AGA 0505 - Análise de Dados em Astronomia

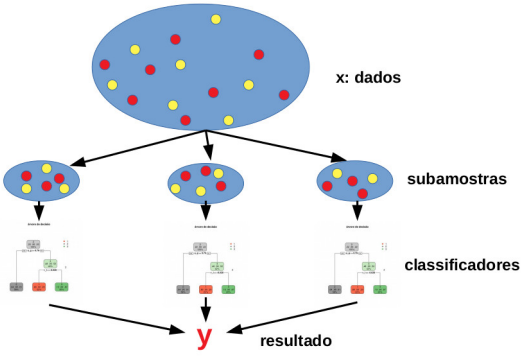
9. Aprendizado de Máquina: Princípios Gerais

Laerte Sodré Jr.

1o. semestre, 2024

aula de hoje

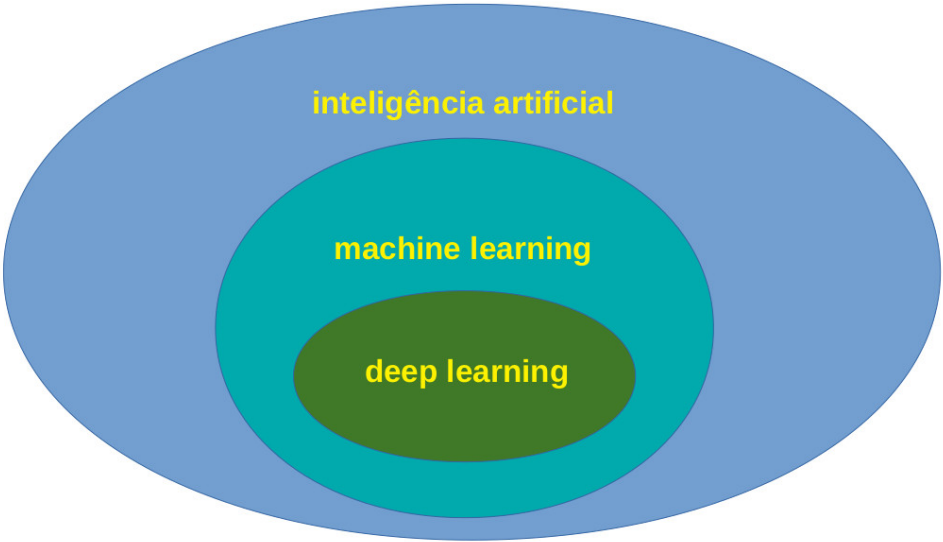
1. inteligência artificial, machine learning e deep learning
2. dados
3. aprendizagem
4. tipos de aprendizagem
5. modelos paramétricos e não-paramétricos
6. generalização e overfitting
7. aplicações de ML: aprendizado não-supervisionado



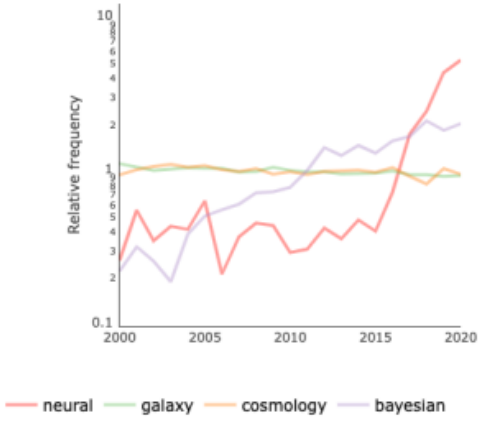
If a machine can think, it might think more intelligently than we do.

