

# ***DATA SCIENCE***

---

## **AULA 4 - Aprendizagem Supervisionada**

**Prof. Gabriel Resende Machado**



[gabrielmachado@unifeso.edu.com](mailto:gabrielmachado@unifeso.edu.com)



<https://www.linkedin.com/in/machadogabriel>



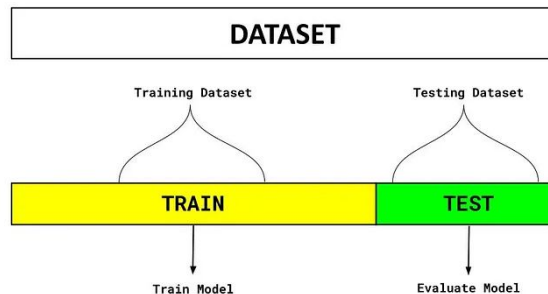
<https://github.com/UNIFESO-Gabriel/data-science>

# Técnicas de amostragem dos dados

- Métodos de amostragem resultam em estimativas de desempenho mais confiáveis dos modelos de aprendizado a partir de conjuntos disjuntos de treinamento e teste;
  - os dados de treino servem para ajustar os parâmetros do modelo buscando generalização;
  - os dados de teste simulam a apresentação de novas amostras ao modelo;
  - os principais métodos de amostragem dos dados são o (i) *holdout*, (ii) validação cruzada e (iii) *bootstrap*.

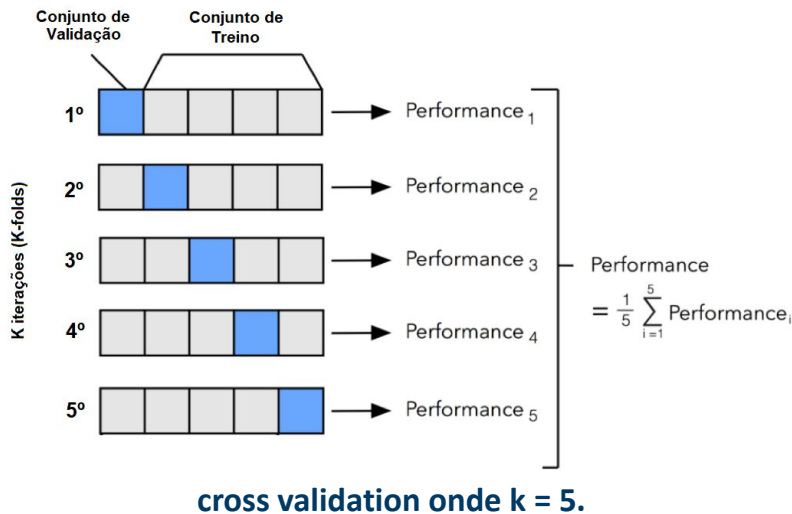
# Técnicas de amostragem dos dados: *holdout*

- No caso do *holdout*, o conjunto de dados é dividido em uma proporção  $p$  para treinamento e  $1-p$  para teste;
  - Normalmente, emprega-se  $p = \{2/3, 0,7 \text{ ou } 0,8\}$ . Em tarefas de Deep Learning, com conjuntos de dados muito grandes, emprega-se  $p \geq 0,95$ ;
  - É comum haver também uma variação do *holdout* que realiza a separação dos dados em treino, validação e teste.
    - O conjunto de validação serve, principalmente, para avaliar o desempenho do modelo sem gerar vieses com o conjunto de teste.
- O *holdout* não permite avaliar o desempenho do modelo perante diferentes combinações, como é feito pela validação cruzada.



