# INTRODUÇÃO AO MACHINE LEARNING E PROCESSAMENTO DE LINGUAGEM NATURAL

github.com/joaorafaelm
joaoraf@me.com

## AGENDA

- o que é e como funciona
- fluxo de desenvolvimento
- o que é NLP
- problemas e aplicações
- exemplo prático

1

# DEFINIÇÃO

*Uma área da Ciência da Computação que dá aos computadores a habilidade de "aprender" sem ter sido programado explicitamente.*

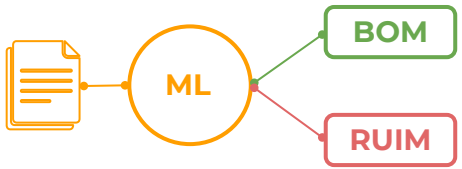
# MÉTODOS

- Supervisionado
- Não Supervisionado
- Reforço

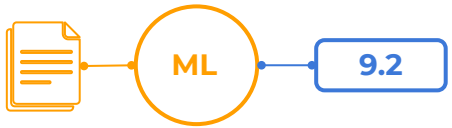
# SUPERVISIONADO

Onde o sistema tenta *aprender* a partir dos exemplos fornecidos.

## CLASSIFICAÇÃO




## REGRESSÃO



# NÃO SUPERVISIONADO

Onde o sistema tenta detectar *padrões* nos dados de entrada.

## ASSOCIAÇÕES

*Exemplo: Pessoas que compraram **X**, tendem a comprar o produto **Y**.*

## CLUSTERIZAÇÃO

*Exemplo: Detecção de fraude.*

# REFORÇO

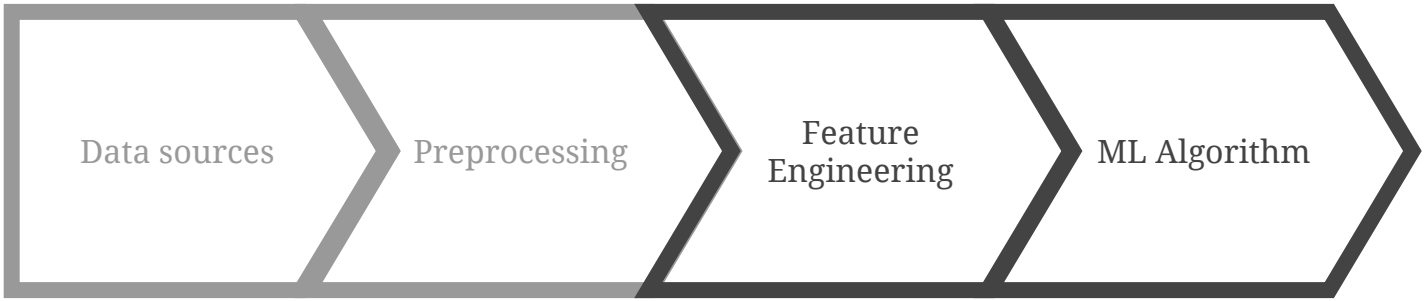
Onde o sistema tem um objetivo específico, dentro de um ambiente dinâmico.

*Exemplos: melhorar desempenho em jogos (poker, tetris etc); carros autônomos etc.*

2

# FLUXO

# FLUXO


Data sources
Preprocessing
Feature Engineering
ML Algorithm

# DADOS

A quantidade e qualidade dos dados determina diretamente o quão bom será o modelo preditivo.

Nem todos os dados são relevantes para o seu modelo preditivo.

| | Cor | Álcool (%) | Ano de criação |
|---|---|---|---|
| Cerveja | **laranja** | **5** | **1870** |
| Vinho | **vermelho** | **13** | **1970** |

Decision Trees

Logistic Regression

Naive Bayes

Multilayer Perceptron

SVM

Nearest Neighbors

?

3

# NLP

# NLP

O objetivo é fazer com que o computador *entenda* linguagem natural.

# APLICAÇÕES

Sumarização de texto

Chat bot

Gerar keywords ou tags

Extração de entidades (data, pessoa etc)

# FEATURES

## DOCUMENTO

*Metadados: título, autor etc.*

*Parágrafos*

*Sentenças*

## PALAVRA

*Vocabulário*

*Forma*

*Frequência*

# VECTORIZAÇÃO

A representação básica é um vetor contendo todas as palavras do vocabulário dos dados

# VECTORIZAÇÃO

*"Eu gosto de quentão"*

*"Eu não gosto de quentão"*


- *As posições do vetor normalmente se baseiam na ordem léxica*

# BAG OF WORDS

*Uma das formas mais simples de computar a frequência das palavras.*

*"Eu gosto de quentão"*

# PROBLEMAS

*O vocabulário pode ser muito grande, o que resultará em um vetor com muitas colunas.*

*Dependendo da representação escolhida, coisas importantes como contexto e ordem podem se perder.*

# SOLUÇÕES

*Remover palavras que podem não ser relevantes para resolver o problema, como "a", "o", "para" etc.*

*Reduzir o número de palavras, normalizando elas em uma forma comum.*

# NLTK

***Stemming**: Remove representações morfológicas da palavra.*

```
>>> from nltk.stem.snowball import SnowballStemmer
>>> stemmer = SnowballStemmer('english')
>>> print(stemmer.stem('running'))
run
```

# NLTK

***Lemmatization**: Retorna a palavra em seu estado canônico.*

```
>>> from nltk.stem import WordNetLemmatizer
>>> lemmer = WordNetLemmatizer()
>>> print(lemmer.lemmatize('going', pos='v'))
go
>>> print(lemmer.lemmatize('went', pos='v'))
go
```

# NLTK

***Stop words**: lista de palavras comuns que podem não ter uma relevância.*

```
>>> from nltk.corpus import stopwords
>>> print(stopwords.words('english'))
{'ourselves', 'hers', 'between', 'yourself', 'but', 'again', 'there', 'about' ...}
```

# EXEMPLO PRÁTICO

# SPAM

Fazer um modelo para classificar um texto em spam ou ham.

# importando dados

```
import json

# Load data
training = json.load(open('training.json', encoding='utf-8'))
test = json.load(open('test.json', encoding='utf-8'))
```

## Definindo o método de *tokenização*

```
from sklearn.feature_extraction.stop_words import ENGLISH_STOP_WORDS
from nltk import wordpunct_tokenize


# POS tag sentences and lemmatize each word
def tokenizer(text):
   for token in wordpunct_tokenize(text):
       if token not in ENGLISH_STOP_WORDS:
           yield token
```

# SPAM

## Definindo o fluxo de processamento

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.stop_words import ENGLISH_STOP_WORDS
from sklearn.pipeline import Pipeline
from sklearn.linear_model import SGDClassifier


# Pipeline definition
pipeline = Pipeline([
   ('vectorizer', TfidfVectorizer(
       tokenizer=tokenizer,
       ngram_range=(1, 2)
   )),
   ('classifier', SGDClassifier()),
])
```

# SPAM

*Treinando* o classificador

```
pipeline.fit(training.get('data'), training.get('label'))
```

# SPAM

*Prevendo...*

```
print(pipeline.predict('você acabou de ganhar 1 milhão de
reais !!!!1111onze!!'))

SPAM
```

OBRIGADO
:)

# Perguntas?

github.com/joaorafaelm

joaoraf@me.com