

Técnicas Tradicionais de Classificação de Imagens

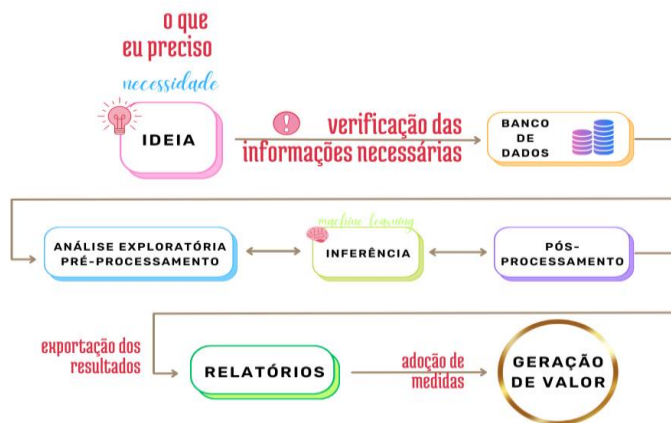
Classificação: Decision Trees e Random Forest



Recapitulação

Esquema Básico

Projeto de Visão Computacional



- Desbalanceamento
- Visualização
- Medidas resumo: características
- Metadados de captura
- Metadados técnicos
- Redução de Dimensionalidade (tópico visto em maior profundidade na aula passada com o Leonardo)
- Redimensionamento x Patches x Crop
- Análise de Variância
- Data Augmentation (ainda a ser muito estudado e praticado)
- Histogram Matching vs Histogram Equalization

Esquema Básico

Treinamento/Inferência

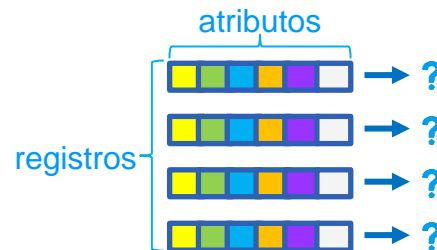
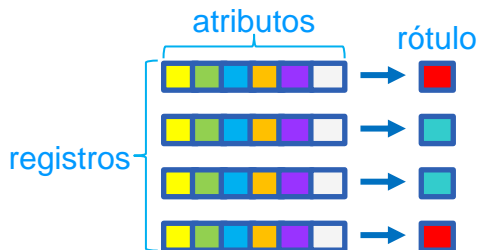
Machine Learning

Supervisionado

Não Supervisionado

Classificação
Regressão
Previsão de séries temporais

Agrupamento
Associação



Esquema Básico

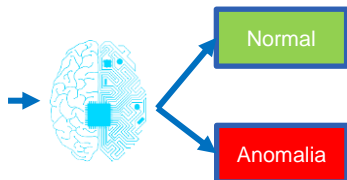
Aprendizado Supervisionado

Classificação

Rótulo é categórico.



Imagens de um equipamento



Regressão

Rótulo é contínuo.



Imagens de um equipamento

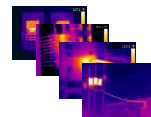


Previsão de Séries

Rótulo é contínuo e dependente do tempo.

Temperatura
Ou estado

Dados
históricos

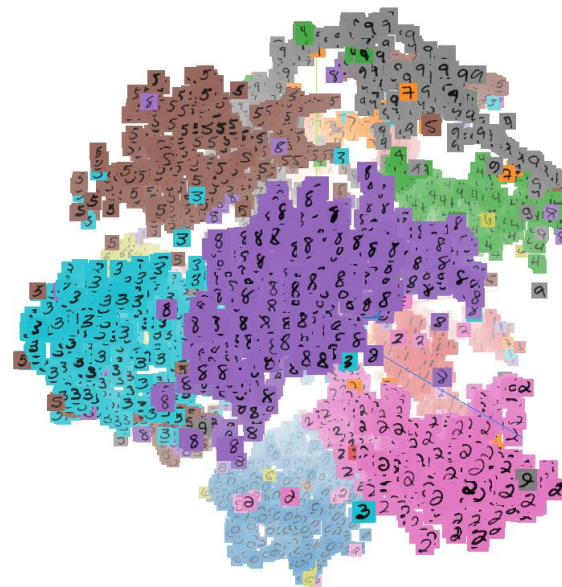
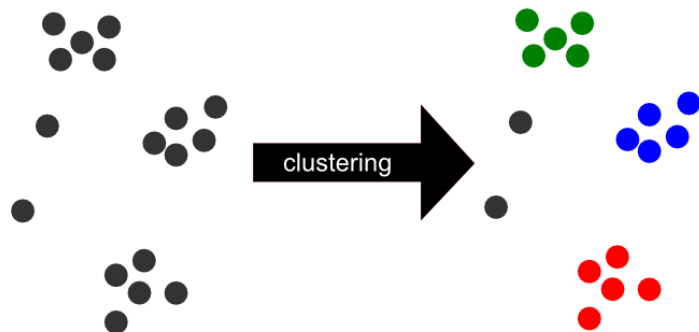


Esquema Básico

Aprendizado Não Supervisionado

Agrupamento

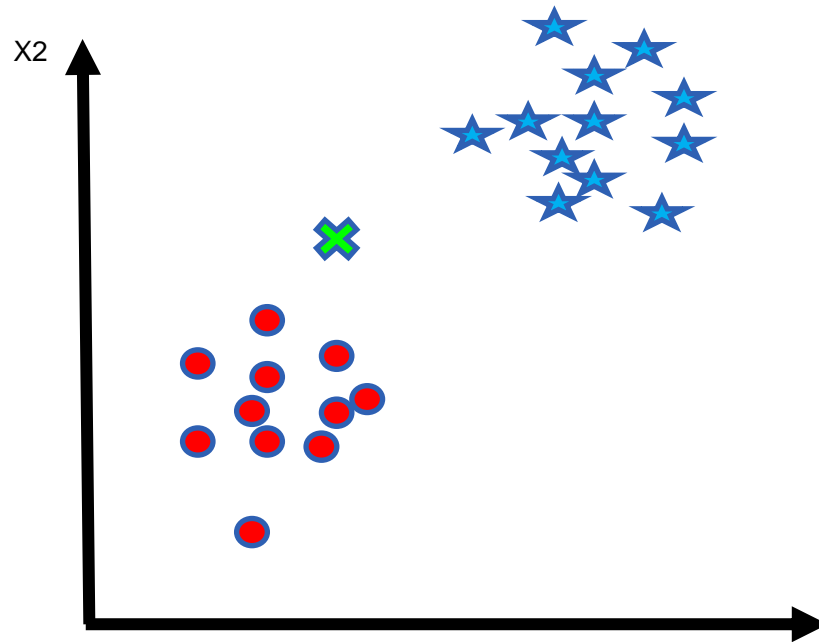
Descoberta de semelhanças e grupos entre registros.