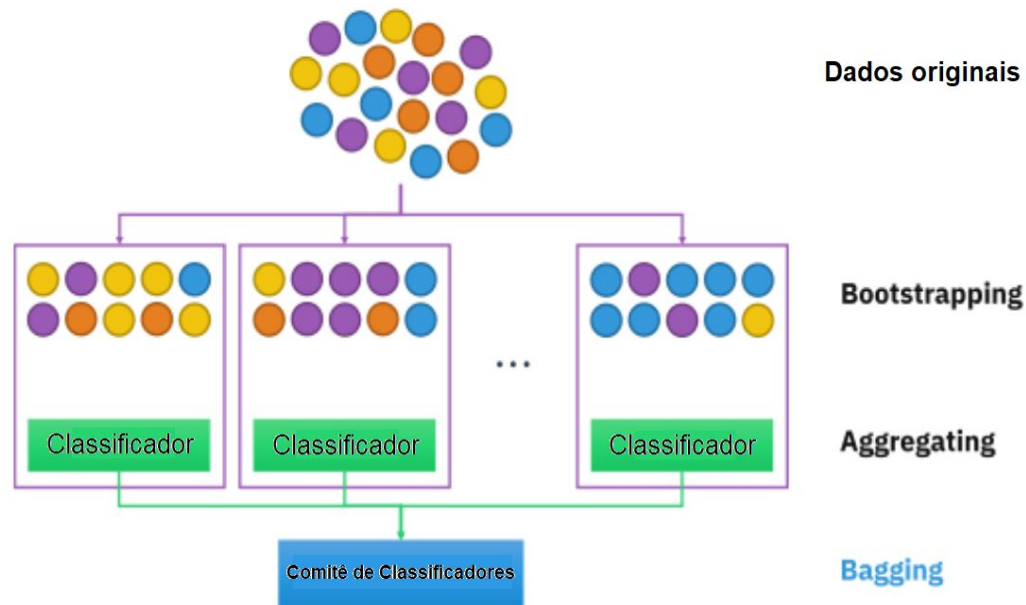
# Técnicas de amostragem dos dados: *cross validation*

- Na validação cruzada, o conjunto de exemplos é dividido em  $k$  subconjuntos do mesmo tamanho.
- Em um subconjunto  $k$ ,  $n-1$  partições são utilizadas para treino de um modelo específico, que é avaliado na partição restante;
- O processo é repetido  $k$  vezes e o desempenho final do modelo é dado pela média dos desempenhos observados em cada subconjunto.
  - É importante ressaltar que é importante exibir o desvio padrão juntamente com a média.
- Uma variação dessa abordagem é aplicar o holdout e, após, o cross validation no conjunto de treino.



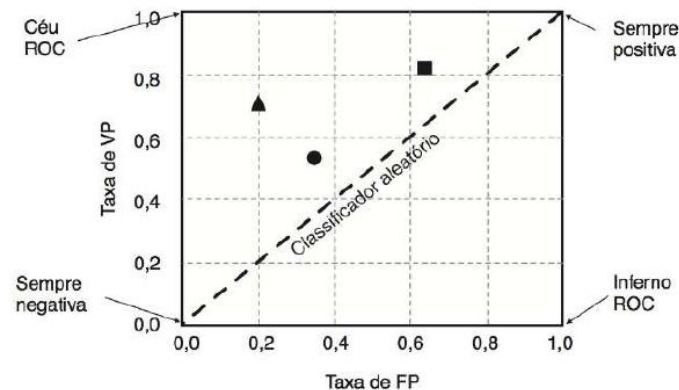
# Técnicas de amostragem dos dados: *bootstrap*

- *Bootstrap* é uma técnica de reamostragem que envolve criar vários *datasets* escolhendo registros aleatoriamente com reposição;
- Com isso, um mesmo exemplo pode estar presente em um determinado subconjunto mais de uma vez. Os exemplos não selecionados compõem o conjunto de teste;
- A abordagem de treinar vários classificadores em conjuntos de dados gerados por *bootstrap* se chama **bagging**. A classificação final do comitê pode ser feita via média ou voto majoritário.

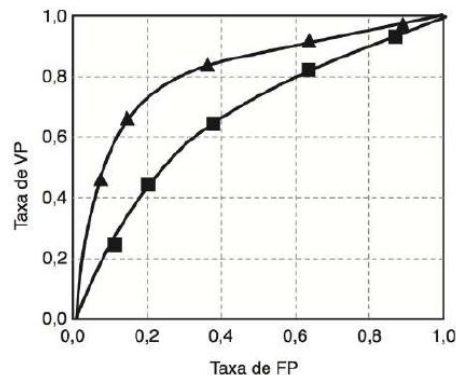


# A Curva ROC e AUC

- O gráfico ROC é uma forma alternativa de se avaliar classificadores binários, onde os eixos X e Y representam as taxas de falsos positivos e verdadeiros positivos, respectivamente;
- Um gráfico ROC possui uma **linha diagonal** que representa o desempenho aleatório de um classificador. Qualquer classificador abaixo dessa linha possui um desempenho pior do que um classificador aleatório.
- O **AUC** é o valor da **área abaixo da curva ROC** que varia de 0 (pior) a 1 (melhor), plotada a partir da escolha de diferentes limiares para um determinado classificador.