Alan Turing

inteligência artificial, machine learning e deep learning

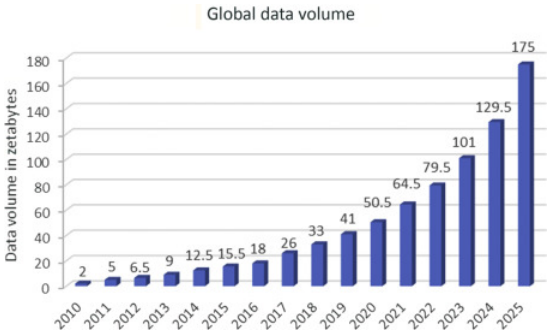


aprendizado de máquina em astronomia



arXiv:2210.01813

ML/DL: abordagem baseada em dados (*data-driven*)
necessária para a era do *tsunami* de dados:

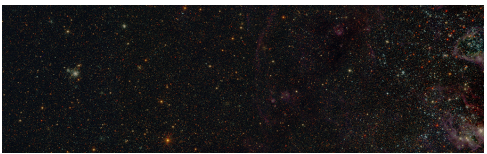


Lei & Kong, 2020

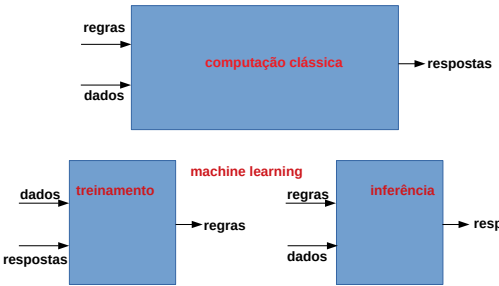
aprendizado de máquina x computação convencional

exemplo: estrela ou galáxia?

- programação clássica
  - sequência de regras específicas ou algoritmos que um computador deve seguir para resolver um problema
  - ex.: SExtractor: a partir da análise de imagens, constrói catálogos de objetos e suas propriedades e usa essas propriedades para calcular a *estelaridade* de um objeto



- machine learning
  sistemas que aprendem diretamente dos dados, sem serem explicitamente programados para uma tarefa específica



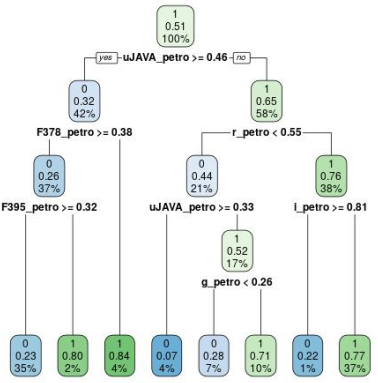
o que é *aprendizado* em ML

- um problema comum em astronomia: estimar uma variável y a partir de um conjunto de outras variáveis, x: $y = f(x; w)$
  x and y podem ser escalares, vetores, tensores ...
  w: parâmetros da função

- um algoritmo de ML promove um mapeamento de x para y: ele implementa uma **função** com parâmetros w

- aprendizado em ML: determinação de w
  ($\sim$ inferência de parâmetros)

- diferentes algoritmos de ML implementam diferentes funções e estratégias de aprendizado



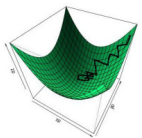
uma árvore de decisão usando fotometria do S-PLUS para classificação estrela/galáxia

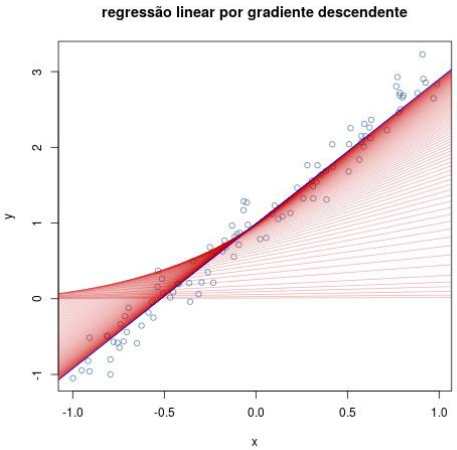
o motor do aprendizado: o algoritmo da *descida do gradiente*

- aprendizado como inferência de parâmetros: minimização de uma *função de custo* $l(w)$ com os dados de um conjunto de treinamento
- $l(w)$ mede os erros no ajuste do modelo com dados do conjunto de treinamento
- algoritmo da descida do gradiente:
  - inicialização: valores "aleatórios" de w
  - atualização iterativa de w:

$$w \longleftarrow w - \lambda \times \nabla l(w)$$

λ: taxa de aprendizado (*hiperparâmetro*)





