Data Science
Bootcamp

WOMAKERS CODE

Host Oficial

sãojudas universidade

MÓDULO #5

Machine Learning – Parte 1

Ensemble

Vivian Yamassaki @ViviYamassaki
Jéssica dos Santos @j3ssicaSant0s

Jéssica dos Santos

Cientista de Dados na NeuralMed

j3ssicaSant0s

jessica-santos-oliveira

Vivian Yamassaki

Cientista de Dados na Creditas

vivianyamassaki

vivianyamassaki

O que veremos na aula de hoje?

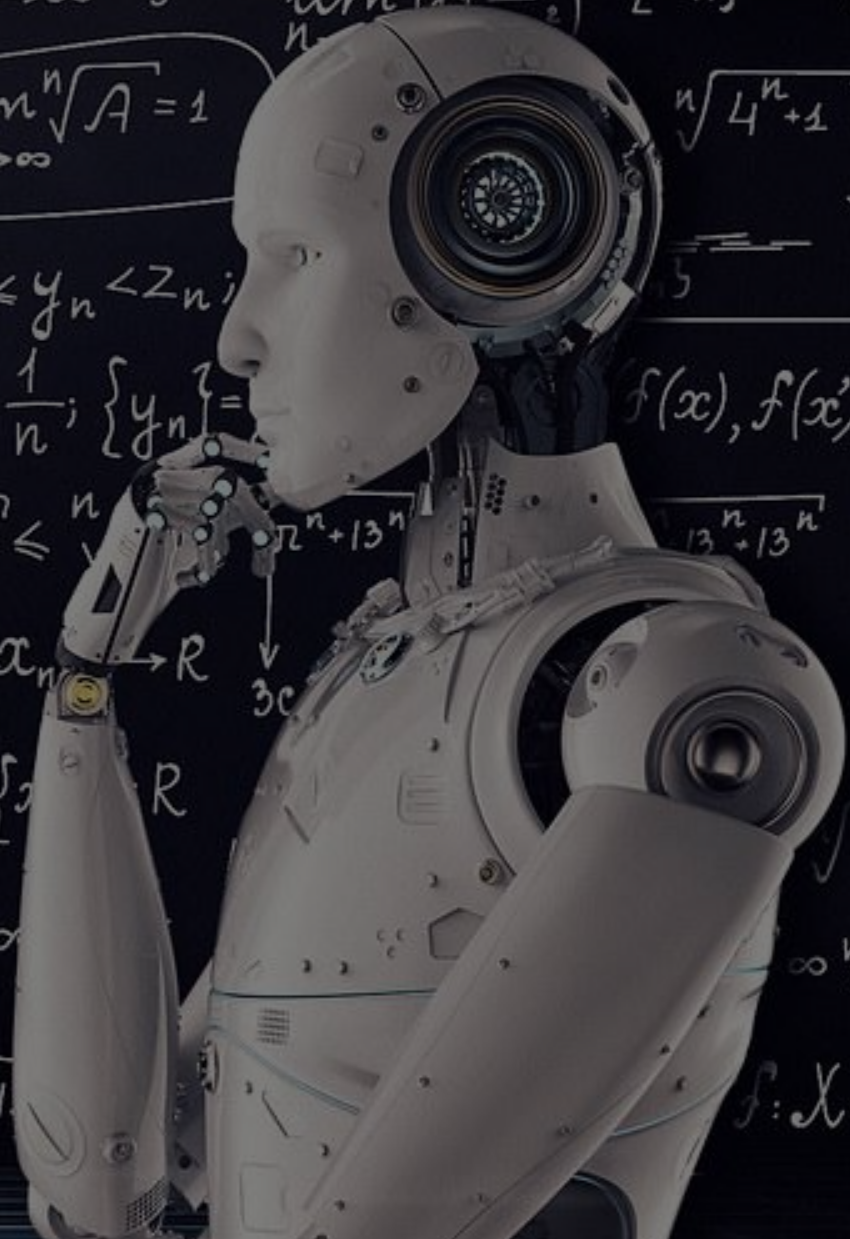
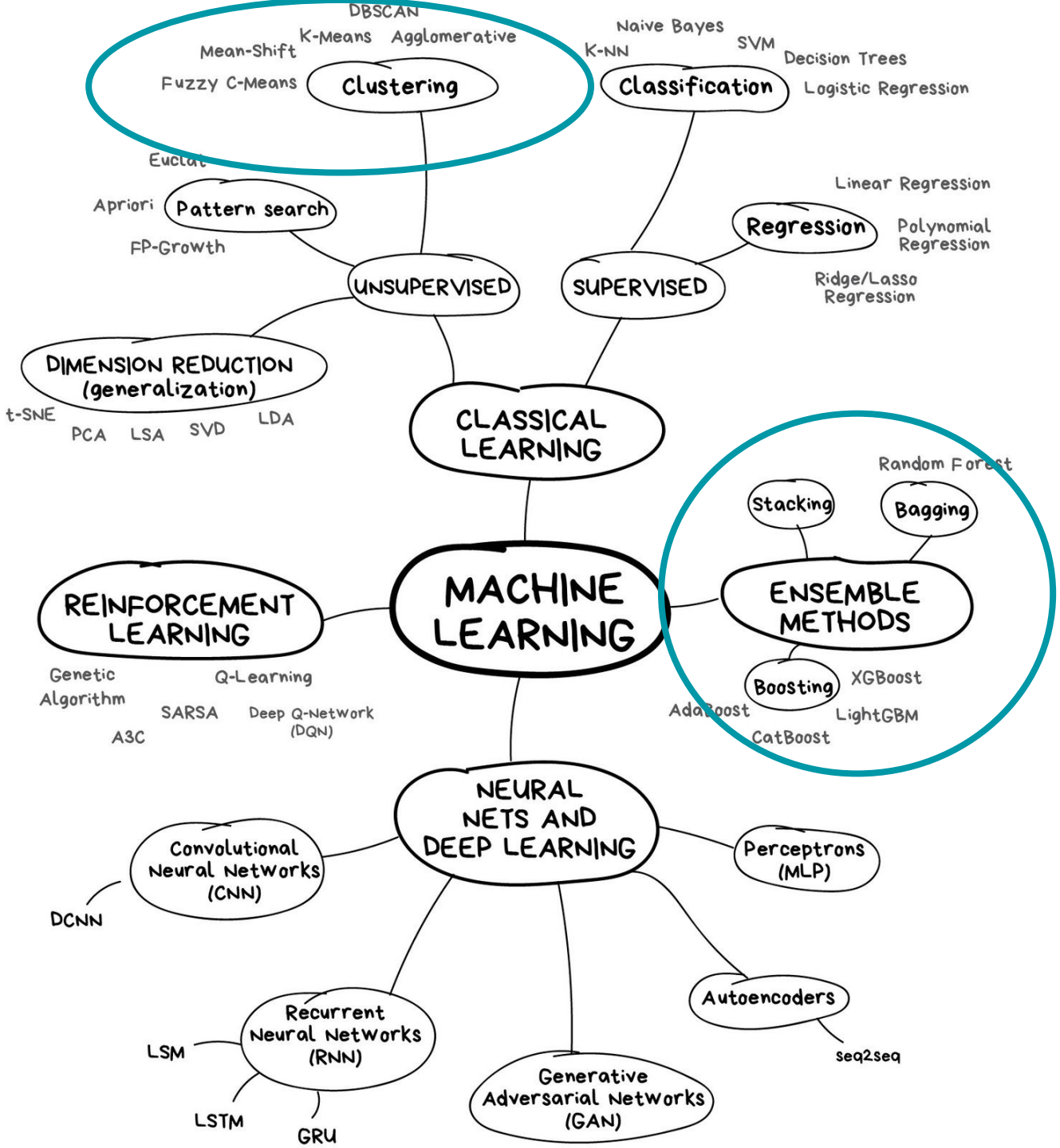


Gráfico com mais algumas categorias e exemplos de métodos



Mas antes...