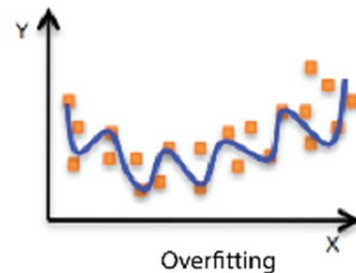
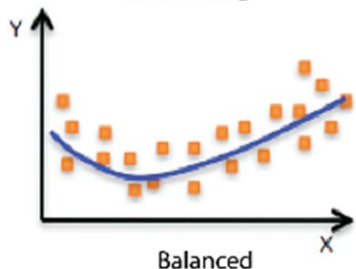
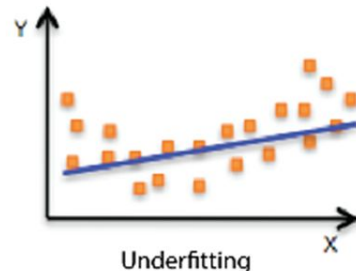
▲ Algoritmo A ■ Algoritmo B ● Algoritmo C



▲ Algoritmo A ■ Algoritmo B

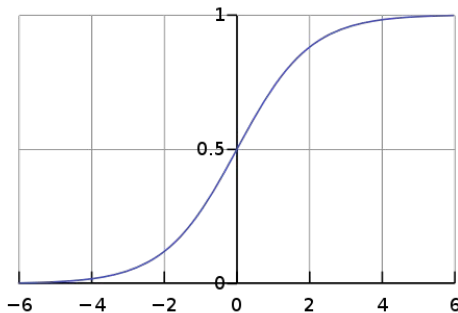
# Trade-off entre Viés e Variância

- **Baixo Viés e Baixa Variância:** É um modelo que apresenta um bom desempenho tanto nos dados de treinamento quanto nas previsões;
- **Baixo Viés e Alta Variância:** o modelo está superajustado (*overfitting*) nos dados de treino e não generaliza bem em novos dados.
- **Alto Viés e Baixa Variância:** O modelo está subajustado (*underfitting*) nos dados de treinamento e não captura a relação entre as variáveis preditoras e a variável resposta.
- **Alto Viés e Alta Variância:** O modelo está inconsistente e com um acurácia muito baixa nas previsões.
- O *trade-off* ocorre porque se a complexidade do modelo aumentar, o viés é reduzido, mas aumenta-se a variância. Da mesma forma, se a complexidade do modelo diminuir, a variância é reduzida, mas aumenta-se o viés do modelo.



# Regressão Logística

- Ao contrário da Regressão Linear, a Regressão Logística é voltada para **problemas de classificação**;
- É um algoritmo recomendado para problemas de **classificação binária**;
- Produz como saída **uma probabilidade de uma amostra pertencer à uma determinada classe**;
  - Utiliza a função sigmoide para realizar o cálculo das probabilidades:  $\frac{1}{1 + e^{-(\beta_0 + \vec{\beta}_1 x)}}$
  - $\beta_0$  e  $\vec{\beta}_1$  são atualizados via gradiente descendente e o rótulo é atribuído por meio de um limiar (geralmente 0,5).





# Regressão Logística: Gradiente Descendente

- Objetivo: encontrar os parâmetros  $\beta_0$  e  $\vec{\beta}_1$  tal que minimizem a função de custo J;

$$J(\beta_0, \vec{\beta}_1) = -\frac{1}{m} \sum_{i=1}^m y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})$$

- Aplicar iterativa e simultaneamente o gradiente descendente à função de custo J:

- $\beta_0 = \beta_0 - \alpha \frac{\partial}{\partial \beta_0} J(\beta_0, \vec{\beta}_1)$

- $\beta_{1_j} = \beta_{1_j} - \alpha \frac{\partial}{\partial \beta_{1_j}} J(\beta_0, \vec{\beta}_1)$

- Onde, tem-se:

- $\frac{\partial}{\partial \beta_{1_j}} J(\beta_0, \vec{\beta}_1) = \frac{1}{m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)}) x^{(i)}$

- $\frac{\partial}{\partial \beta_0} J(\beta_0, \vec{\beta}_1) = \frac{1}{m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})$

# Exercício 1











- Realize a predição do estado de um carro a partir de seus atributos. Utilize a Regressão Logística para isso. Exiba ao final a curva ROC do modelo.