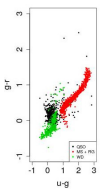
tipos de aprendizado

aprendizado supervisionado

- temos variáveis de entrada e saída, x e y, e o algoritmo *aprende* um mapeamento de x em y usando exemplos de x em um *conjunto de treinamento* para o qual os valores de y são conhecidos
- esses valores de y são denominados alvos ou *targets*
- classificação- y é uma variável qualitativa ou categórica ou discreta:
  'estrela', 'galáxia Scd', 'detectado', 'classe 3', ...
- regressão: y é um número real:
  redshift, metalicidade, massa ...

aprendizado não-supervisionado

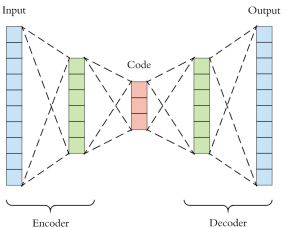
- temos os dados x e o objetivo é identificar estruturas ou propriedades interessantes nos dados
- útil para exploração dos dados:
  - análise de agrupamento (*cluster analysis*)
  - estimativa de densidade
  - redução de dimensionalidade



tipos de aprendizado

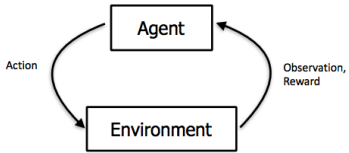
aprendizado auto-supervisionado

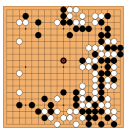
- tipo de aprendizado supervisionado “sem” intervenção humana (∼aprendizado não-supervisionado)
- autoencoders: o algoritmo aprende o *input*



aprendizado com reforço

- um agente aprende ações que maximizam o sucesso ou a recompensa: veículos autônomos, jogos, busca por *outliers*

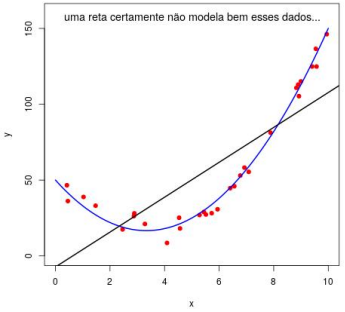




modelos paramétricos e não-paramétricos

- **modelos paramétricos: aprendizado supervisionado**
  - modelos com uma forma funcional conhecida, com um número finito de parâmetros w
  - exemplo: regressão linear $y = w_1 + w_2 x$
  - a predição de y para uma nova observação x depende apenas de w, não dos dados:
    - os parâmetros são estimados dos dados: $P(w|D)$
    - a predição de y depende só de w: $y = f(x; w)$
    - nesse caso ML implementa uma função

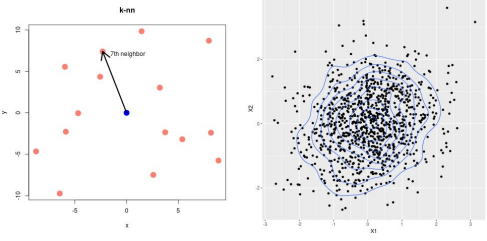
- a complexidade do modelo é limitada pela forma funcional considerada : modelos de *baixa flexibilidade*



modelos paramétricos e não-paramétricos

- **modelos não-paramétricos: aprendizado não-supervisionado**
  - modelos “sem suposições” sobre os dados
  - não temos y!
  - o objetivo é descobrir propriedades interessantes nos dados
  - úteis para exploração e visualização dos dados
  - os modelos não assumem nenhuma forma funcional: dependem apenas dos dados exemplo: estimativa de densidades com o k-ésimo vizinho mais próximo (k-NN)
  - *modelos flexíveis*: podem ser aplicados a dados de qualquer complexidade

- atenção: *modelos não-paramétricos têm hiperparâmetros*
- ex.: k no algoritmo k-NN
- k é um *hiperparâmetro* : parâmetro do algoritmo de aprendizagem

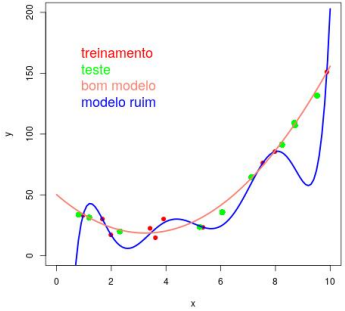


generalização

- modelos são treinados com dados de um conjunto de treinamento
  (“o prior está nos dados”)
- como eles se comportam com novos dados?
- bons modelos devem ser capazes de fazer boas previsões
- bons modelos devem *generalizar* bem!
- em geral os dados são divididos em 3 partes: *conjuntos de treinamento, validação e teste*
- algumas vezes o conjunto de treinamento é usado também para validação