KNN

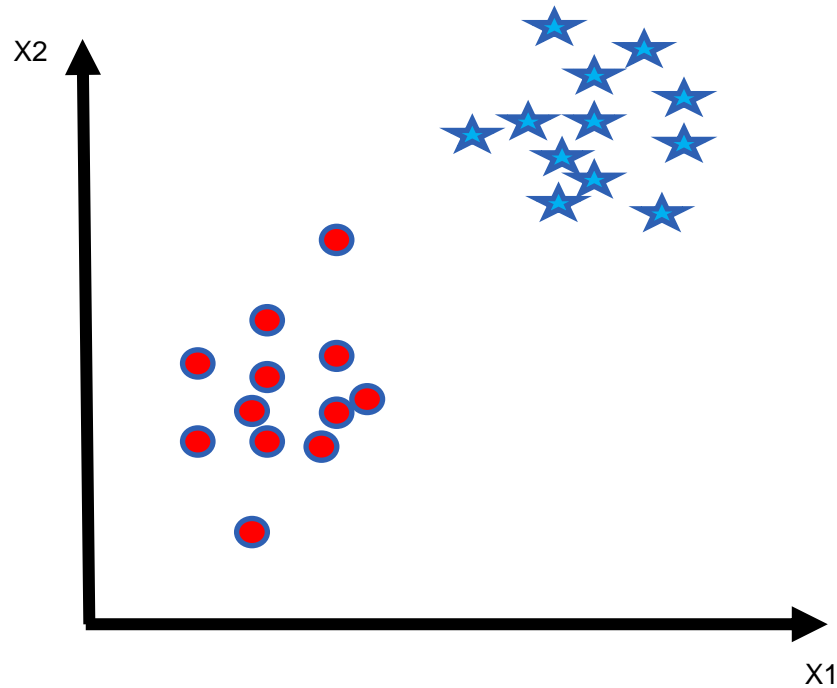
Classificação

KNN



Classificação

KNN



Estudo de Caso

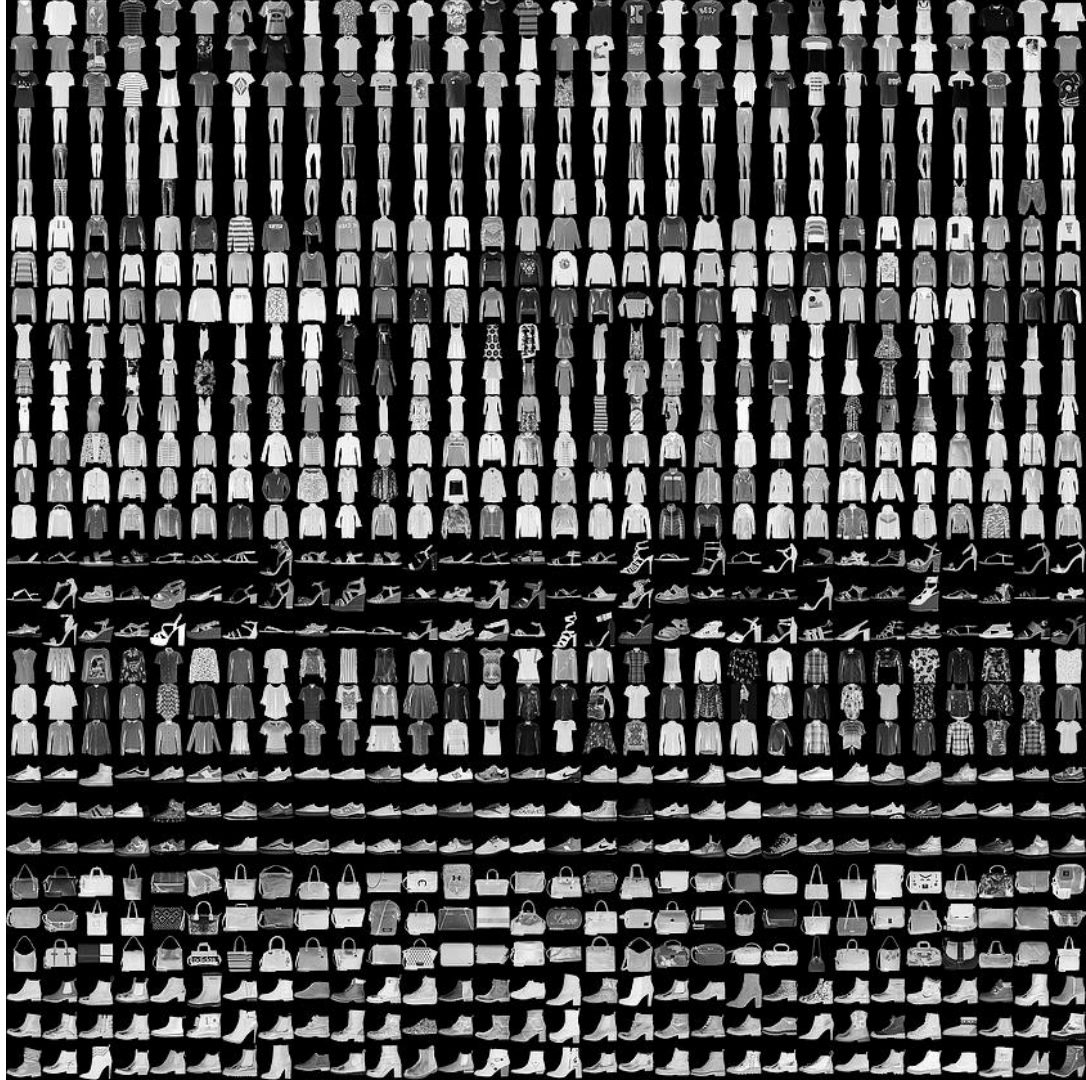
Classificação – Estudo de Caso

KNN

- Fashion-MNIST
 - 60k imagens de treino
 - 10k imagens de teste
 - 10 classes
 - 28x28
 - 8 bits
 - Grayscale

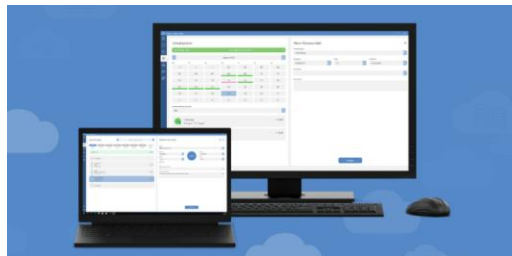
Curiosidades:

- Introdução de uma base de imagens 28 x28 (8 bits) grayscale com um pouco mais de complexidade que o dataset original Mnist.



Deploy com API

Deploy API



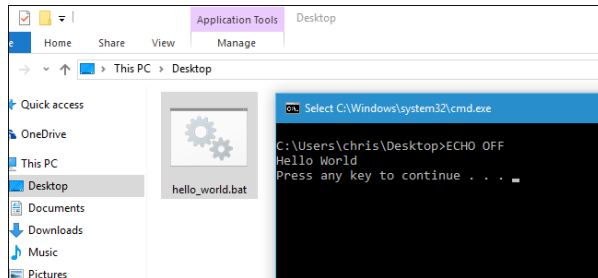
App desktop ou web

Executável por linha de comando

```
C:\Windows\system32\cmd.exe

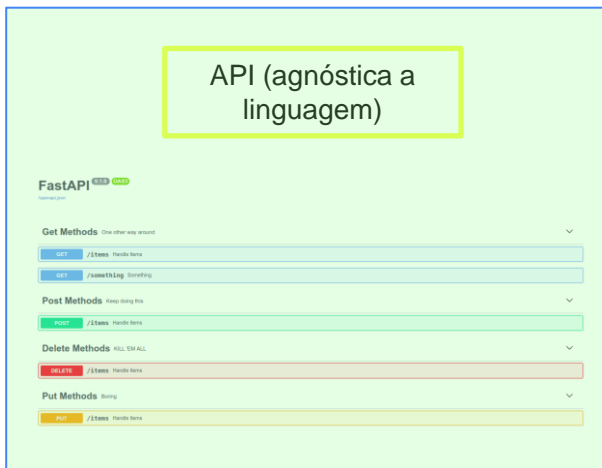
codemantic -help
usage: codemantic [-command] [-path] [-option] [-value] [-option] [-value]]
[command]:
  Config      Create or modify a project file with the specified options.
  Generate    Generates the app specified in the project file.
  Refresh     Refreshes metadata and generated the app. Data models and
              controllers are re-created.
[option]:
  AppName      Specifies a user-friendly application name.
  BatchEdit    Enables batch editing in the app.
  ClassLibrary Enables packaging of app framework in class library.
  Copyright    Specifies copyright text.
  DBConn       Enables ODBC (Database Management System) in the database.
  DBConnection Specifies a database connection string for the app.
  DBMembership Enables user and role management in the database.
  DBProvider   Specifies data provider (Oracle, ODBC, SQLDatabase, etc.)
  DBSessionState Enables session state management in the database.
  DiscoveryPath Specifies a relationship discovery path for data model.
  Framework    Specifies the version of Microsoft .NET Framework for the app.
  Language     Specifies C# or VisualBasic for the app implementation.
  MultipleSelect Enables multiple selection in the app.
  NameSpace    Specifies a namespace for the app source code.
  NoGenerate    Generates database objects defined in the app.
  PageHeader   Specifies a header display in all pages of the app.
  PageType     Specifies HTML or ASP page implementation.
  Preview       Enables launching of default web browser to preview the app.
  UIInterface  Specifies TouchUI or Classic for the app user interface.
  Reports      Enables reporting based on Microsoft Report Viewer.
  ShowDates    Enables smart display of dates.
  Theme        Specifies the name of the app theme.

Use -OptionName [-Value] to change project options.
Option names are case-insensitive. Inter-semicolon spaces around option values
with "space" character. Unless "true" can be omitted for boolean options.
>
```



Bat script

API (agnóstica a linguagem)



Via código

Estamos em ambiente de produção.