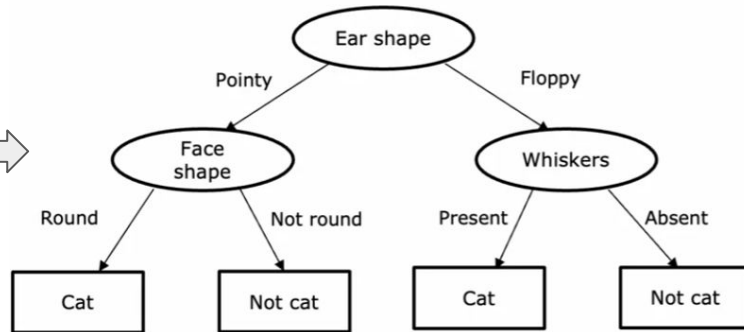
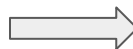


Link do dataset: <https://shorturl.at/joyHW>.

# Árvores de Decisão

- Modelo de aprendizado supervisionado de caixa-branca, formado por uma estrutura hierárquica que consiste em um nó raiz, ramos, nós internos e nós folhas;
- O objetivo é criar um modelo que realiza a predição de um valor para uma variável alvo a partir de regras de decisão simples, geradas a partir dos dados de treino;
- Os algoritmos de árvores de decisão mais conhecidos são o C4.5 e o CART (este implementado no Scikit-Learn).

	Ear shape	Face shape	Whiskers	Cat
	Pointy	Round	Present	1
	Floppy	Not round	Present	1
	Floppy	Round	Absent	0
	Pointy	Not round	Present	0
	Pointy	Round	Present	1
	Pointy	Round	Absent	1
	Floppy	Not round	Absent	0
	Pointy	Round	Absent	1
	Floppy	Round	Absent	0
	Floppy	Round	Absent	0

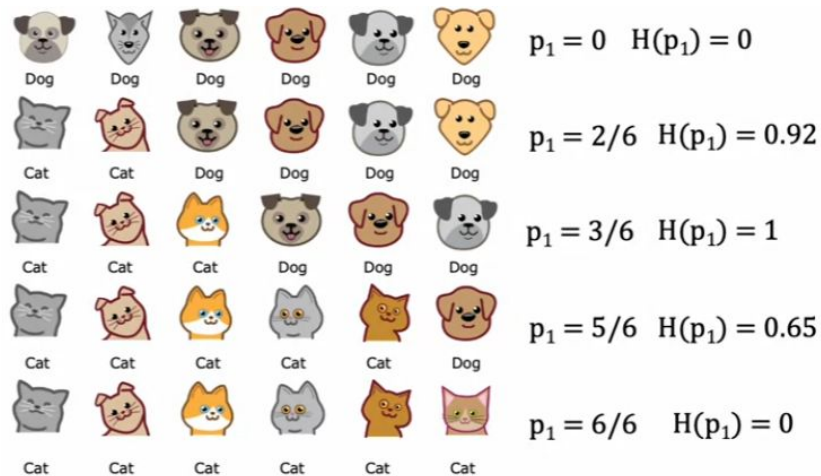
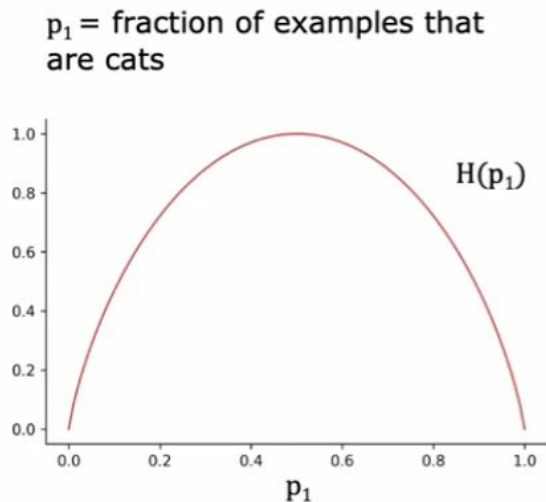


# Árvores de Decisão: Questões

- **Pergunta 1: Qual característica escolher para um determinado nó da árvore de decisão?**
  - Aquela que maximiza a taxa de pureza (diminui a entropia).
- **Pergunta 2: Quando finalizar o processo de repartição da árvore em nós folhas?**
  - Quando um nó é composto por amostras pertencentes a apenas uma classe;
  - Quando a repartição de um nó em nós folhas exceder o limiar de profundidade máxima;
  - Quando a taxa de pureza tiver um ganho menor do que um limiar pré-definido;
  - Quando o número de amostras em um nó estiver abaixo de um limiar pré-definido.