Vamos fazer uma dinâmica!

Acessem o formulário aqui:

https://forms.gle/Cztn2aEqKA5dgiY19

Ensemble

O que é ensemble?

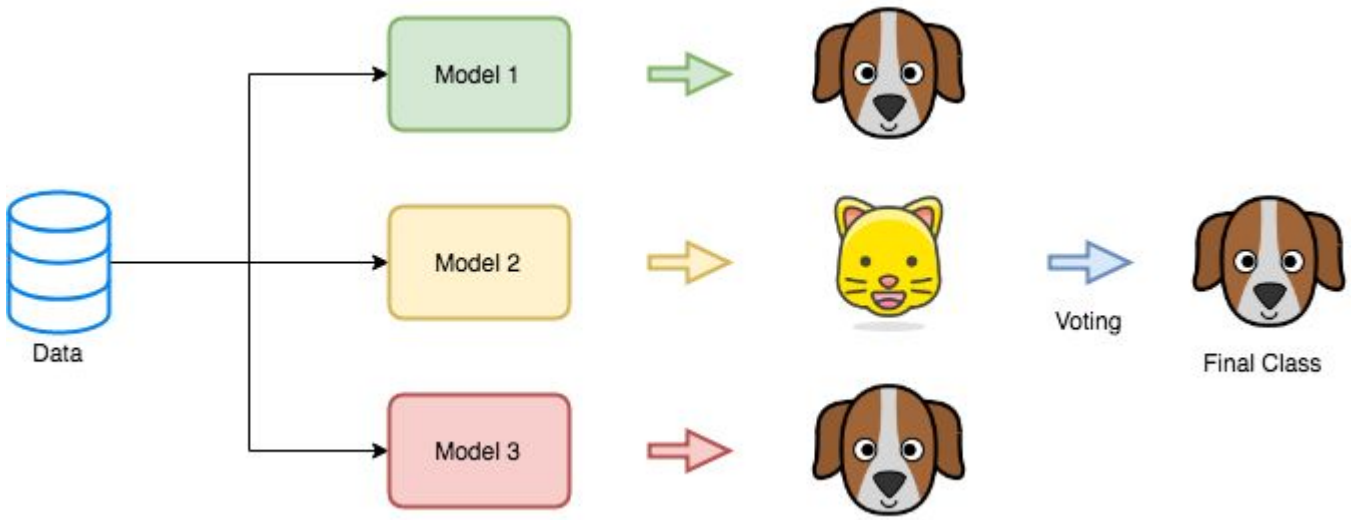
Junção de vários modelos, geralmente mais fracos, que juntos geram um melhor preditor. Basicamente segue a ideia de que várias "cabeças" pensam melhor do que uma.

Vamos ver na prática:
<Notebook>

Tipos de Ensemble:

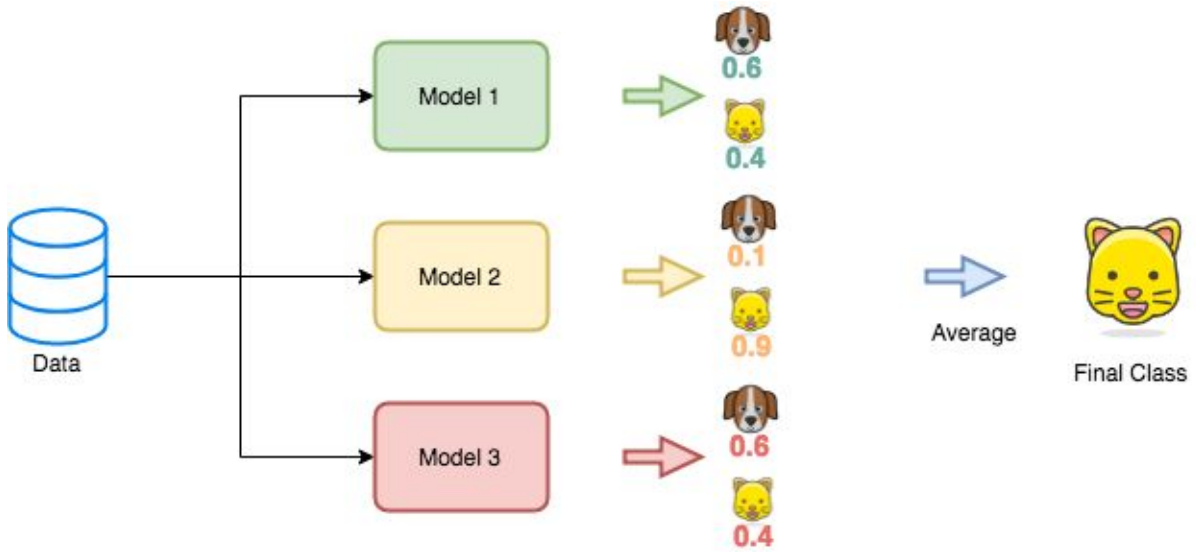
- Voting Based Classifier (o que acabamos de ver):
  - Majority Vote
  - Average Classifier
- Stacking
- Boosting
- Bagging

Majority Vote



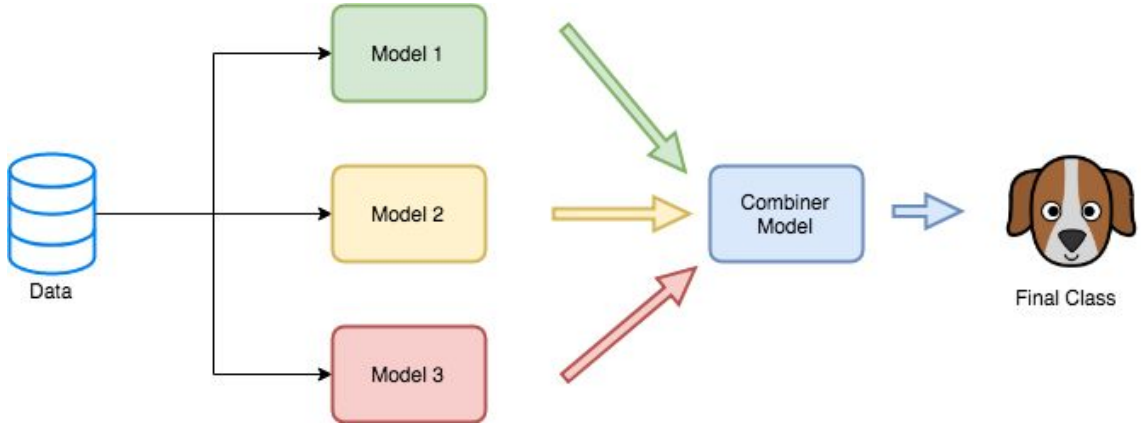
A ideia é fazer uma votação entre as predições dos modelos. A classe que tiver mais votos vence. Também podemos ter uma variação desse algoritmo, o **Weighted Voting Classifier**, em que na votação alguns modelos tem mais peso que outros.