(ex.: 70%, 10%, 20%)

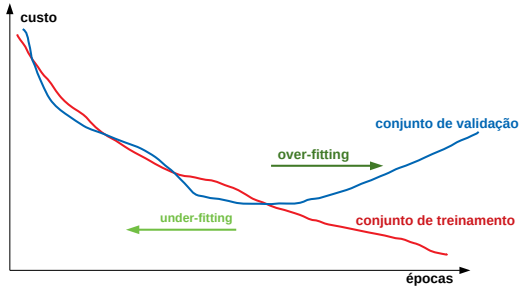
- conjunto de treinamento: usado para ajustar os parâmetros do modelo
- conjunto de validação: monitora o aprendizado do conjunto de treinamento
- conjunto de teste: usado para medir o desempenho do modelo treinado- não deve ser usado nem para treinamento nem para validação



generalização

- conforme o algoritmo vai aprendendo, a função de custo (erros) dos conjuntos de treinamento e validação decresce

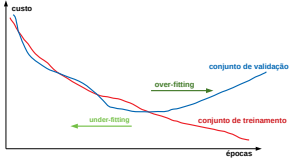
  o modelo está sendo otimizado
- depois de um certo ponto, o custo do conjunto de treinamento continua a cair, mas o do conjunto de validação começa a crescer
- este ponto maximiza a generalização

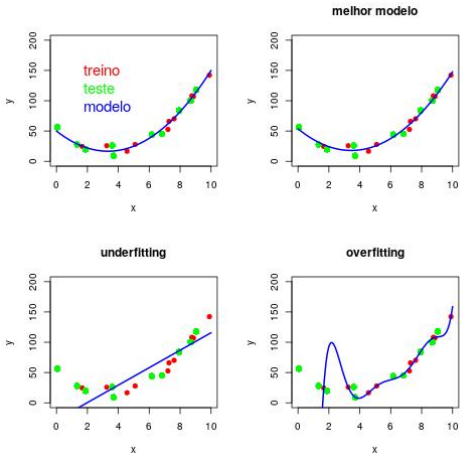


em cada época o algoritmo *processa* todo o conjunto de treinamento

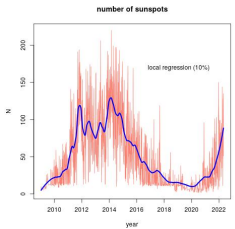
generalização

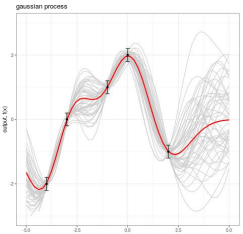
- interromper o aprendizado antes ou depois da época ótima leva a:
  - underfitting: o modelo não foi suficientemente treinado, ou
  - overfitting: o modelo começa a aprender padrões específicos do conjunto de treinamento (como o ruído), que não podem ser generalizados

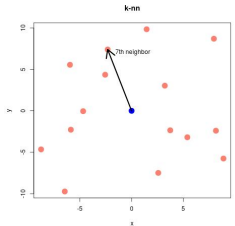


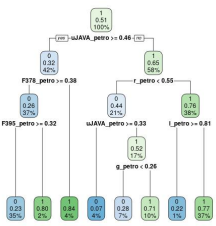


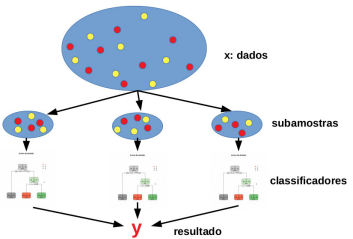
alguns algoritmos de ML

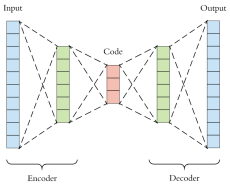


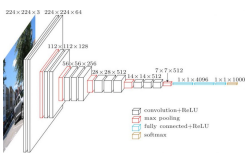












aplicações de ML

aplicações:

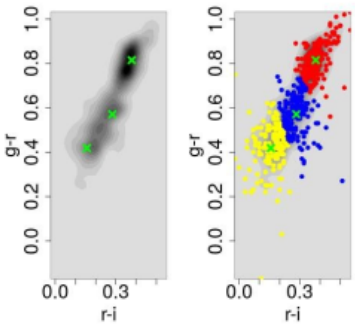
- classificação
- regressão
- estimativa de densidade
- análise de agrupamento
- redução de dimensionalidade
- detecção de anomalias
- séries temporais
- recuperação da informação
- sistemas de recomendação
- robótica e visão computacional
- jogos

a fauna:

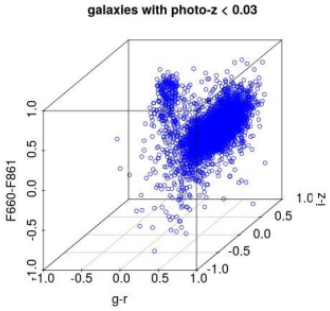
- regressão logística
- árvores de decisão
- *naive bayes*
- k-nn: k-ésimo vizinho mais próximo
- *support vector machines*- SVM
- floresta aleatória & *bagging*
- *boosting*
- redes neurais
- LLM: *large language models*- chatGPT

aplicações: exploração de dados

- estimativa de densidade
- análise de agrupamento
- redução de dimensionalidade



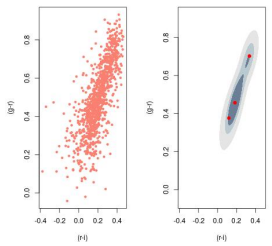
- ferramentas para descobertas!
- úteis para visualização de dados
- úteis para descrição dos dados

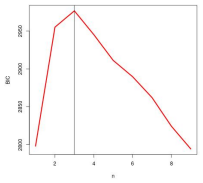


aplicações: estimativa de densidades

- objetivo: modelagem da distribuição de dados no *espaço de dados*
- permite determinar a função de distribuição de probabilidades dos dados
- exemplo de um algoritmo paramétrico: modelo de misturas gaussianas (*gaussian mixture models*): GMM

  modelos de distribuições de densidades com uma superposição de gaussianas multivariadas com diferentes médias, variâncias e orientações
- o número de gaussianas pode ser determinado usando BIC ou AIC





aplicações: estimativa de densidades

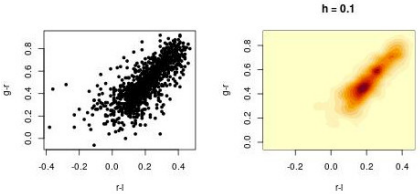
- exemplo de um algoritmo não-paramétrico: estimativa de densidade com kernel (*kernel density estimation*, KDE)
- densidade em um ponto x em um espaço de dados de dimensão D:

$$\hat{f}(x;h) = \frac{1}{Nh^D} \sum_{i=1}^{N} K\left(\frac{d(x,x_i)}{h}\right)$$

onde

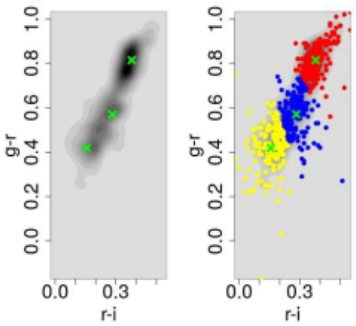
- $K(u)$: função kernel
- $d(x_1, x_2)$: "distância"entre x_1 e x_2
- h: largura de banda do kernel
- N: número de pontos

- h é frequentemente estimado por *validação-cruzada*



aplicações: análise de agrupamento (*cluster analysis*)

- objetivo: identificar grupos de dados (*clusters*) no espaço de dados
- grupos: objetos com propriedades similares
- técnica não-supervisionada: os grupos não são conhecidos a priori
- diferente da classificação, que é um procedimento supervisionado onde os objetos estão associados a classes/grupos pré-definidos
- algoritmos: K-means, métodos hierárquicos



aplicações: análise de agrupamento (*cluster analysis*)

exemplo: algoritmo K-means

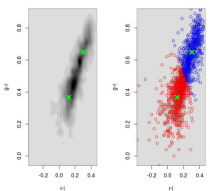
- objetivo: atribuir N objetos a K grupos, $K << N$
  supõe-se K conhecido!