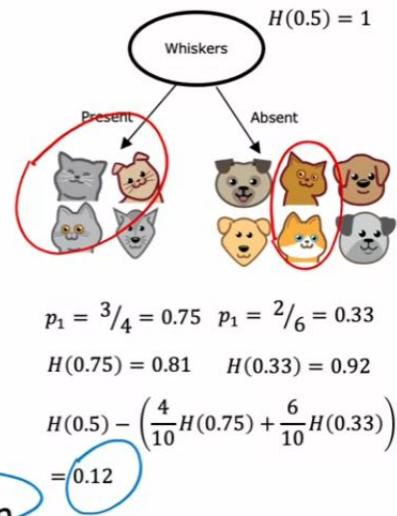
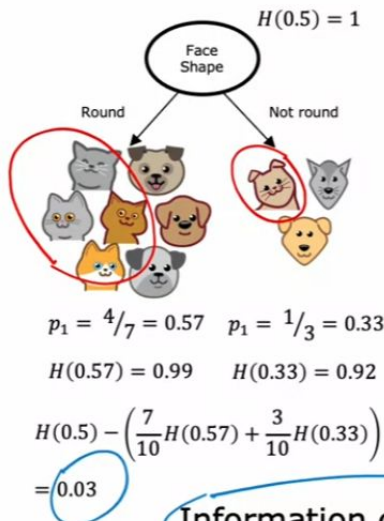
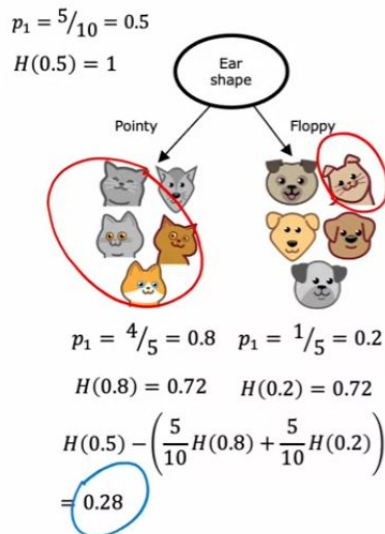
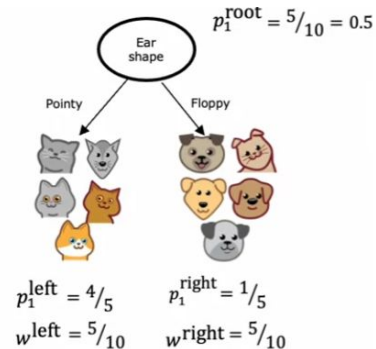
# Árvores de Decisão: Entropia

- Calcula a taxa de impureza em um nó da árvore;
- A entropia é calculada a partir da equação  $H(p_1) = -p_1 \log_2(p_1) - (1 - p_1) \log_2(1 - p_1)$ 
  - Considerar  $0 \log_2(0) = 0$  .



# Árvores de Decisão: Repartição do Nó

- O processo de repartir um nó é definido via *Information Gain* (IG);
- O *Information Gain* mede a redução na entropia após uma repartição;
- A Equação para o cálculo do IG é:  $H(p_1^{root}) - (w^{left} H(p_1^{left}) + w^{right} H(p_1^{right}))$



Information gain

# Árvores de Decisão: Etapas

1. Iniciar com todas as amostras no nó raíz;
2. Calcular o IG para todas os atributos, e selecionar aquele com o maior valor de *Information Gain*;
3. Repartir o dataset de acordo com o atributo selecionado para o nó, e criar os ramos da esquerda e direita da árvore;
4. Continuar o processo de repartição até que algum dos critérios de parada seja satisfeito:
  - Quando um nó estiver homogêneo, *i.e.*, contendo apenas amostras referentes a uma classe;
  - Quando a repartição de um nó resultar em uma profundidade maior que o limiar pré-definido;
  - O IG for menor que um determinado limiar;
  - Quando o número de amostras em um nó estiver abaixo de um limiar.

