

SCC0270 - Neural Networks and Deep Learning

Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo

Professor: Tiago Santana de Nazare
E-mail: tiagosn@alumni.usp.br

1º semestre de 2023



Me apresentando...

Tiago Santana de Nazare (tiagosn@alumni.usp.br)

Graduação:

Ciências de Computação ICMC-USP (2010-2015)

Doutorado:

Ciências de Computação ICMC-USP (2015-2020)

Profissional:

Cientista de Dados (2017 - ...)

Professor (2018 - ...)





Avaliação

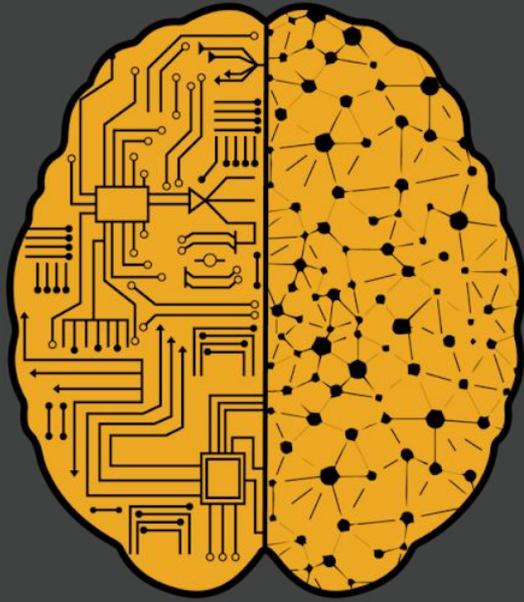
Termos 3 trabalhos durante a disciplina

- Pelo menos 2 semanas para a entrega
- Questões para serem respondidas
- Código em Python
- Presença: entregar pelo menos 2 trabalhos

Nota final (NF):

$$NF = \frac{T_1 + T_2 + T_3}{3}$$

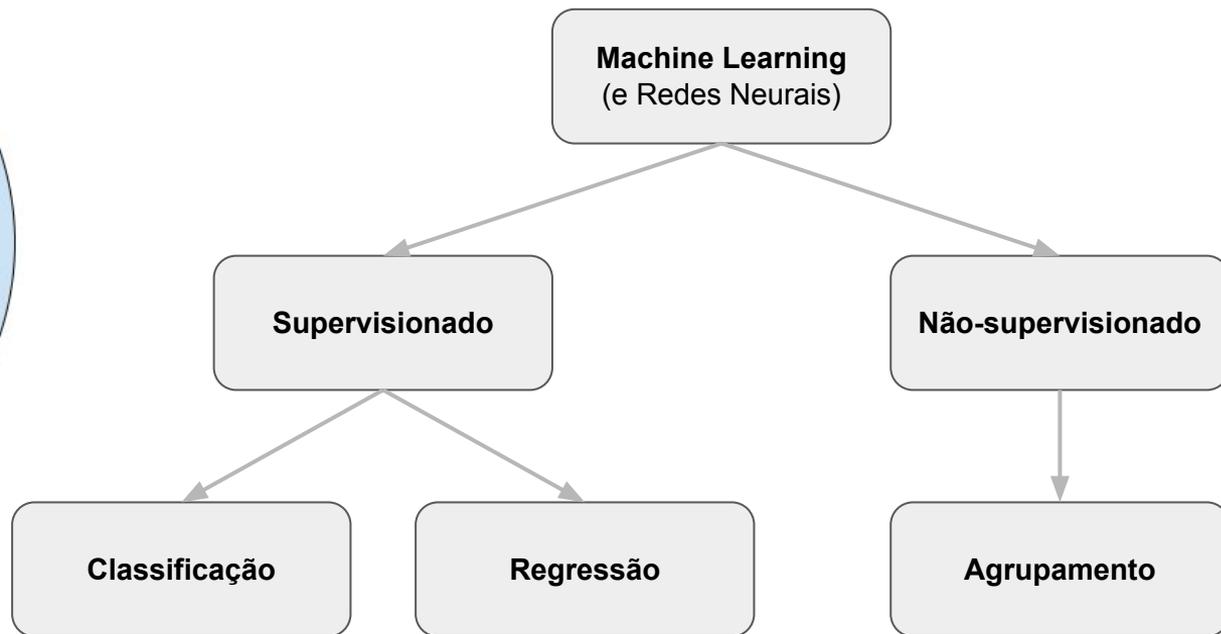
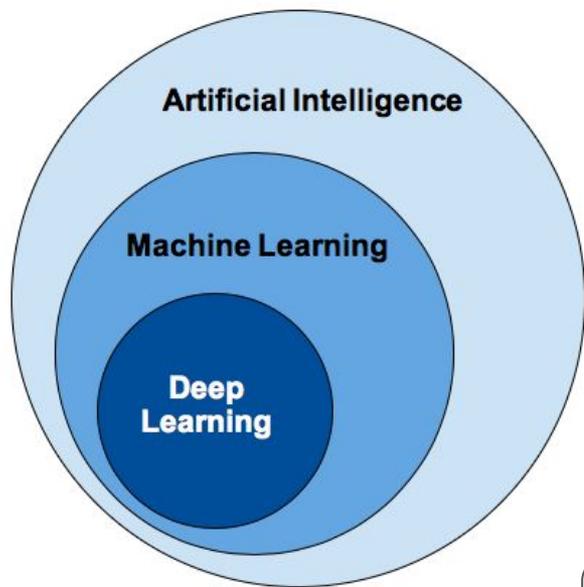