1. Carregaremos o modelo treinado
2. Carregaremos novos dados a serem inferidos
3. Faremos as inferências
4. FIM

```
[ ] # 1. Carregaremos o modelo treinado
import pickle
filename = 'model.sav'
model = pickle.load(open(filename, 'rb'))

[ ] # 2. Carregaremos novos dados a serem inferidos
import pandas as pd
new_data = pd.read_csv('Dataset_spine_unknown.csv')
new_data.head()

[ ] # 3. Faremos as inferências
inferences = model.predict(new_data)
print(inferences)
```

FIM

Deploy API

- Queremos criar uma API (rodando localmente) usando o framework FastAPI com documentação e utilização através do OpenAPI (Swagger).
- Mais pra frente faremos Deploy no Heroku ou na Oracle!

VC MASTER - PUC-Rio

0.1.0 OAS 3.1

/openapi.json

default

POST /predict Predict

Parameters

CancelReset

No parameters

Request body *required*

multipart/form-data

file *required*

string(\$binary)

Escolher arquivo Bag.png

ExecuteClear

Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:5000/predict' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -F 'file=@Bag.png;type=image/png'
```

Request URL

http://127.0.0.1:5000/predict

Server response

Code	Details
200	<div>Response body<pre>{ "Prediction": ["Bag"] }</pre></div> <div>Response headers<pre>content-length: 22 content-type: application/json date: Fri, 05 Jul 2024 15:15:45 GMT server: uvicorn</pre></div>

Responses

Code	Description	Links
200	Successful Response	No links

Árvores de Decisão

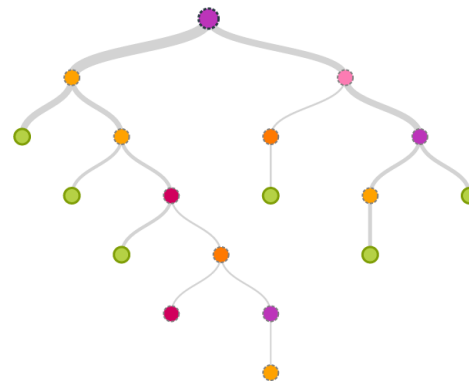
Classificação

Árvore de Decisão

- Árvores de decisão criam modelos de **classificação** na forma de estruturas;
- Quebra-se um conjunto de dados em menores e menores **subconjuntos** enquanto simultaneamente são criadas árvores de decisão associadas;
- O resultado final é uma **árvore com nós de decisão e nós folhas**;
- Fáceis de serem implementadas e interpretadas;

Classificação

Árvore de Decisão