Average Classifier



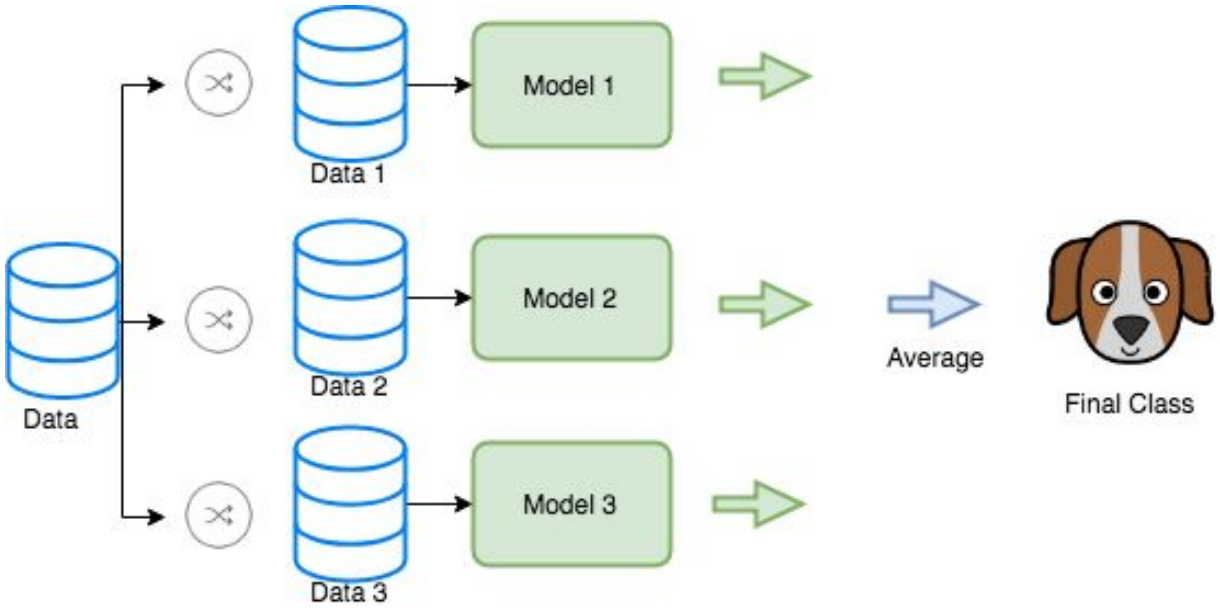
A ideia é similar ao anterior, porém ao invés de uma votação é calculada a média das predições. Da mesma forma podemos ter alguns modelos com mais peso que outros tendo um **Weighted Average Classifier**

Stacking



Nesse modelo as predições dos modelos anteriores são combinadas por um outro modelo para obter a saída final. Podem ser criadas **várias camadas** com modelos diferentes.

Bagging



Todos os modelos deste tipo de ensemble são **do mesmo algoritmo**, porém os dados de entrada de cada um são **amostras do dado original**, com a mesma quantidade de dados do dataset original, selecionadas usando o método **bootstrap** (aleatória com repetição).

Ex.: **RANDOM FOREST**

Random Forest

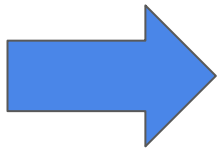
Etapas do algoritmo:

1. Criar dataset **com bootstrap** (seleção aleatória com repetição)

Random Forest

DATASET ORIGINAL

Cor	Estampa	Categoria	Bem avaliado
Verde	Flores	Casaco	Não
Azul	Liso	Casaco	Sim
Amarelo	Flores	Saia	Não
Azul	Flores	Saia	Sim



NOVO DATASET

Cor	Estampa	Categoria	Bem avaliado
Verde	Flores	Casaco	Não
Azul	Liso	Casaco	Sim
Amarelo	Flores	Saia	Não
Azul	Flores	Saia	Sim

* Perceba que a linha 1 foi selecionada duas vezes, enquanto a linha 4 não foi selecionada nenhuma vez nesse exemplo

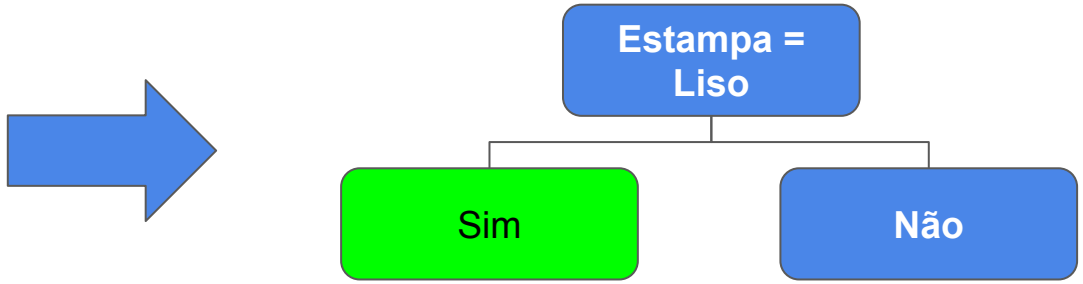
Random Forest

Etapas do algoritmo:

1. Criar dataset **com bootstrap** (seleção aleatória com repetição)
2. Criar uma árvore de decisão para o novo dataset utilizando um subconjunto randômico de variáveis

Random Forest

Cor	Estampa	Categoria	Bem avaliado
Verde	Flores	Casaco	Não
Azul	Liso	Casaco	Sim
Amarelo	Flores	Saia	Não
Azul	Flores	Saia	Sim



Random Forest

Etapas do algoritmo:

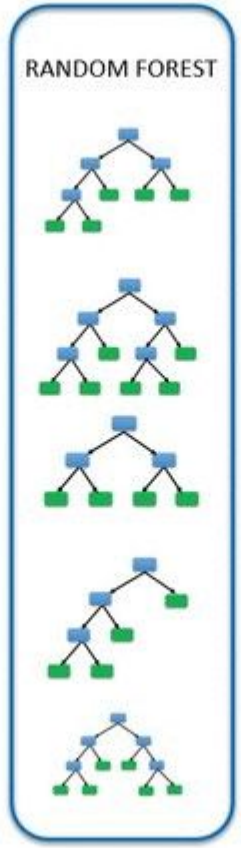
1. Criar dataset **com bootstrap** (seleção aleatória com repetição)
2. Criar uma árvore de decisão para o novo dataset utilizando um subconjunto randômico de variáveis
3. Repita esse processo várias e várias vezes, criando diferentes árvores (em média 100 árvores)
4. O classificador final é a média (ou voto) de todas as árvores

Random Forest

OOB Score:

Cada dado que não foi utilizado em uma árvore é usado para calcular o desempenho da mesma.