- custo com distância euclidiana: o algoritmo minimiza iterativamente a quantidade

$$J = \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{x}_i - \mu_k)^2$$

onde μ_k é o centroide do k-ésimo grupo

- algoritmo:
  1. defina o número de grupos, K
  2. inicialize escolhendo K objetos ao acaso como centros dos grupos
  3. atribua cada objeto ao grupo mais próximo e recalcule o centróide do grupo
  4. repita até a convergência
     (os grupos “congelam”)

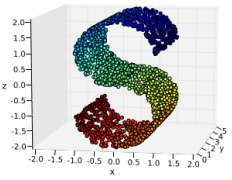


aplicações: redução de dimensionalidade

- X: dados em D dimensões
- queremos uma nova representação de X, que chamaremos Y, em $d << D$ dimensões

$$X = \{x_1, x_2, ..., x_D\} \longrightarrow Y = \{y_1, y_2, ..., y_d\}$$

- útil para compressão de dados e visualização
- método linear: *análise de componentes principais* (PCA)
- métodos não-lineares: LLE (*locally linear embedding*), IsoMap, t-SNE (*t-distributed stochastic neighbor embedding*)
- *embedding*: uma estrutura matemática está contida dentro de outra

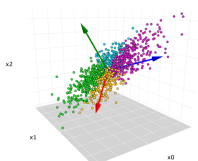


aplicações: redução de dimensionalidade

PCA: análise de componentes principais

- X: dados em um espaço de dados $\mathcal{D}$, d-dimensional
- interpretação geométrica:
  PCA define um conjunto de eixos ortogonais (PCs) em $\mathcal{D}$, tal que
  - PC1 é na direção de máxima variância de $\mathcal{D}$
  - PC2 é na direção de máxima variância do subespaço perpendicular a PC1
  - PC3 é na direção de máxima variância do subespaço perpendicular a PC1 e PC2
  - ...

- PCA é útil para representar os dados com $k << d$ variáveis
- PCA é resolvido como um problema de álgebra linear: é rápido
- cada PC é uma combinação linear de todas as variáveis de X: cuidado com a interpretação física!

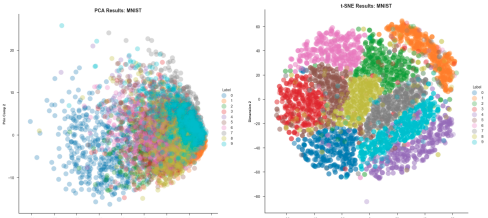


Casey Cheng, https://towardsdatascience.com/principal-component-analysis-pca-explained-visually-with-zero-math-1cbf392b9e7d

aplicações: redução de dimensionalidade

t-SNE: *t-Distributed Stochastic Neighbor Embedding*

- usado pricipalmente para *visualização*
- objetivo: representar dados multi-dimensionais X em um espaço Y de baixa dimensionalidade (2 ou 3 dimensões), preservando *similaridades locais* dos dados
- atenção: os resultados podem mudar cada vez que se roda o algoritmo; t-SNE pode produzir grupos sem significado físico



Andre Violante, https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1

referências

- Statistics, Data Mining, and Machine Learning in Astronomy, Ivezić, Connolly, VanderPlas & Gray, 2014 (https://www.astroml.org/)
- An Introduction to Statistical Learning, James, Witten, Hastie & Tibishirani, 2021 (https://www.statlearning.com/)
- Deep Learning, Goodfellow, Bengio & Courville, 2016 (https://www.deeplearningbook.org/)
- Deep Learning with R, 2a. edição, Chollet, 2022 (https://www.manning.com/books/deep-learning-with-r-second-edition)
- Deep Learning with Python, Chollet, 2018 (https://www.manning.com/books/deep-learning-with-python)
- Modern Statistical Methods for Astronomy: With R Applications , Feigelson & Babu, 2012
- The Dawes Review 10: The impact of deep learning for the analysis of galaxy surveys, M. Huertas-Company & F. Lanusse, arXiv:2210.01813, 2022