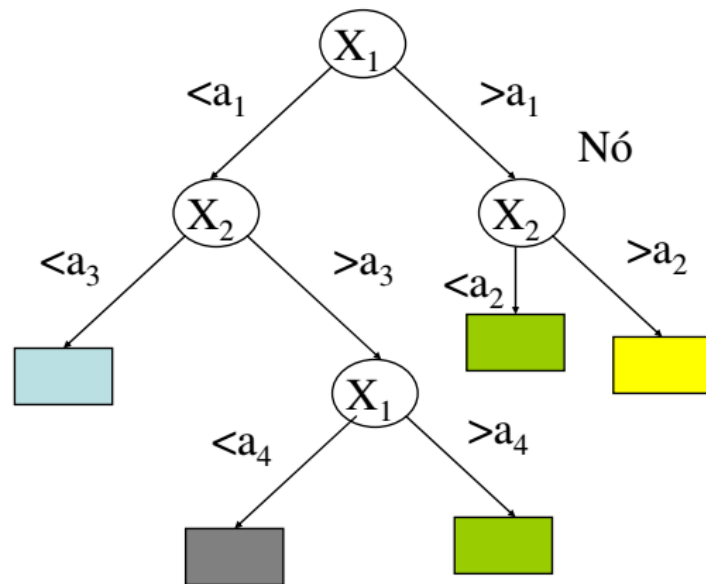


- Uma árvore de decisão possui um ou mais **ramos**;
- Nós **folhas** representam uma classificação ou decisão;
- O nó mais alto da árvore corresponde ao melhor classificador, chamado de nó **raiz**;
- Árvores de decisão podem lidar tanto com dados **numéricos** quanto **categóricos**.

Classificação

Árvore de Decisão

- Cada **nó de decisão** contém um **teste de um atributo**;
- Cada **folha** está associada a uma **classe**;
- Cada **percurso na árvore** (raiz à folha) corresponde a uma **regra de classificação**.

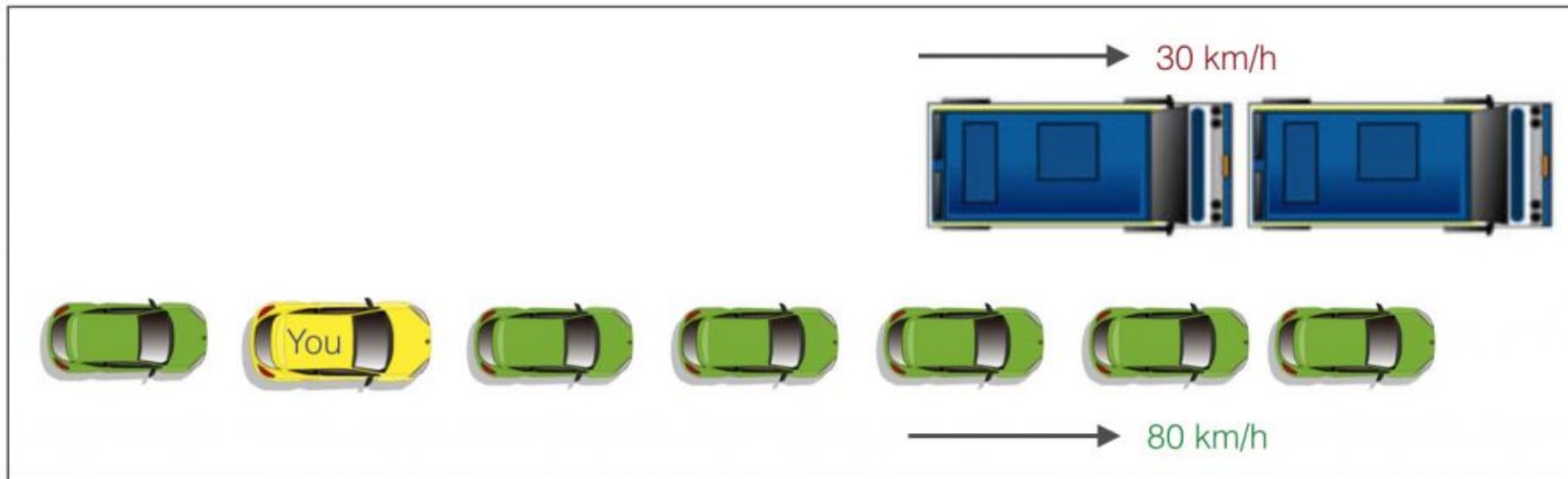


Classificação

Árvore de Decisão

Algoritmo ID3 (J. R. Quinlan)

- Busca gulosa de cima para baixo pelo espaço de possíveis ramos;



Classificação

Árvore de Decisão

Algoritmo ID3 (J. R. Quinlan)

- Busca gulosa de cima para baixo pelo espaço de possíveis ramos;
- Sem backtracking;
- Pode ficar preso em um ótimo local;
- Difícil de se usar em variáveis contínuas;

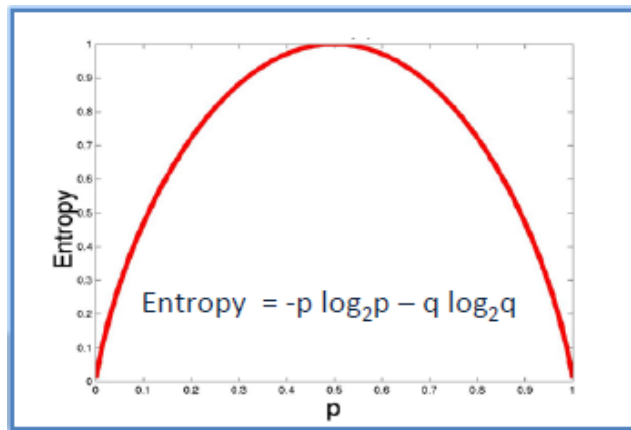


Classificação

Árvore de Decisão

Algoritmo ID3 (J. R. Quinlan)

- Utiliza a **entropia** para **calcular** a **homogeneidade** dos dados. Se os dados são completamente homogêneos, a entropia é zero. Se os dados estão divididos igualmente, a entropia é 1.



$$\text{Entropy} = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$$

Classificação

Árvore de Decisão



1. Calcula-se a entropia para classe (0 = dados homogêneos; 1 = dados igualmente distribuídos).

Play Golf	
Yes	No
9	5

$$\begin{aligned}
 \text{Entropy(PlayGolf)} &= \text{Entropy}(5,9) \\
 &= \text{Entropy}(0.36, 0.64) \\
 &= -(0.36 \log_2 0.36) - (0.64 \log_2 0.64) \\
 &= 0.94
 \end{aligned}$$

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

Classificação

Árvore de Decisão

2. Divide-se a base nos diferentes atributos e calcula-se a entropia para cada um deles. Calcula-se o ganho de informação (entropia para classe – entropia para atributo).