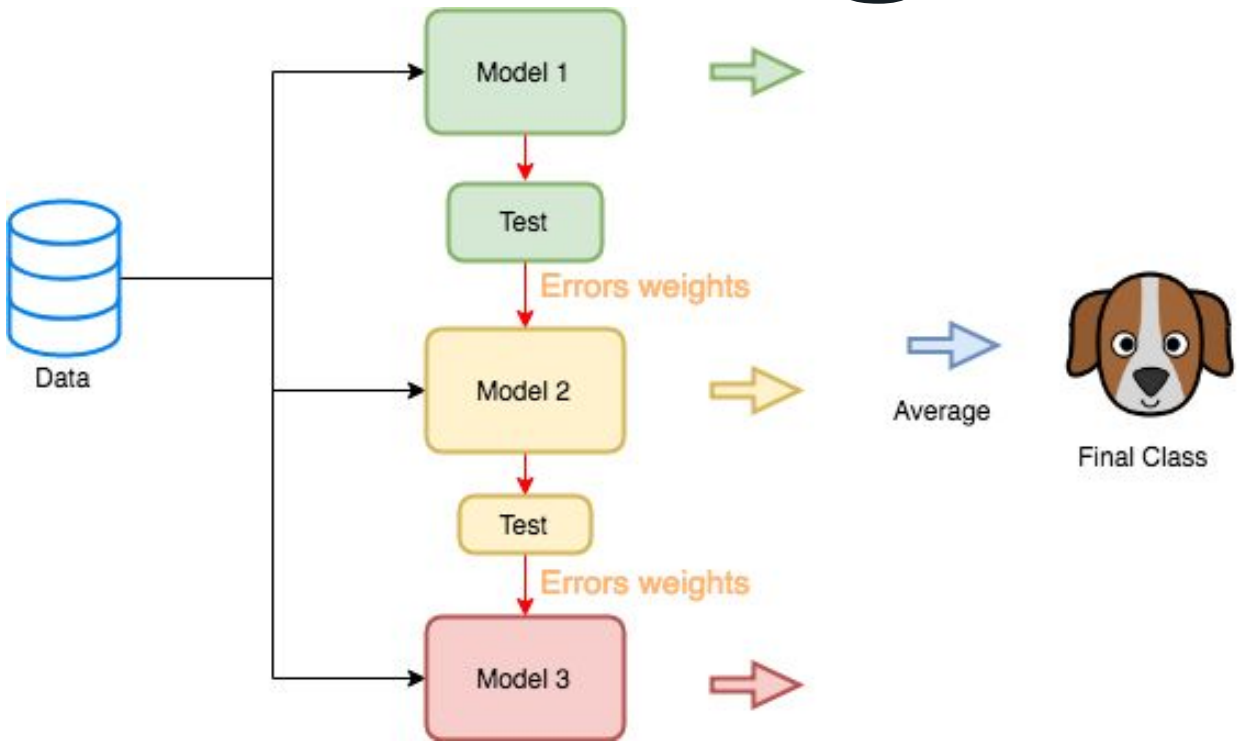
NEW DATA ARRIVES FOR TESTING



MISTAKES	CORRECT PREDICTIONS
0	0

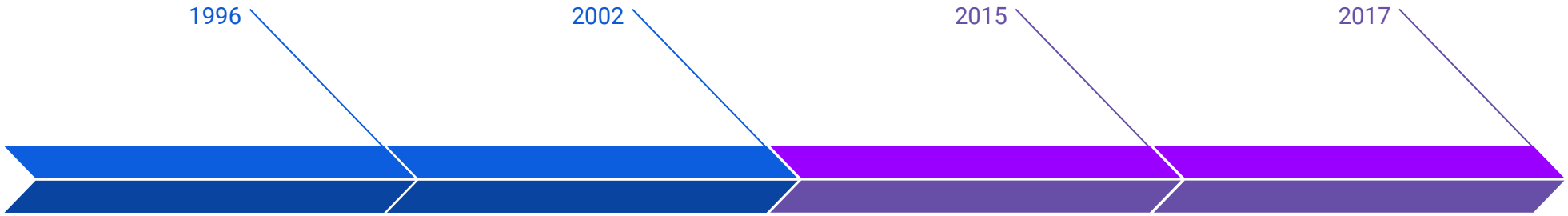
Implementando Random Forest: <Notebook>

Boosting



Os modelos são treinados com os mesmos datasets, porém os pesos das instâncias são ajustados de acordo com o erro das predições anteriores. Ex. XGBoost, Light GBM

Implementações do Boosting



AdaBoost

Implementa a ideia de boosting geralmente usando como classificador fraco árvores de nível 1 (stump).

Gradient Boosting

Define um classificador inicial, calcula os resíduos e cria um novo classificador a partir dele

XGBoost

Primeira implementação do GBM, muito utilizado em competições do Kaggle

Light GBM

Versão com algumas alterações na implementação para melhorar performance

* Há também o CatBoost lançado em 2018

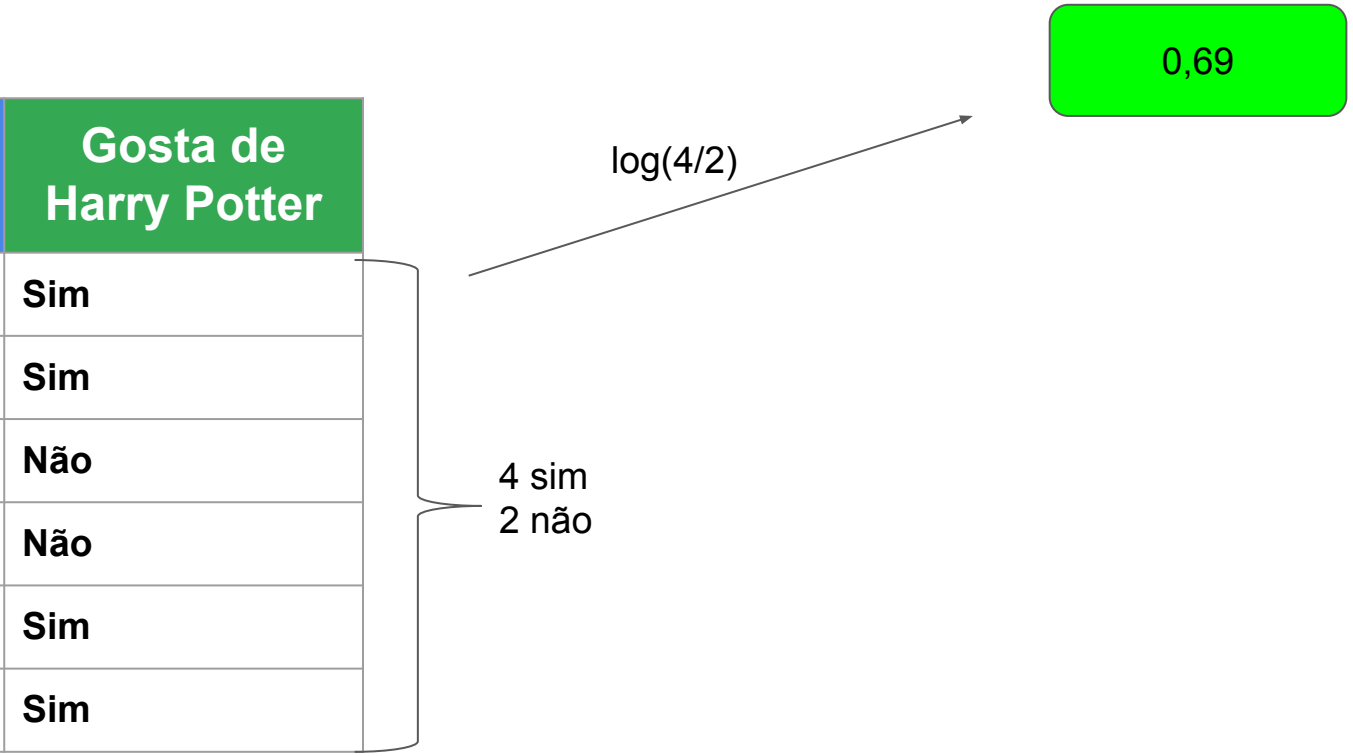
Gradient Boosting

Etapas do algoritmo:

1. Criamos uma folha inicial, que será nosso primeiro classificador. Para simplificar vamos imaginar que ela sempre será a média dos valores resposta.