Deep Learning

Mas primeiro...

Contexto



Aprendizado Supervisionado

- Objetivo: criar um modelo para prever uma variável futura (variável resposta) usando algumas variáveis conhecidas (variáveis explicativas)
- Requerimentos:
 - dados para treinar o modelo e avaliá-lo → variável resposta já obtida
 - uma métrica de sucesso para avaliar o resultado do modelo criado
- Tipos de problemas que resolvemos:
 - Classificação: quando a variável resposta é categórica (discreta)
 - Regressão: quando a variável resposta é contínua



Detectar fraudes em cartões de crédito
(problema de classificação)



Prover o valor de venda de uma casa
(problema de regressão)

Aprendizado Não-supervisionado

- Objetivo: Descobrir subgrupos (exemplos similares) em um conjunto de dados
- Requerimentos:
 - dados de treinamento para treinar o modelo e avaliá-lo
 - **NÃO** precisa de uma variável resposta

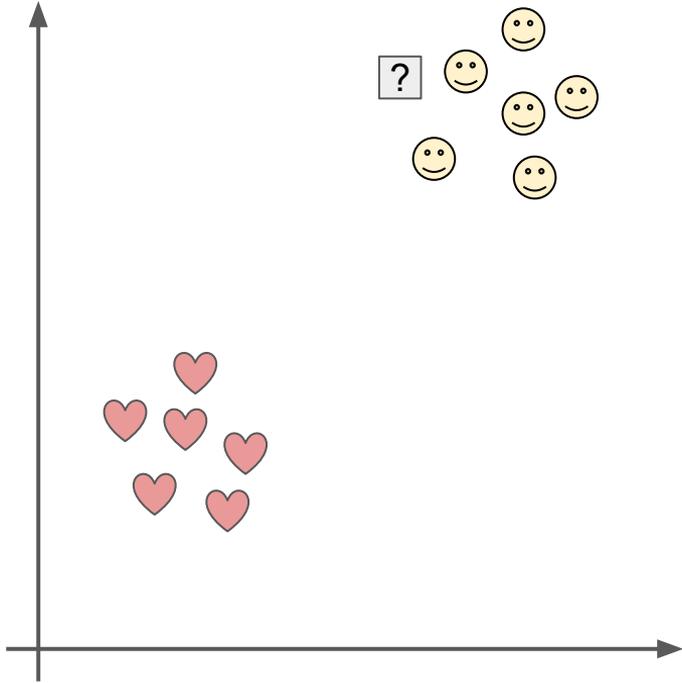


Segmentar grupos de clientes



Sistemas de recomendação de produtos

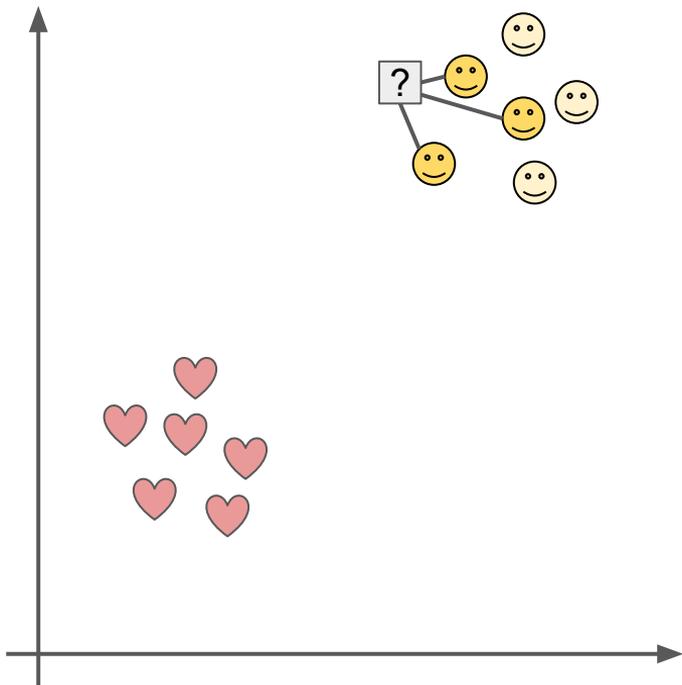
K-Nearest Neighbors (kNN) - Classificação



Intuitivamente falando...

- Qual é a classe do novo elemento?
- Porque?

K-Nearest Neighbors (kNN) - Classificação



Hiperparâmetros:

- $k \rightarrow$ número de vizinhos mais próximos a serem olhados

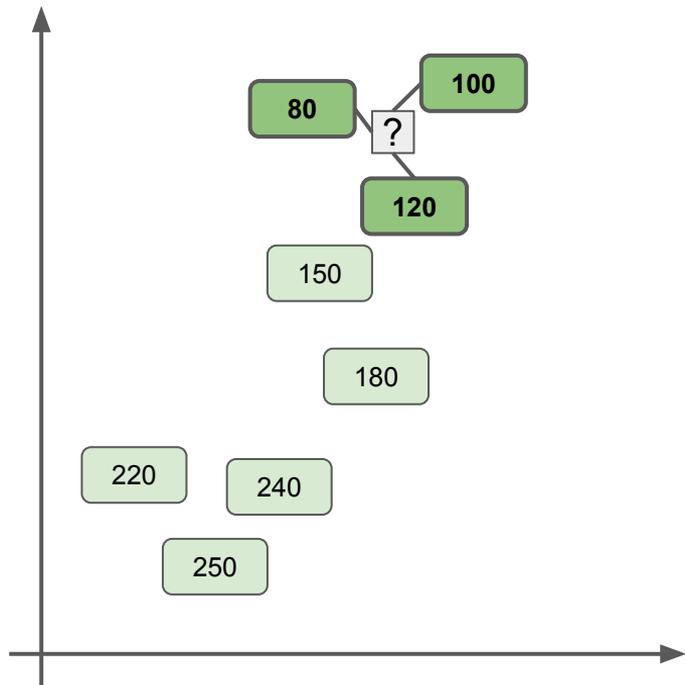
Treinamento:

- Salva a base de treinamento \rightarrow lazy learner

Inferência (previsão para novos dados):

- Encontrar os k vizinhos mais próximos na base de treino
- Atribui a classe da maioria dos vizinhos ao novo exemplo
- Proporção dos vizinhos de cada classe pode ser usada para computar probabilidades

K-Nearest Neighbors (kNN) - Regressão



Hiperparâmetros:

- $k \rightarrow$ número de vizinhos mais próximos a serem olhados

Treinamento:

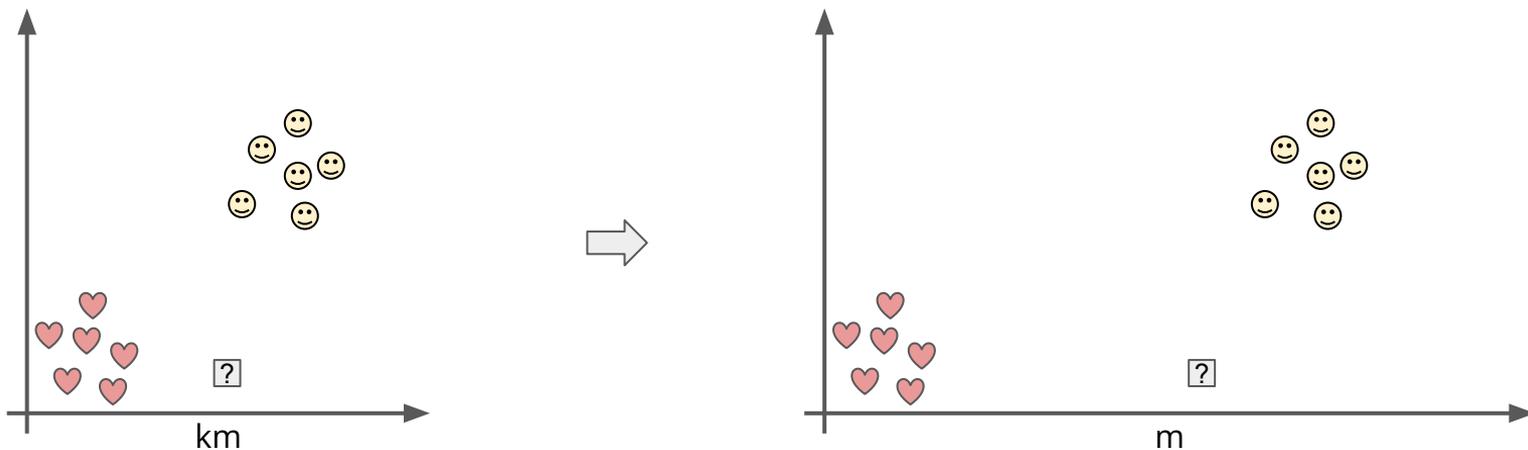
- Salva a base de treinamento \rightarrow lazy learner

Inferência (previsão para novos dados):

- Encontrar os k vizinhos mais próximos na base de treino
- Atribui a média dos valores da variável resposta dos vizinhos ao novo exemplo

Problemas do kNN

1. Custo computacional: comparar com a base toda (índices ajudam)
2. Variáveis com diferentes escalas (ou irrelevantes) podem atrapalhar
 - Transformar todas as variáveis para a mesma escala
 - Remover variáveis irrelevantes



Como medir a qualidade do modelo supervisionado

1. Usar dados não vistos pelo modelos

- Base de treinamento: experimentamos com vários modelos e hiperparâmetros
 - podemos subdividir em treino e validação, para otimizar os hiperparâmetros
- Base de teste: fica reservada até que o melhor modelo seja escolhido

2. Escolher uma métrica apropriada ao problema

- Classificação:
 - Acurácia: % de acertos na base onde o modelo é avaliado
 - AUC: Area Under The Curve → dados desbalanceados
- Regressão:
 - MSE, MAE, ...
 - nem sempre o resultado tem tanto significado prático
 - é comum fazer outros estudos para determinar a qualidade do modelo

$$Acc = \frac{\#acertos}{\#total \text{ de exemplos}}$$

Vamos praticar um pouco...



Mas pra fazer isso... temos que configurar o Python

The logo for Google Colab, featuring the word "colab" in a bold, lowercase, sans-serif font. The letters are colored in a gradient from yellow to orange.