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14



$$\begin{aligned} E(\text{PlayGolf, Outlook}) &= P(\text{Sunny}) * E(3,2) + P(\text{Overcast}) * E(4,0) + P(\text{Rainy}) * E(2,3) \\ &= (5/14) * 0.971 + (4/14) * 0.0 + (5/14) * 0.971 \\ &= 0.693 \end{aligned}$$

$$\begin{aligned} G(\text{PlayGolf, Outlook}) &= E(\text{PlayGolf}) - E(\text{PlayGolf, Outlook}) \\ &= 0.940 - 0.693 = 0.247 \end{aligned}$$

O ganho de informação é baseado na diminuição da entropia depois que uma base de dados é subdividida em um atributo.

Classificação

Árvore de Decisão

2. Divide-se a base nos diferentes atributos e calcula-se a entropia para cada um deles. Calcula-se o ganho de informação (entropia para classe – entropia para atributo).

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			


O ganho de informação é baseado na diminuição da entropia depois que uma base de dados é subdividida em um atributo.

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

Classificação

Árvore de Decisão


3. Escolhe-se atributo com maior ganho de informação como nó de decisão.

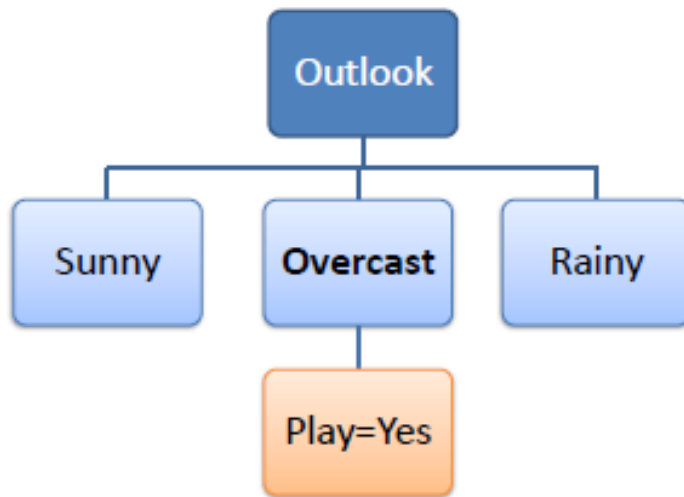
		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

Classificação

Árvore de Decisão

3. Escolhe-se atributo com maior ganho de informação como nó de decisão.
 - a. Um ramo que tenha entropia 0 é uma folha.

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

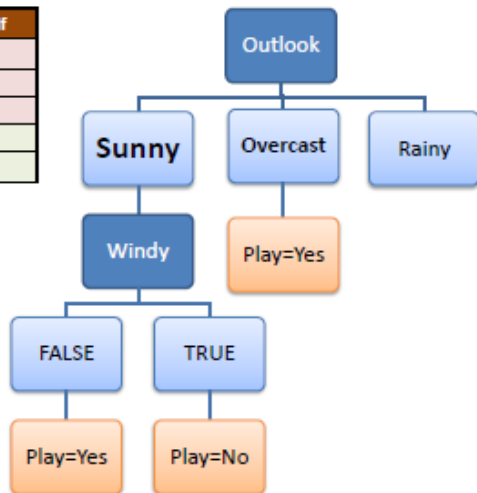



Classificação

Árvore de Decisão

3. Escolhe-se atributo com maior ganho de informação como nó de decisão.
 - a. Um ramo que tenha entropia 0 é uma folha.
 - b. Um ramo com entropia maior que 0 precisa ser subdividido.

Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

Classificação

Árvore de Decisão

4. Algoritmo é rodado recursivamente para todos os ramos sem folha até que todos os dados sejam classificados.

