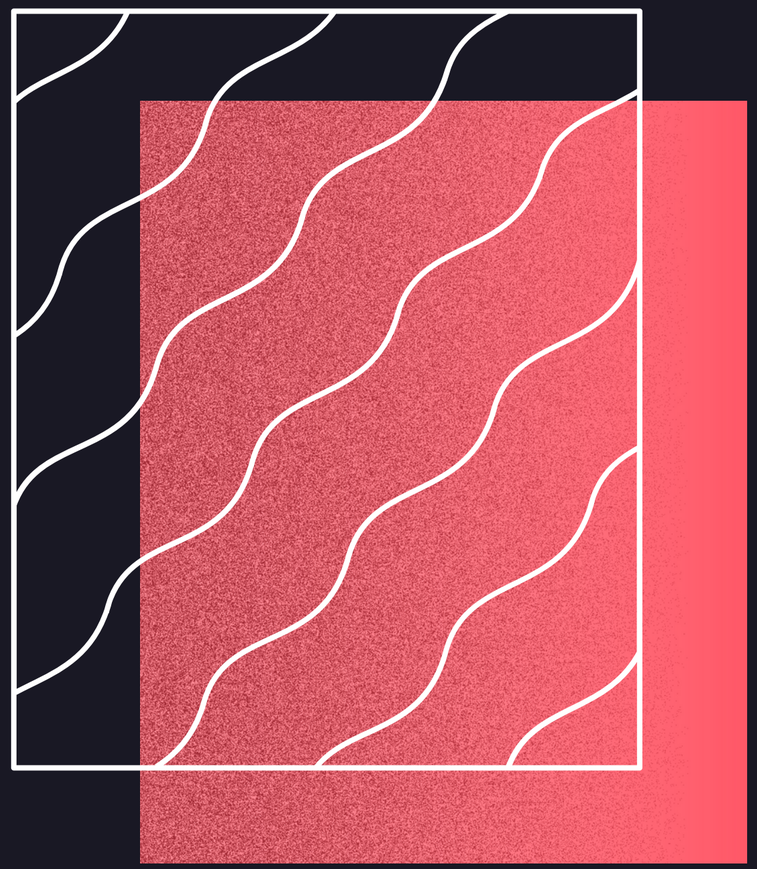
1. Tratamento dos dados
2. Análise exploratória
3. Separação em dataset de treino e teste
4. Construção do modelo
    a. Regressão Logística
    b. Árvore de Decisão
    c. Floresta Aleatória
    d. KNN
5. Validação do modelo

Os modelos foram validados a partir da biblioteca Scikit Learn.

```
from sklearn import metrics

previsoes = clf.predict(X_test)
previsoes = pd.DataFrame(previsoes)
previsoes.columns = ['Classe']

mc = metrics.confusion_matrix(y_test,
previsoes['Classe'])
KS = mc[1, 1] / mc[1].sum() + mc[0, 0] /
mc[0].sum() - 1

metrics.classification_report(y_test,
previsoes['Classe'])
```

RESULTADOS

	Métrica	Regressão Logística	Árvore de Decisão	Floresta Aleatória	KNN	Melhor Performance
Ataque Cardíaco	Acurácia	83%	78%	81%	67%	Regressão Logística
	Revocação	79%	68%	71%	68%	Regressão Logística
	Precisão	83%	81%	87%	63%	Floresta Aleatória
	F1-Score	81%	74%	78%	66%	Regressão Logística
	KS	65%	54%	62%	34%	Regressão Logística
Classificação de Drogas	Acurácia	82%	87%	95%	80%	Floresta Aleatória
	Revocação	71%	100%	94%	41%	Árvore de Decisão
	Precisão	67%	68%	89%	78%	Floresta Aleatória
	F1-Score	69%	81%	91%	54%	Floresta Aleatória
	KS	57%	81%	89%	35%	Floresta Aleatória



CONCLUSÕES

Não existe um algoritmo melhor que outro. O melhor algoritmo varia de acordo com vários atributos, dentre eles:

- Objetivos do projeto;
- Conjunto de dados;
- Tipo das variáveis.

Lidando com clientes:

- Se o cliente não tem muito conhecimento dos algoritmos, é interessante utilizar uma metodologia fácil de ser explicada;
- Muitas vezes o cliente tem interesse em entender o comportamento das variáveis. Neste caso, é importante utilizar uma metodologia que possibilite isso também.

Obrigad@!