Link: <https://colab.research.google.com>

Vantagens: não precisa instalar nada, na nuvem

Desvantagens: não tem muito recurso computacional



docker

Link: <https://hub.docker.com/r/jupyter/scipy-notebook>

Vantagens: já vem com muitos pacotes instalados

Desvantagens: tem que instalar e configurar o Docker



ANACONDA®

Link: <https://www.anaconda.com> (instalar versão 64 bits)

Vantagens: já vem com muitos pacotes instalados

Desvantagens: tem que instalar e configurar (mas é fácil)



WinPython

Link: <http://tiny.cc/if8wtz>

Vantagens: Python portable com pacotes instalados

Desvantagens: só tem para Windows

Classificação de dados médicos

Notebook para estudo:

- `knn_classification.ipynb`

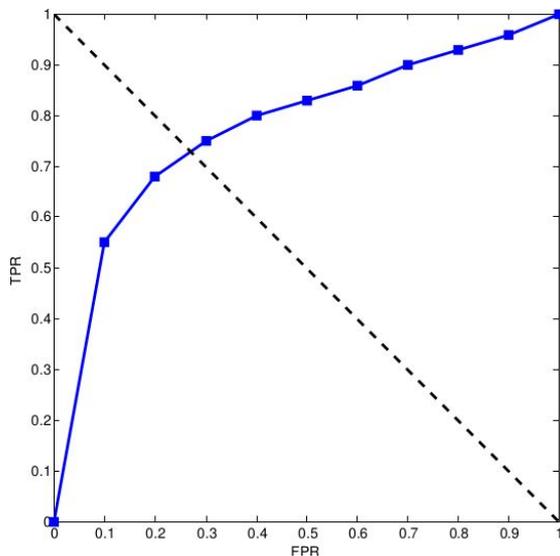


Problemas com dados desbalanceados

Um modelo com a acurácia alta é sempre bom?

- Não, ele pode sempre classificar novos dados com a classe majoritária

AUC pode ajudar nesses casos:



$$\left\{ \begin{array}{l} \text{TPR} = \frac{\text{TP}}{\text{P}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \\ \text{FPR} = \frac{\text{FP}}{\text{N}} = \frac{\text{FP}}{\text{FP} + \text{TN}} \end{array} \right.$$

Previsão do valor de imóveis

Notebook para estudo:

- `knn_regression.ipynb`



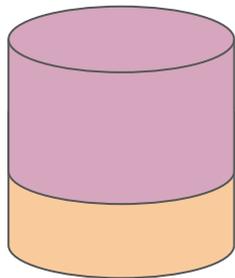
Como escolher os valores dos hiperparâmetros

Testar várias combinações

- Grid-search, Randomized search, ...
- Só usar os dados de treinamento para isso (**teste fica guardado**)

Como usar os dados de treinamento:

Hold-out



Dados de treinamento

Dados de validação (avaliar o modelo)

K-fold cross validation

Fold 1	Fold 2	Fold 3
Fold 1	Fold 2	Fold 3
Fold 1	Fold 2	Fold 3

K-means - Agrupamento

Inicialização:

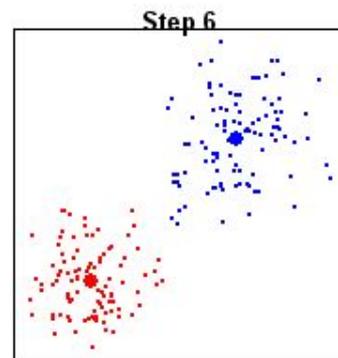
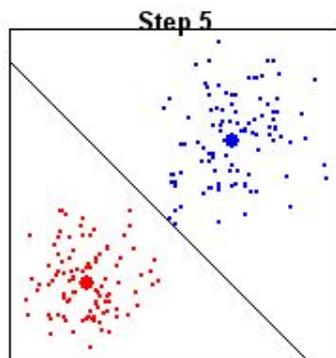
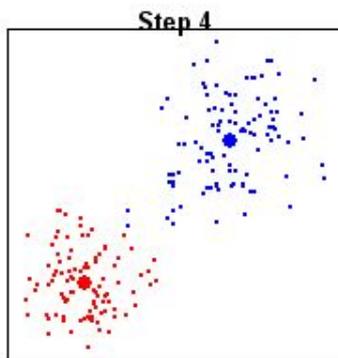
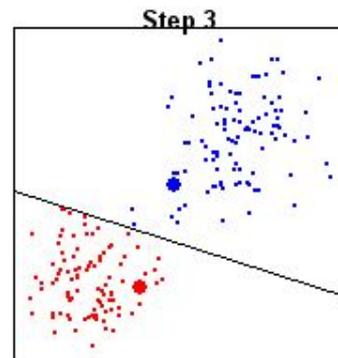
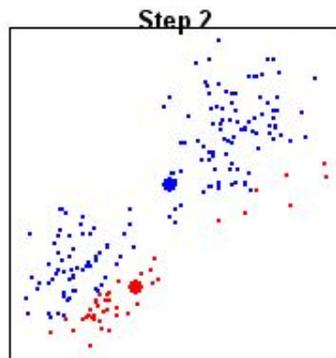
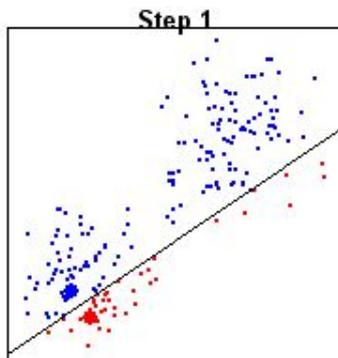
- Criar k centróides aleatórios

Iteração:

- Associar os exemplos ao centróide mais próximo
- Atualizar os centróides \rightarrow média dos elementos associados a ele

Outras versões:

- K-medoids

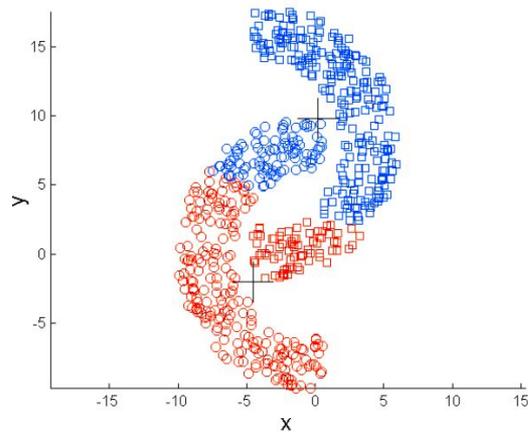
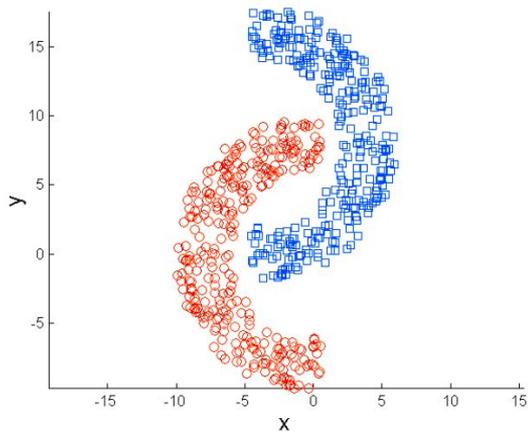


K-means - Limitações

Escolher o valor de k

- Podemos usar a silhueta

Grupos com formatos não hiper esféricos ou com outliers



Quantização de imagens

Notebook para estudo:

- `kmeans_img_quantization.ipynb`



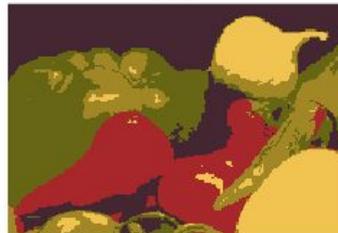
Original



k=2



k=5



k=10





Links interessantes

- <http://neuralnetworksanddeeplearning.com/> ← Livro texto!!!
- <https://scikit-learn.org/>