Regras de
decisão

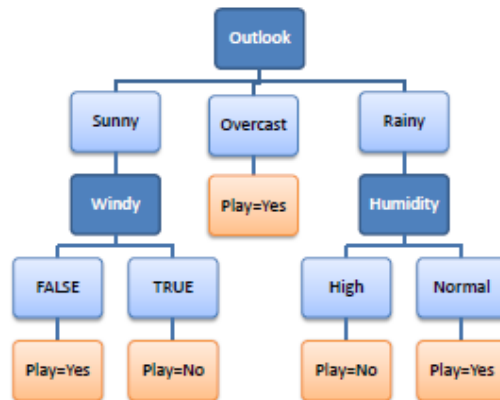
R_1 : IF (Outlook=Sunny) AND
(Windy=FALSE) THEN Play=Yes

R_2 : IF (Outlook=Sunny) AND
(Windy=TRUE) THEN Play=No

R_3 : IF (Outlook=Overcast) THEN
Play=Yes

R_4 : IF (Outlook=Rainy) AND
(Humidity=High) THEN Play=No

R_5 : IF (Outlook=Rain) AND
(Humidity=Normal) THEN
Play=Yes



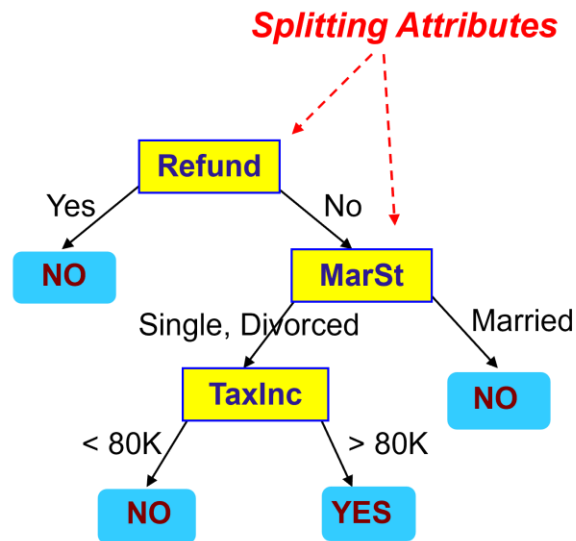
Classificação

Árvore de Decisão

- Exemplo

<i>Tid</i>	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data

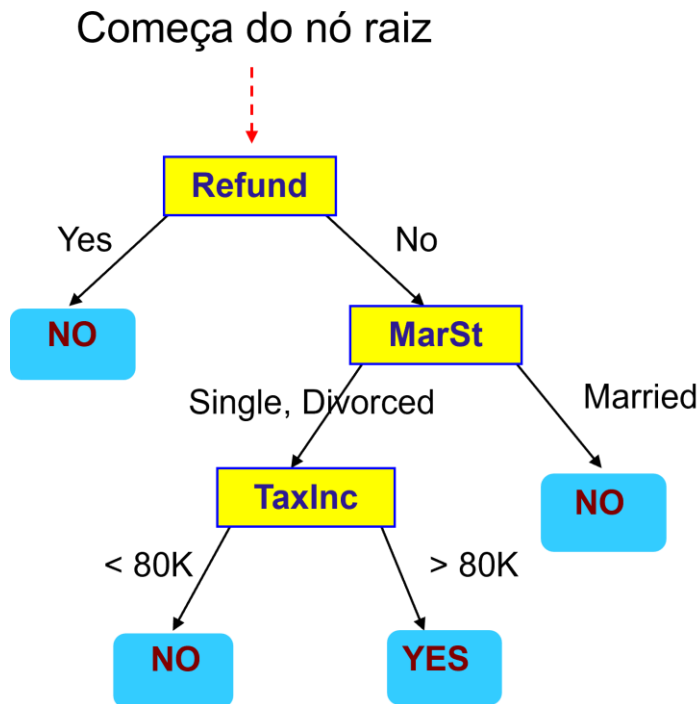


Model: Decision Tree

Classificação

Árvore de Decisão

- Exemplo



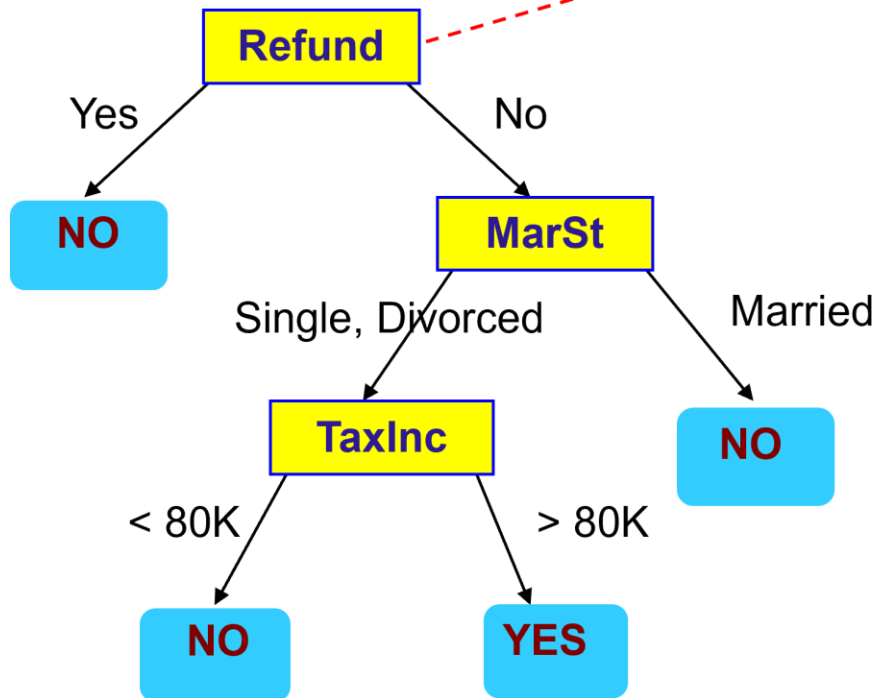
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Classificação

Árvore de Decisão

- Exemplo



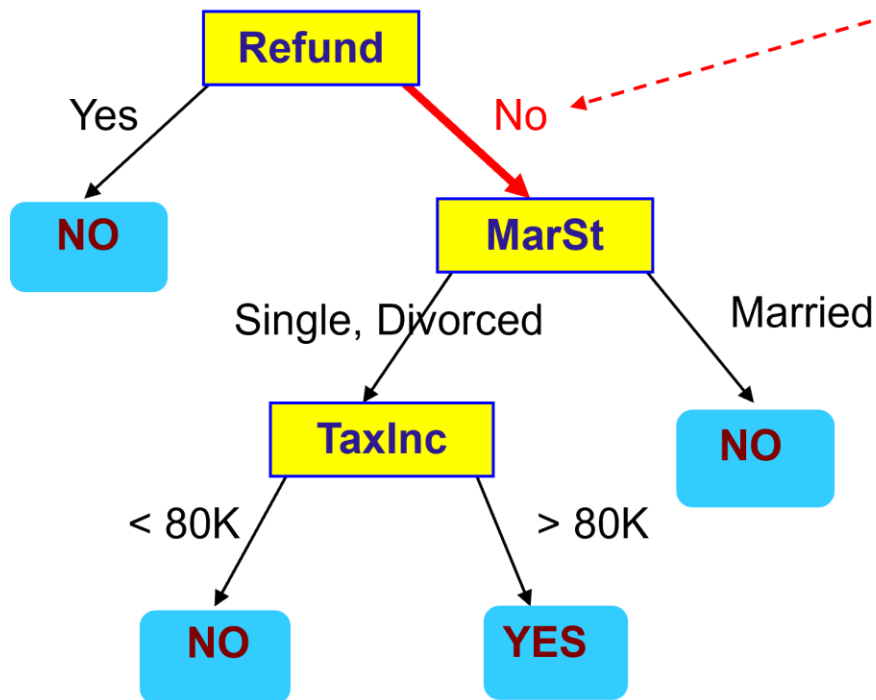
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Classificação

Árvore de Decisão

- Exemplo



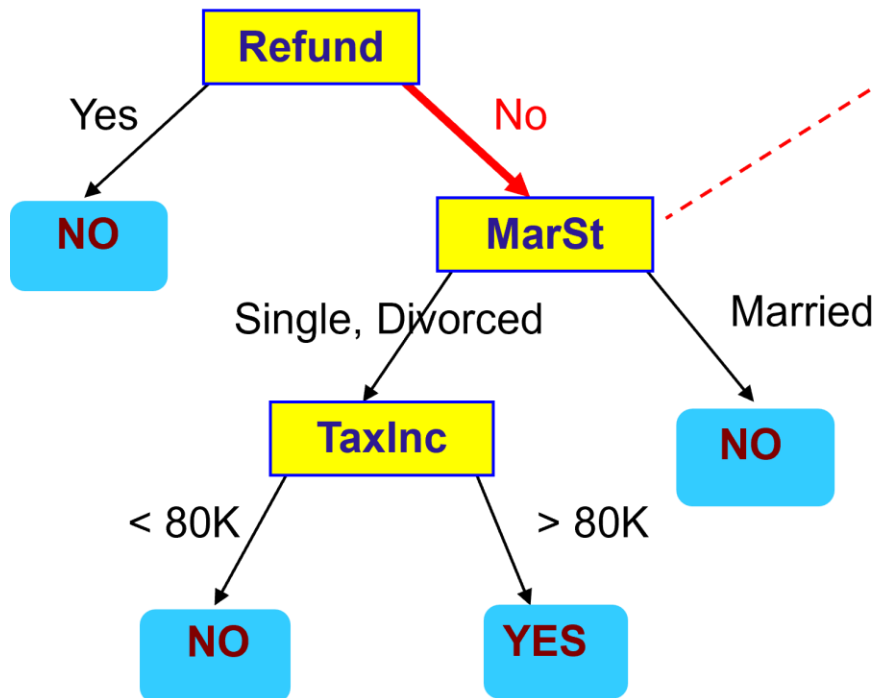
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Classificação

Árvore de Decisão

- Exemplo



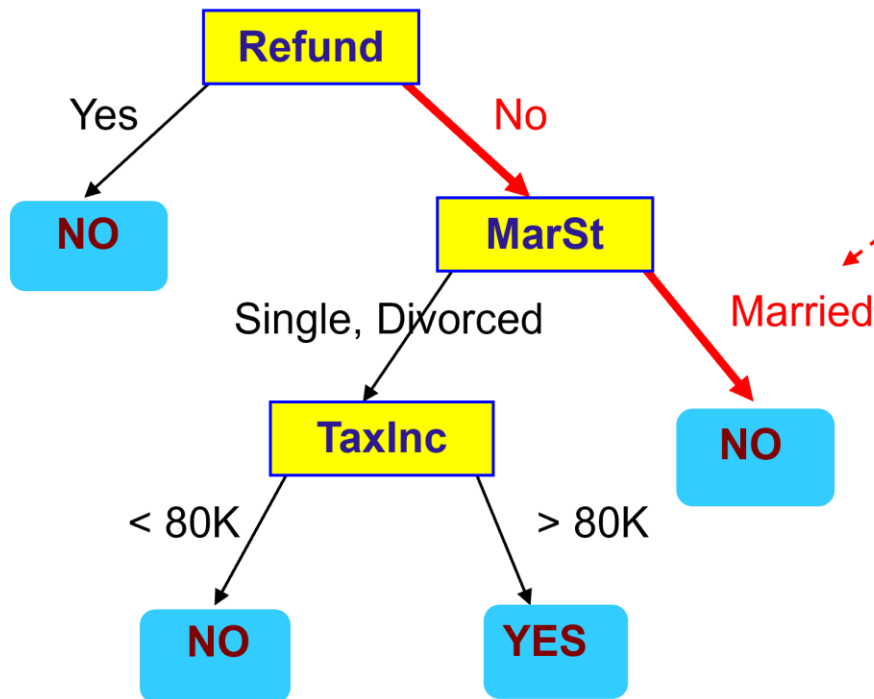
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Classificação

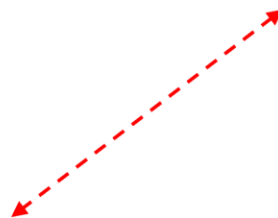
Árvore de Decisão

- Exemplo



Test Data

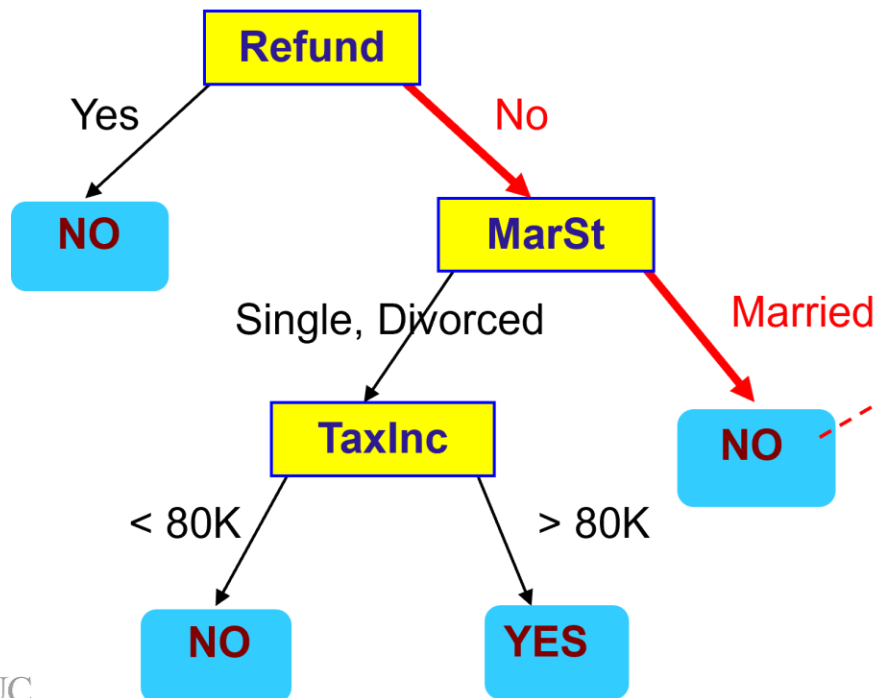
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Classificação

Árvore de Decisão

- Exemplo



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Valor inferido para
'Cheat' é 'No'

Classificação

Árvore de Decisão

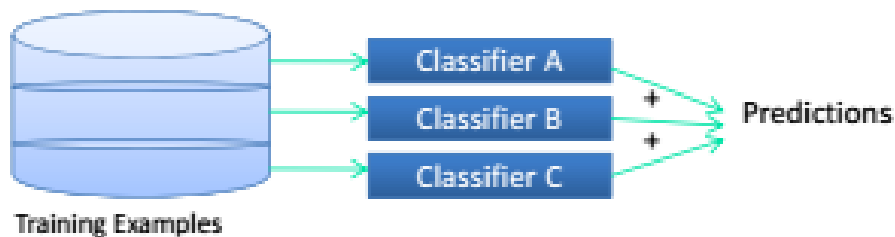
- Árvores de decisão são fáceis de se entender;
- Elas funcionam mais eficientemente com atributos discretos;
- Extremamente rápidas em classificar dados novos;

Comitê

Classificação

Comitê

- Agregar múltiplos modelos treinados com o objetivo de melhorar a desempenho do modelo conjunto.
- Intuição: simula o que fazemos quando combinamos conhecimento de especialistas em um processo de tomada de decisão.