# Árvores de Decisão: C4.5 x CART

## 1. Critério de repartição:

- C4.5 utiliza o *Information Gain* como critério de repartição; CART utiliza o índice Gini, que mede a probabilidade de um erro de classificação em um registro escolhido aleatoriamente;

## 2. Estrutura da árvore:

- C4.5 produz uma árvore que pode ser desbalanceada e complexa; CART produz árvores binárias;

## 3. Manuseio de valores ausentes:

- C4.5 utiliza várias estratégias para lidar com valores nulos, como inferências estatísticas. CART já necessita de técnicas de pré-processamento dos dados *a priori*, como a imputação;

## 4. Disponibilidade:

- o C4.5 é um algoritmo proprietário e sua implementação original não é disponível publicamente. Por sua vez, o CART está disponível em diversas bibliotecas, como o Scikit-Learn.



# Árvores de Decisão: Vantagens

## 1. Flexibilidade:

- Árvores de decisão são algoritmos não-paramétricos, ou seja, não assumem nenhuma distribuição dos dados;

## 2. Seleção de atributos:

- O processo de construção de uma árvore de decisão seleciona os atributos que são utilizados no modelo de decisão. Essa seleção produz modelos mais robustos a atributos redundantes e irrelevantes;

## 3. Interpretabilidade:

- Decisões complexas e globais podem ser aproximadas por uma série de decisões mais simples e locais. Todas as decisões são baseadas nos valores dos atributos usados para descrever o problema.

## 4. Eficiência:

- Árvores de decisão são algoritmos gulosos construídos de cima para baixo (*top-down*) por meio de uma estratégia de dividir para conquistar. Sua complexidade de tempo é linear com o número de exemplos.

# Árvores de Decisão: Desvantagens

## 1. Replicação:

- Uma sequência de testes em uma árvore de decisão pode ser duplicada em diferentes ramos;

## 2. Valores ausentes:

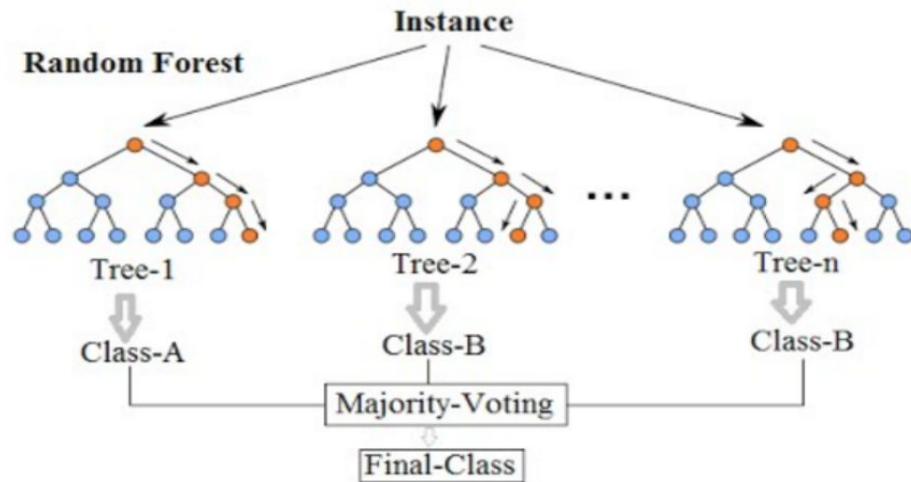
- Uma árvore de decisão é uma hierarquia de testes. Se o valor de um atributo é desconhecido, isso causa problemas em decidir qual ramo seguir;

## 3. Atributos categóricos:

- Os algoritmos disponíveis de árvores de decisão não são adaptados a trabalharem com valores qualitativos, sendo necessário codificá-los para valores quantitativos.

# Florestas Aleatórias

- Modelo baseado na estratégia de comitês de aprendizado;
- É formada por um conjunto de árvores de decisão:
  - Em tarefas de classificação, a classe é predita a partir de voto majoritário;
  - Em tarefas de regressão, é calculada a média dos valores fornecidos por cada árvore.



## Exercício 2 - Entrega até o dia 02/10

- Utilize uma árvore de decisão para prever se a renda salarial anual de uma pessoa excede \$50k. Treine e avalie o modelo em 10 validações cruzadas e exiba a árvore de decisão gerada (utilize a biblioteca *Graphviz*). Por fim, compare os resultados com o desempenho de uma floresta aleatória.



Link do notebook para download: <https://shorturl.at/gmryJ>.

# ***DATA SCIENCE***

---

**AULA 4 - Aprendizagem Supervisionada**

**Dúvidas e/ou perguntas?**



[gabrielmachado@unifeso.edu.com](mailto:gabrielmachado@unifeso.edu.com)



<https://www.linkedin.com/in/machadogabriel>



<https://github.com/UNIFESO-Gabriel/data-science>