Gradient Boosting

Idade	Gosta de Pipoca	Cor Favorita	Gosta de Harry Potter
12	Sim	Azul	**Sim**
87	Sim	Verde	**Sim**
44	Não	Azul	**Não**
19	Sim	Vermelho	**Não**
32	Não	Verde	**Sim**
14	Não	Azul	**Sim**



dataset baseado no do StatQuest

Gradient Boosting

Idade	Gost Pip
12	S
87	
44	
19	S
32	
14	Não

Precisamos transformar nosso resultado em probabilidade para utilizar o classificador. Vamos usar a Função Logística:

$$\frac{e^{\log(odds)}}{1 + e^{\log(odds)}}$$

0,69

prob=0,67

Imagens do canal StatQuest

Gradient Boosting

Etapas do algoritmo:

1. Criamos uma folha inicial, que será nosso primeiro classificador. Para simplificar vamos imaginar que ela sempre será a média dos valores resposta.
2. Usaremos o nosso classificador para calcular os resíduos (erros)

Gradient Boosting

0,7

* arredondado para facilitar

prob=0,7

Idade	Gosta de Pipoca	Cor Favorita	Gosta de Harry Potter	Resíduos
12	Sim	Azul	**Sim**	**0,3**
87	Sim	Verde	**Sim**	**0,3**
44	Não	Azul	**Não**	**-0,7**
19	Sim	Vermelho	**Não**	**-0,7**
32	Não	Verde	**Sim**	**0,3**
14	Não	Azul	**Sim**	**0,3**

Assumimos: Sim=1 e Não=0

Resíduo = (Valor Observado - Valor Predito)

* Esse resíduo na verdade é a derivada da função de perda (Loss Function) e o chamamos de **Gradiente**

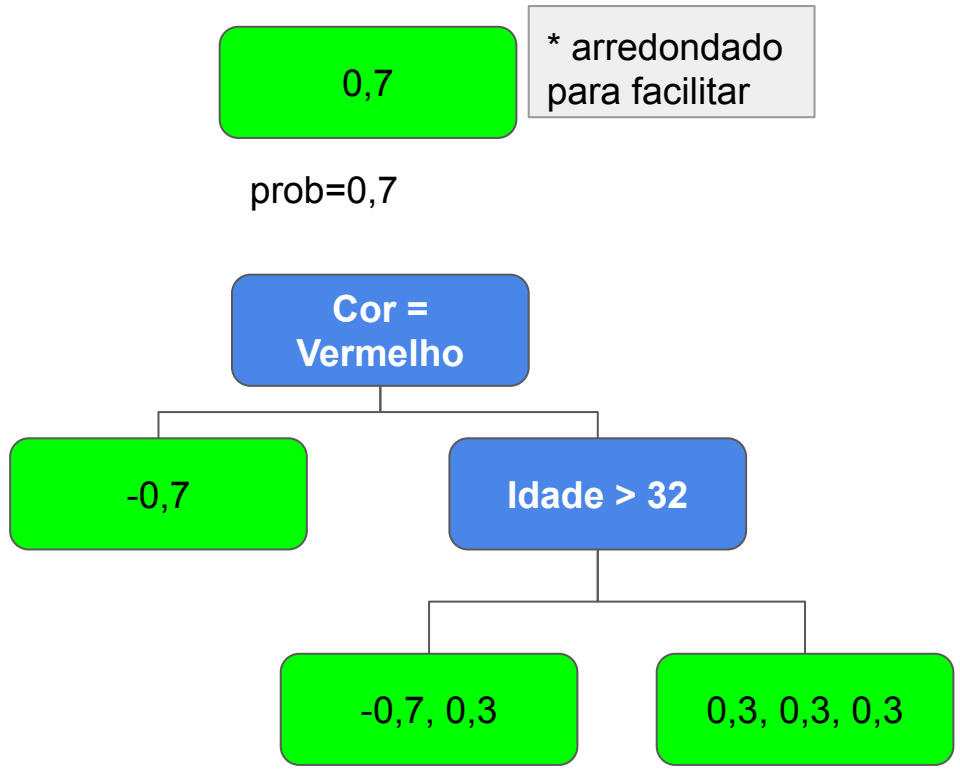
Gradient Boosting

Etapas do algoritmo:

1. Criamos **uma folha inicial**, que será nosso primeiro classificador. Para simplificar vamos imaginar que ela sempre será a média dos valores resposta.
2. Usaremos o nosso classificador para **calcular os resíduos** (erros)
3. Criamos um novo classificador para **predizer os resíduos**

Gradient Boosting

Idade	Gosta de Pipoca	Cor Favorita	Gosta de Harry Potter	Resíduos
12	Sim	Azul	**Sim**	**0,3**
87	Sim	Verde	**Sim**	**0,3**
44	Não	Azul	**Não**	**-0,7**
19	Sim	Vermelho	**Não**	**-0,7**
32	Não	Verde	**Sim**	**0,3**
14	Não	Azul	**Sim**	**0,3**



*no GB há um limite de folhas, geralmente entre 8 e 32.

Gradient Boosting

Etapas do algoritmo:

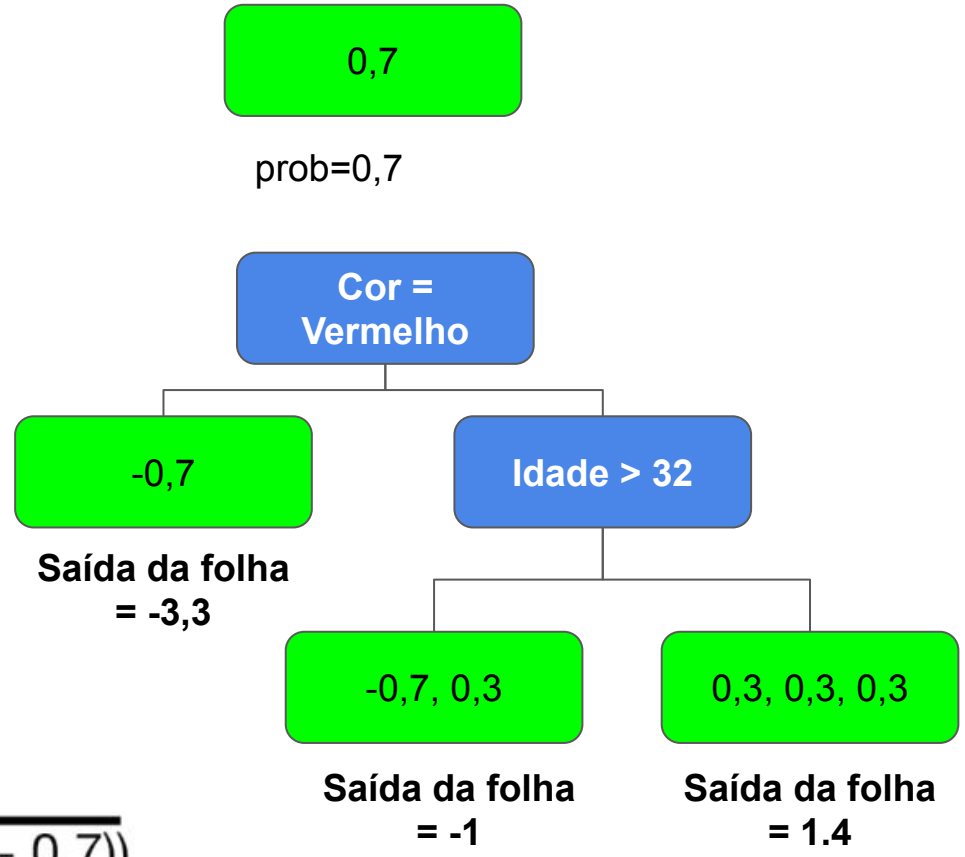
1. Criamos **uma folha inicial**, que será nosso primeiro classificador. Para simplificar vamos imaginar que ela sempre será a média dos valores resposta.
2. Usaremos o nosso classificador para **calcular os resíduos** (erros)
3. Criamos um novo classificador para **predizer os resíduos**
4. Usaremos a **“somatória” dos classificadores** para realizar a nova predição

Gradient Boosting

$$\frac{\sum \text{Residual}_i}{\sum \left[\text{Previous Probability}_i \times (1 - \text{Previous Probability}_i)\right]}$$

$$\frac{-0.7}{0.7 \times (1 - 0.7)}$$

$$\frac{0.3 + -0.7}{(0.7 \times (1 - 0.7)) + (0.7 \times (1 - 0.7))}$$

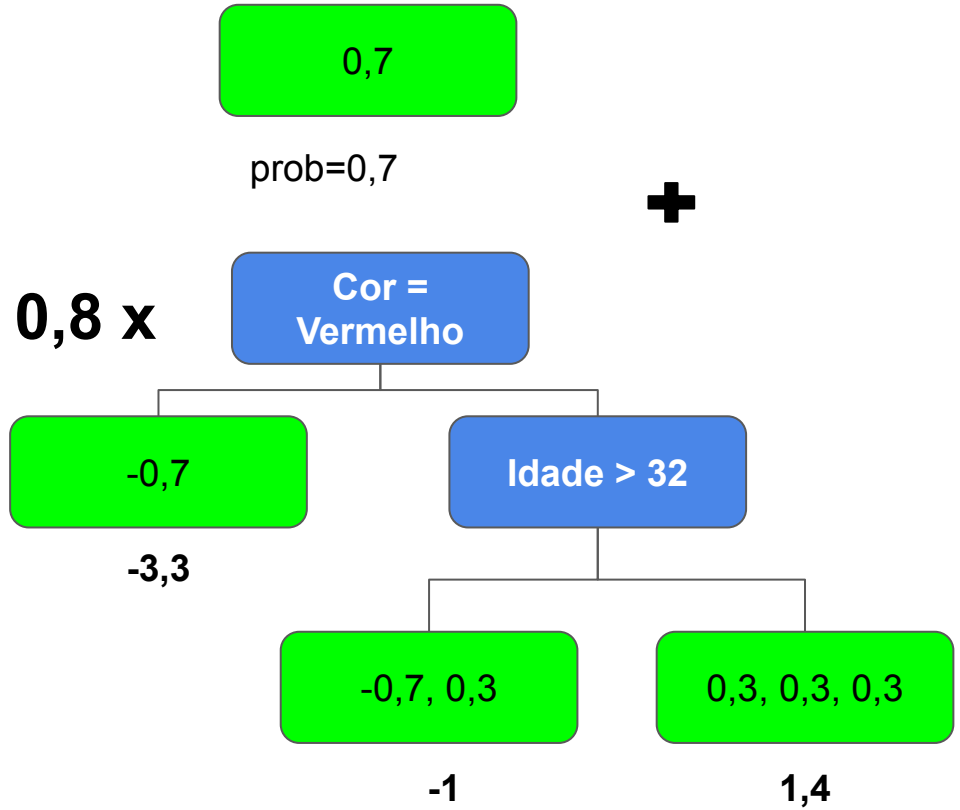


* Se fosse uma regressão apenas somaríamos os resultados, como é uma classificação precisamos antes fazer transformações probabilísticas

Imagens do canal StatQuest

Gradient Boosting

Idade	Gosta de Pipoca	Cor Favorita	Gosta de Harry Potter	Rs	Logs
12	Sim	Azul	**Sim**	**0,3**	**1,8**
87	Sim	Verde	**Sim**	**0,3**	**-0,1**
44	Não	Azul	**Não**	**-0,7**	**-0,1**
19	Sim	Vermelho	**Não**	**-0,7**	**-1,94**
32	Não	Verde	**Sim**	**0,3**	**1,8**
14	Não	Azul	**Sim**	**0,3**	**1,8**

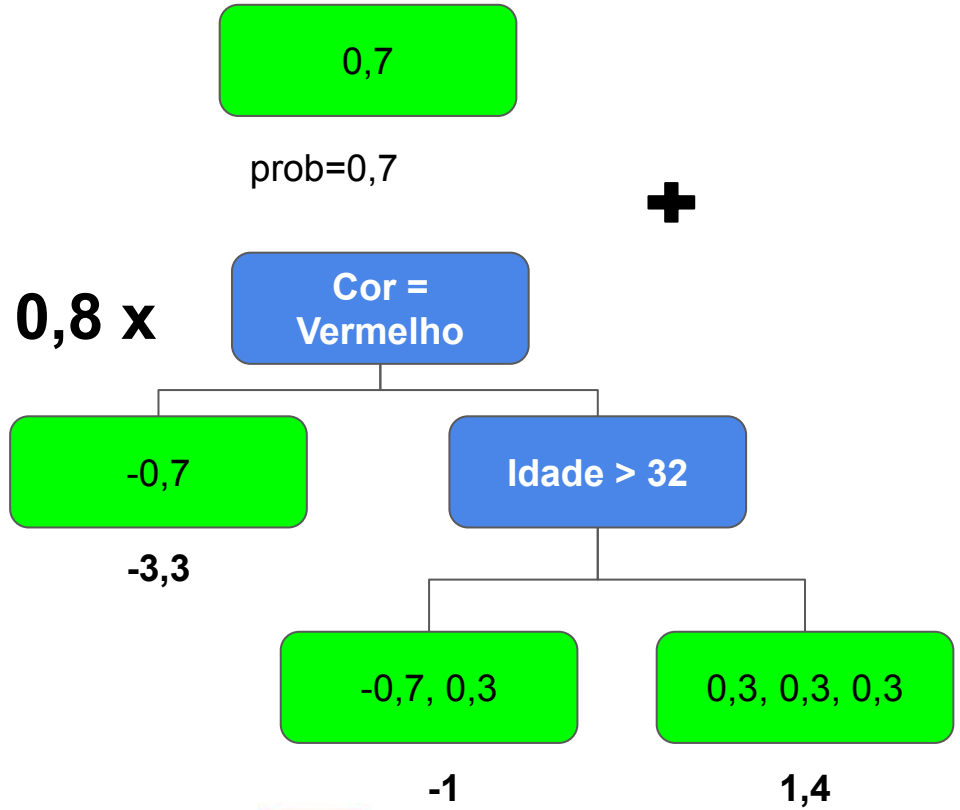


logs: 0,7 + (0,8 * 1,4) = 1,8

*Learning Rate geralmente é pequeno, em torno de 0,1 ou menor.