Classificação

Comitê

Conhecidos também por:

- Comitês especialistas;
- Sistemas múltiplos de classificação;
- Comitê de classificadores;
- Máquina de Comitê;
- Mistura de especialistas;
- Aprendizado em conjunto;
- Ensemble.

Diversos estudos demonstram sua utilização com sucesso em problemas onde um único especialista não funciona bem.

Classificação

Comitê

Aprendizado de Comitês

- Às vezes cada técnica de aprendizado retorna diferentes 'hipóteses' (funções), mas nenhuma hipótese perfeita.
- Poderíamos combinar várias hipóteses imperfeitas para se ter uma hipótese melhor?

• Votação:

- Maioria do votos
- Maioria ponderada dos votos
- Borda count
- Média
- Média ponderada
- Soma
- Soma ponderada
- Produto
- Máximo
- Mínimo
- Mediana

Classificação

Comitê

Analogias:

- Comitês combinam opiniões de especialistas para tomar decisões melhores;
- Estudantes trabalhando em conjunto em um projeto.

Intuição:

- Indivíduos cometem erros, mas a maioria é menos propensa a erros;
- Indivíduos em geral têm conhecimento parcial. Um comitê pode juntar conhecimento para tomar decisões melhores.

Classificação

Comitê

Quando usar?

- Temos um conjunto muito grande de dados;
- A região de domínio do problema é muito complexa;
- Queremos melhorar os resultados de classificadores individuais.

Vantagens

- A combinação de modelos pode apresentar melhor desempenho que um modelo só;
- Neutraliza ou minimiza fortemente a instabilidade inerente aos algoritmos de aprendizagem;

Desvantagens

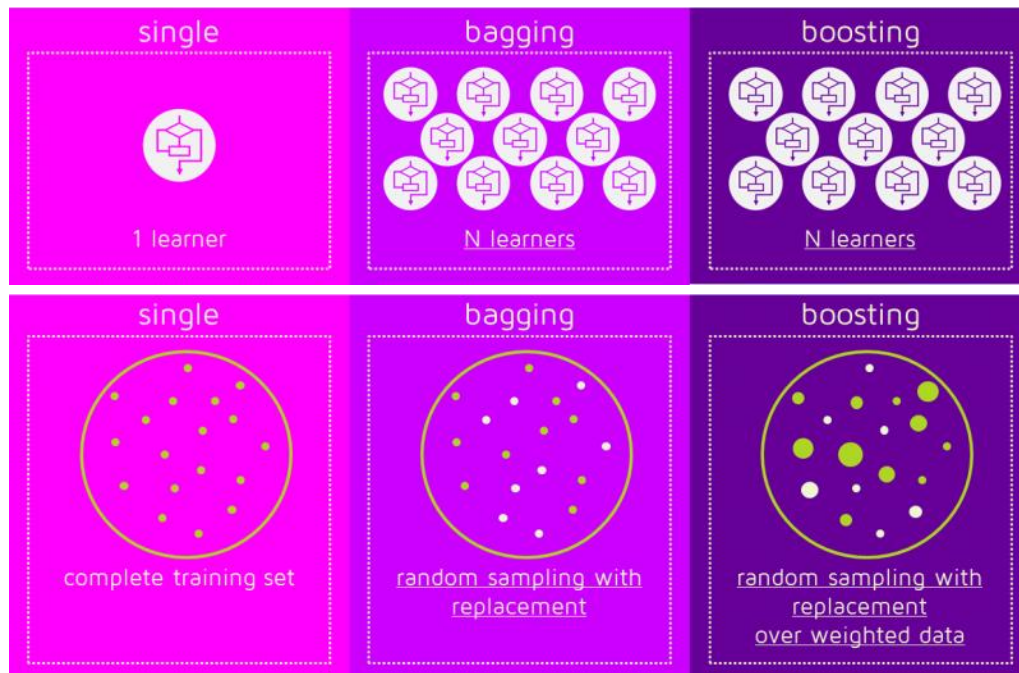
- Não há garantia de que as estruturas modulares apresentem os melhores resultados;
- Modelos combinados são mais difíceis de analisar;
- O custo é alto.

Comitê: Técnicas

Classificação

Comitê - Técnicas

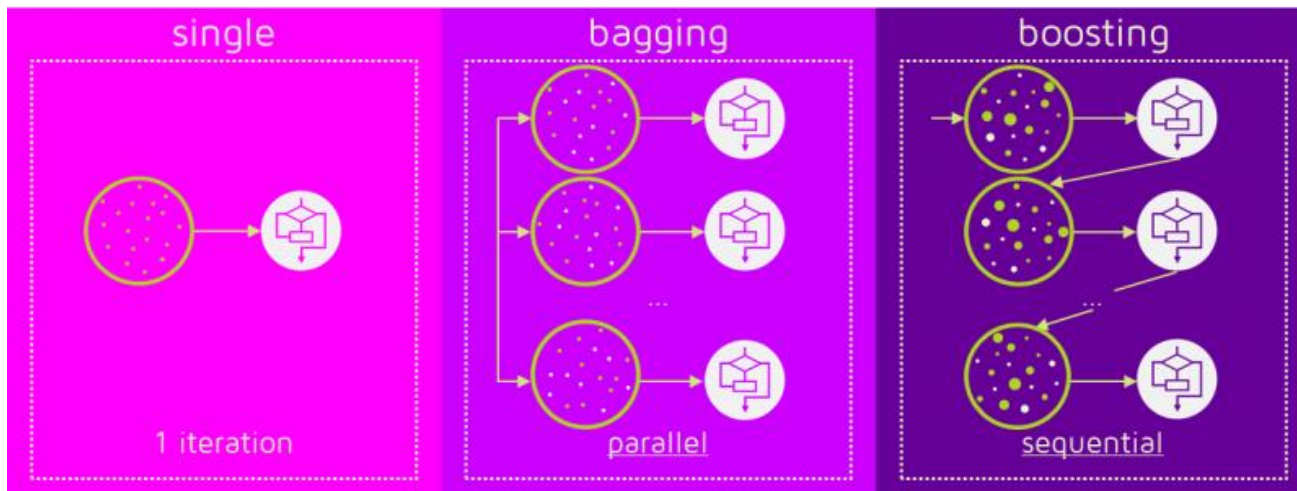
- Bagging vs. Boosting



Classificação

Comitê - Técnicas

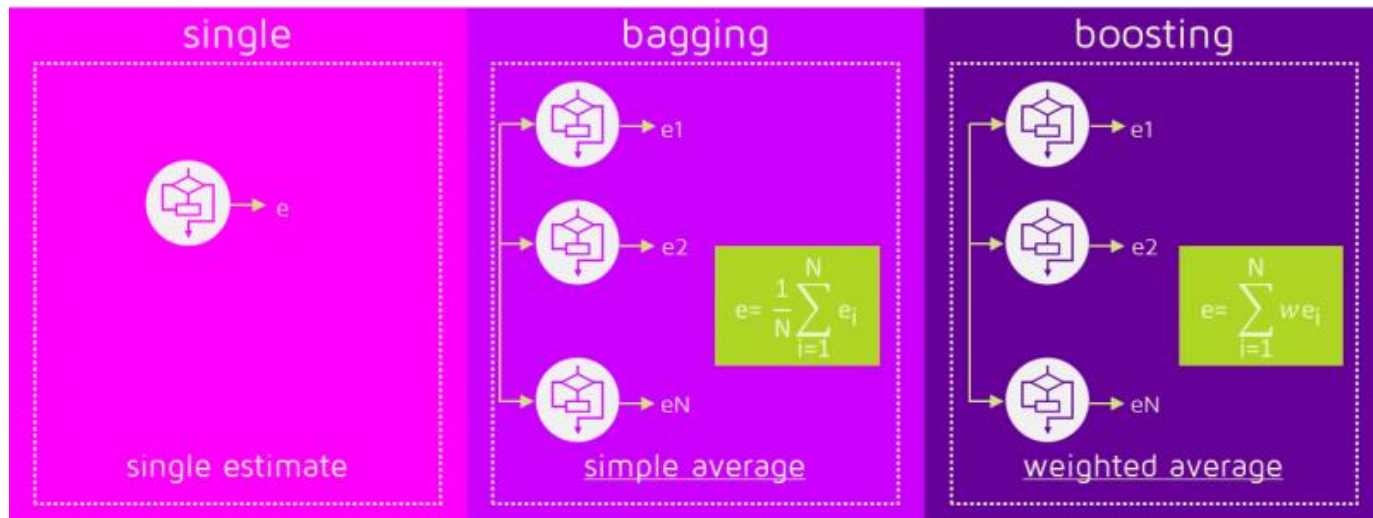
- Bagging vs. Boosting



Classificação

Comitê - Técnicas

- Bagging vs. Boosting



Classificação

Comitê - Técnicas

- Bagging vs. Boosting

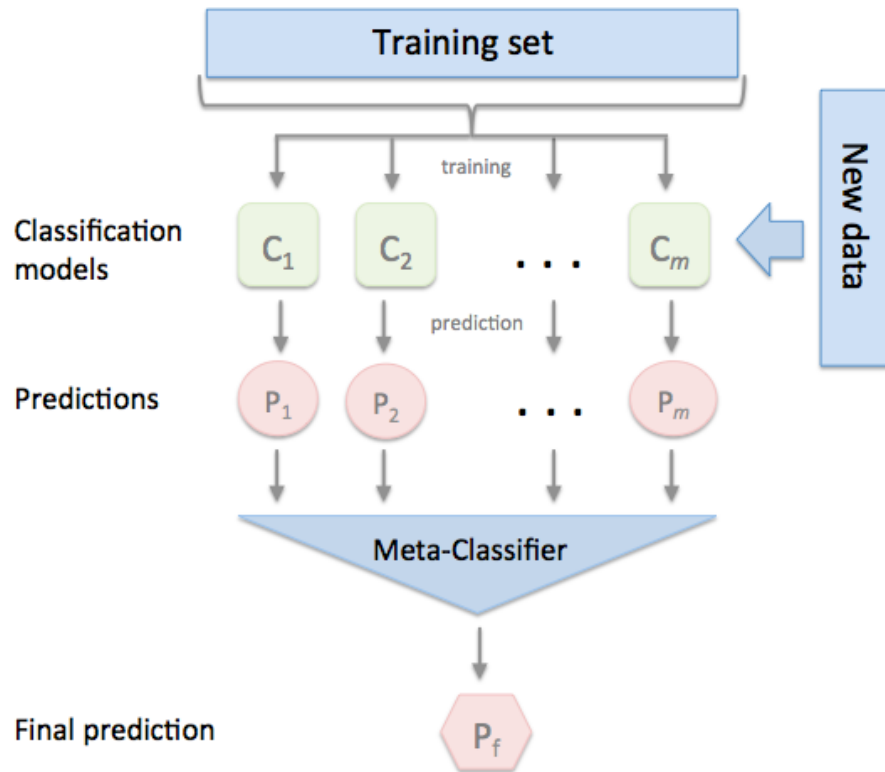


Queridinho do Kaggle: XGBoost

Classificação

Comitê - Técnicas

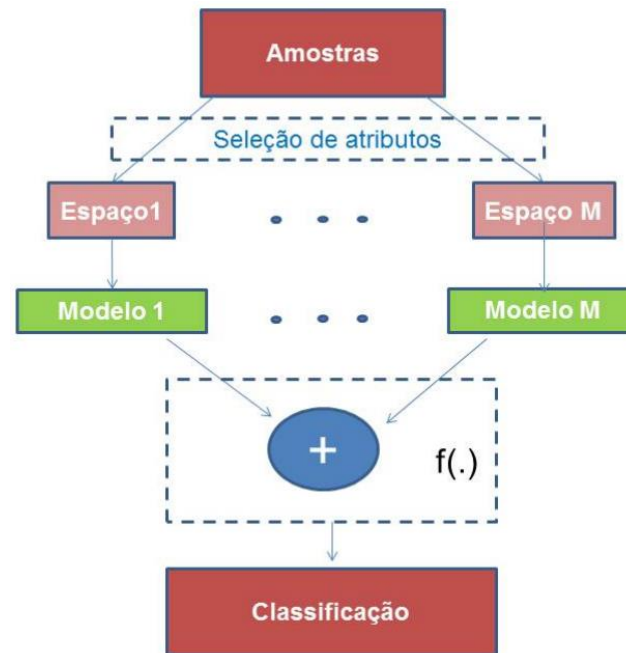
- Stacking



Classificação

Comitê - Técnicas

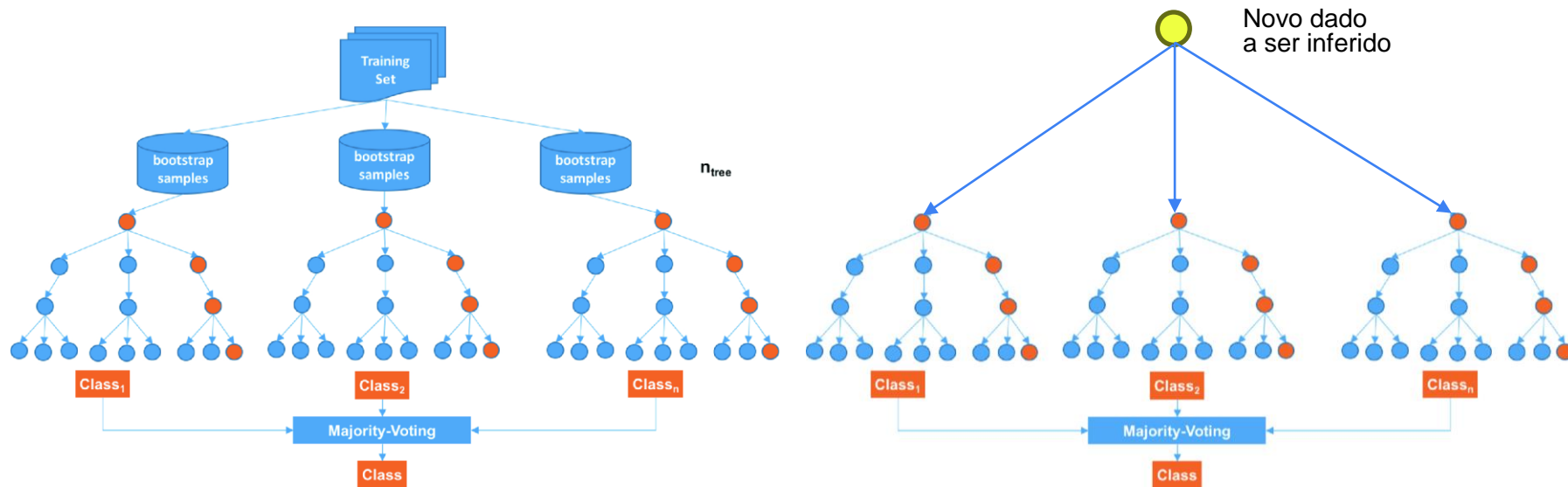
- Random Subspace Method (RSM)
- Similar ao Bagging, mas com aleatorização sobre os atributos.
- Classificadores-base aprendem nos subespaços S de mesma dimensão.
- Decisão final é por votação.



Random Forest

Classificação

Random Forest



A proporção de votos diferentes da classe target em relação ao total de votos é o erro OOB (Out-Of-Bag estimate)

Classificação

Random Forest



Prêmio de 1 milhão de dólares

- Melhora na acurácia do sistema de recomendação de filmes da Netflix em 10%.
- Os melhores times combinaram diversos modelos e algoritmos em um comitê.

Classificação

Random Forest

Tarefa de aprendizado supervisionado

- Dados de treinamento são formados por um conjunto de usuários e as avaliações dos filmes (1,2,3,4,5 estrelas) feitas por esses usuários;
- Construir um classificador que dado um usuário e um filme não avaliado, classifique corretamente aquele filme como 1, 2, 3, 4, ou 5 estrelas;
- Prêmio de \$1 milhão para 10% em melhora na acurácia em relação ao modelo atual.

Estudo de Caso

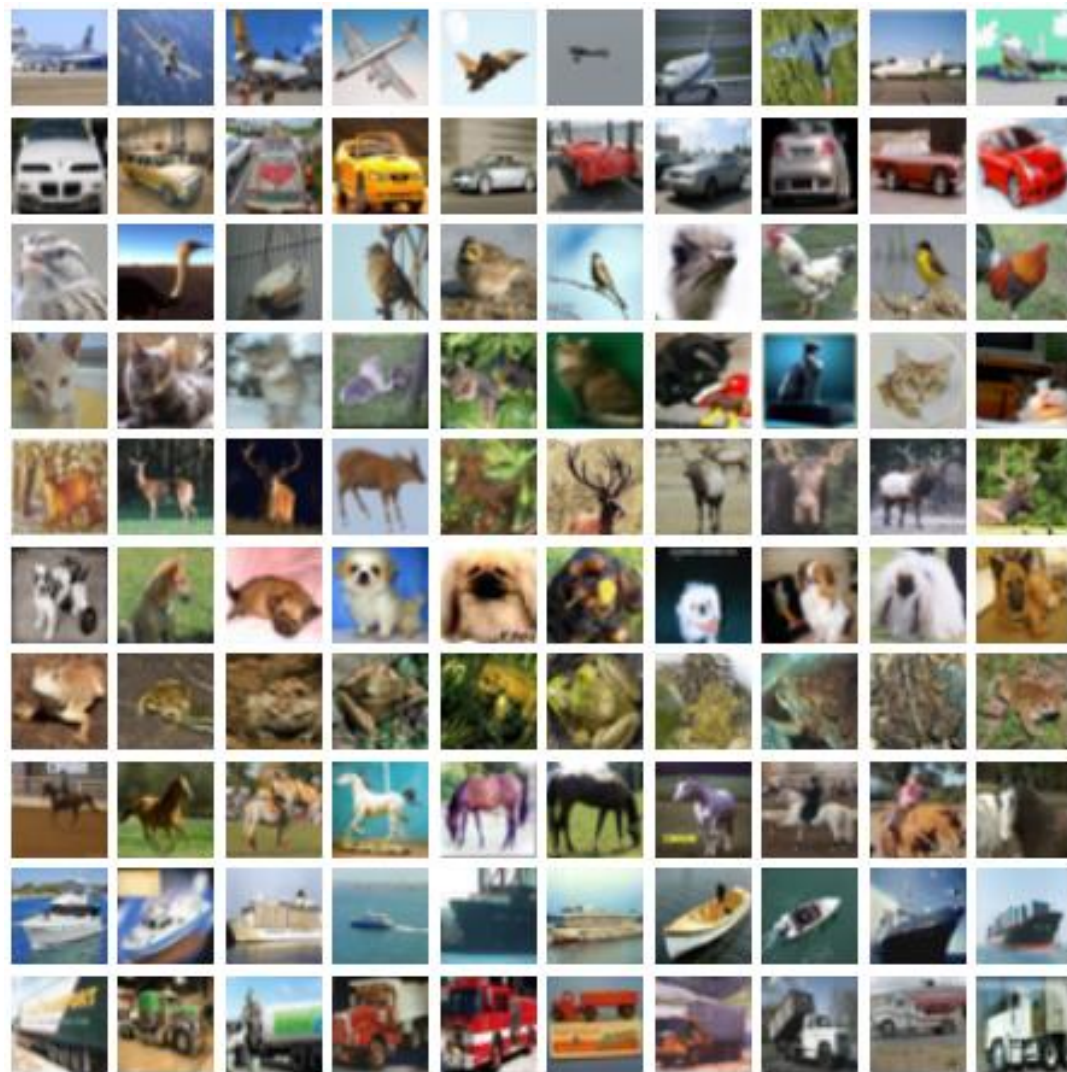
Classificação – Estudo de Caso

KNN

- CIFAR-10
 - 50k imagens de treino
 - 10k imagens de teste
 - 10 classes
 - 32x32x3
 - 8 bits
 - RGB

Curiosidades:

- Geoffrey Hinton foi um dos criadores do dataset. Não só o CIFAR-10, mas também o CIFAR-100 (combinando um total de 80M de imagem rotuladas).



Obrigada!

Prof. Manoela Kohler



prof.Manoela@ica.ele.puc-rio.br



www.linkedin.com/in/manoelakohler