Gradient Boosting

Idade	Gosta de Pipoca	Cor Favorita	Gosta de HP	Rs	logs	probs
12	Sim	Azul	**Sim**	**0,3**	**1,8**	**0,9**
87	Sim	Verde	**Sim**	**0,3**	**-0,1**	**0,5**
44	Não	Azul	**Não**	**-0,7**	**-0,1**	**0,5**
19	Sim	Vermelho	**Não**	**-0,7**	**-1,94**	**0,1**
32	Não	Verde	**Sim**	**0,3**	**1,8**	**0,9**
14	Não	Azul	**Sim**	**0,3**	**1,8**	**0,9**

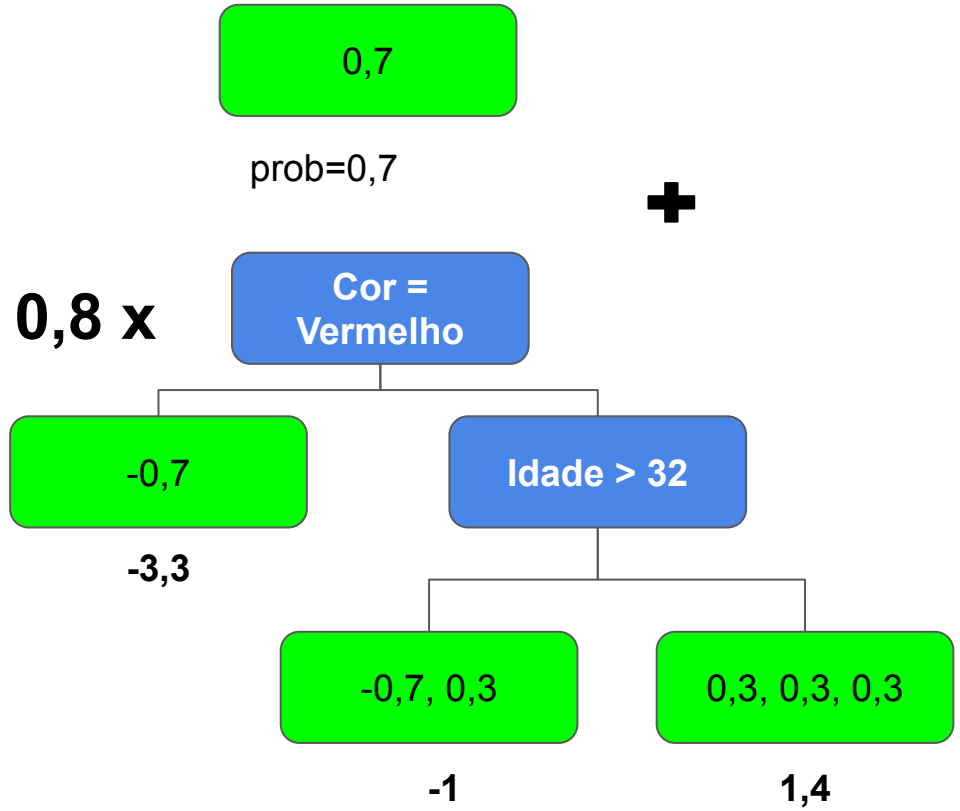


$$\text{Probability} = \frac{e^{1.8}}{1 + e^{1.8}} = 0.9$$

*Learning Rate geralmente é pequeno, em torno de 0,1 ou menor.

Gradient Boosting

Idade	Gosta de Pipoca	Cor Favorita	Gosta de HP	Rs	pb	Rs 2
12	Sim	Azul	**Sim**	**0,3**	**0,9**	**0,1**
87	Sim	Verde	**Sim**	**0,3**	**0,5**	**0,5**
44	Não	Azul	**Não**	**-0,7**	**0,5**	**-0,5**
19	Sim	Vermelho	**Não**	**-0,7**	**0,1**	**-0,1**
32	Não	Verde	**Sim**	**0,3**	**0,9**	**0,1**
14	Não	Azul	**Sim**	**0,3**	**0,9**	**0,1**



*Learning Rate geralmente é pequeno, em torno de 0,1 ou menor.

predizer os novos resíduos:
(Valor Observado - Novo Valor Previsto)

Gradient Boosting

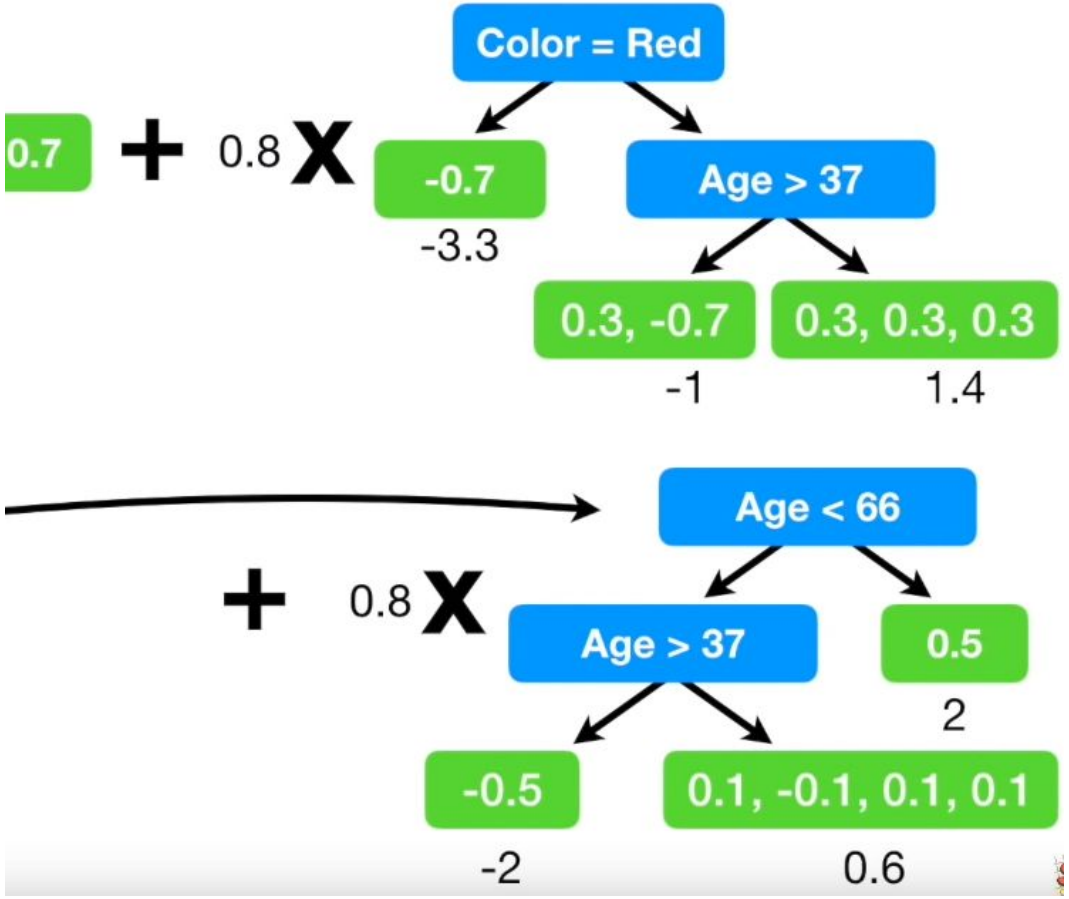
Etapas do algoritmo:

1. Criamos **uma folha inicial**, que será nosso primeiro classificador. Para simplificar vamos imaginar que ela sempre será a média dos valores resposta.
2. Usaremos o nosso classificador para **calcular os resíduos** (erros)
3. Criamos um novo classificador para **predizer os resíduos**
4. Usaremos a **“somatória” dos classificadores** para realizar a nova predição
5. Crie um novo classificador para **predizer os novos resíduos**

predizer os novos resíduos

Gradient Boosting

Idade	Gosta de Pipoca	Cor Favorita	Gosta de HP	Rs	pb	Rs 2
12	Sim	Azul	**Sim**	**0,3**	**0,9**	**0,1**
87	Sim	Verde	**Sim**	**0,3**	**0,5**	**0,5**
44	Não	Azul	**Não**	**-0,7**	**0,5**	**-0,5**
19	Sim	Vermelho	**Não**	**-0,7**	**0,1**	**-0,1**
32	Não	Verde	**Sim**	**0,3**	**0,9**	**0,1**
14	Não	Azul	**Sim**	**0,3**	**0,9**	**0,1**



*Para fazer uma nova predição eu percorro a árvore calculando o log e posteriormente a probabilidade

dataset baseado no do StatQuest

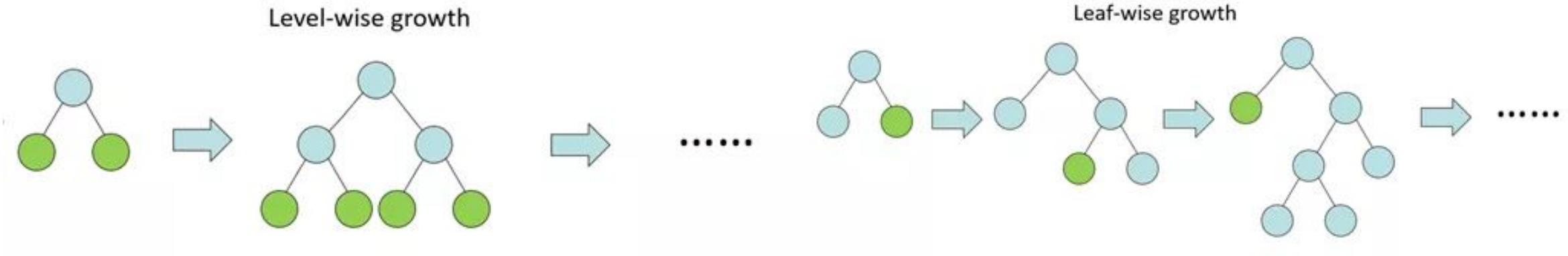
Gradient Boosting

Etapas do algoritmo:

1. Criamos **uma folha inicial**, que será nosso primeiro classificador. Para simplificar vamos imaginar que ela sempre será a média dos valores resposta.
2. Usaremos o nosso classificador para **calcular os resíduos** (erros)
3. Criamos um novo classificador para **predizer os resíduos**
4. Usaremos a **“somatória” dos classificadores** para realizar a nova predição
5. Crie um novo classificador para **predizer os novos resíduos**

... e assim por diante até atingir o número máximo de árvores (geralmente 100) ou um valor mínimo de resíduo

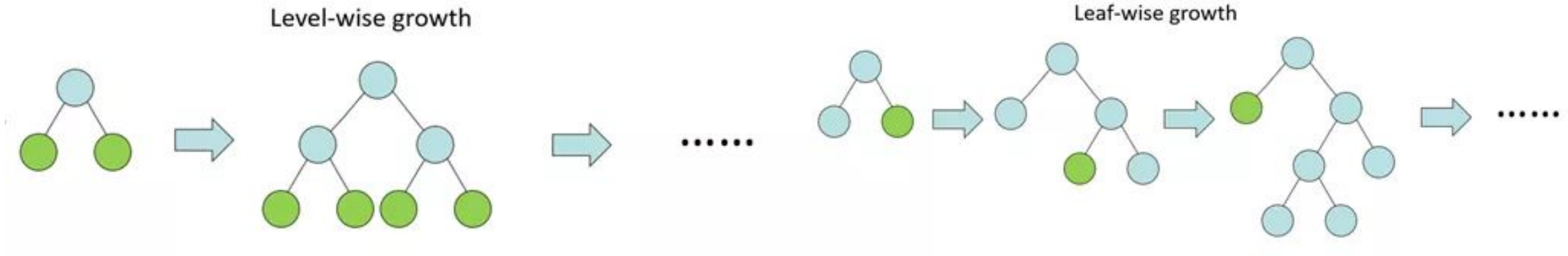
XG Boosting vs Light GBM



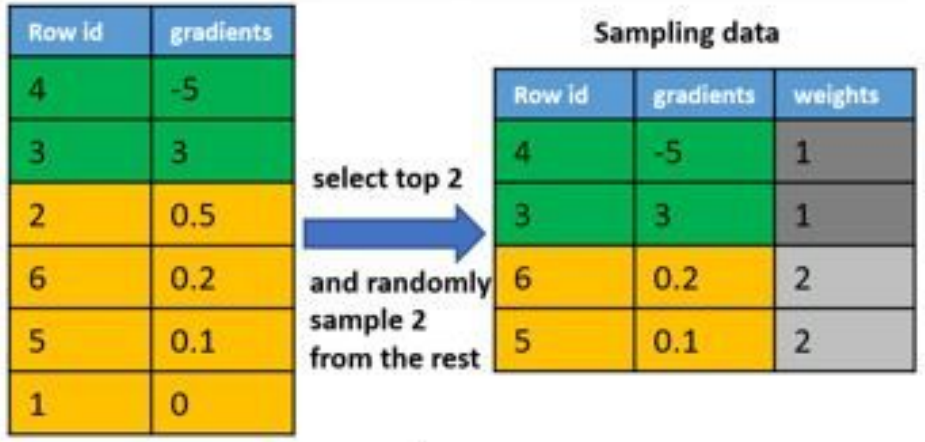
- Cresce as árvores em nível**
- Separação por histograma: cria bins (categorias) para as features contínuas.

- Cresce as árvores por folha
- Separação por histograma, porém faz uma vez para todo o treinamento
- usa GOSS (Gradient Based One Side Sampling): faz downsampling do dataset:
  - 1. calcula o gradiente de cada linha
  - 2. seleciona todas as que tem um alto valor de gradiente
  - 3. faz seleção aleatória das que tem baixo valor

XG Boosting vs Light GBM

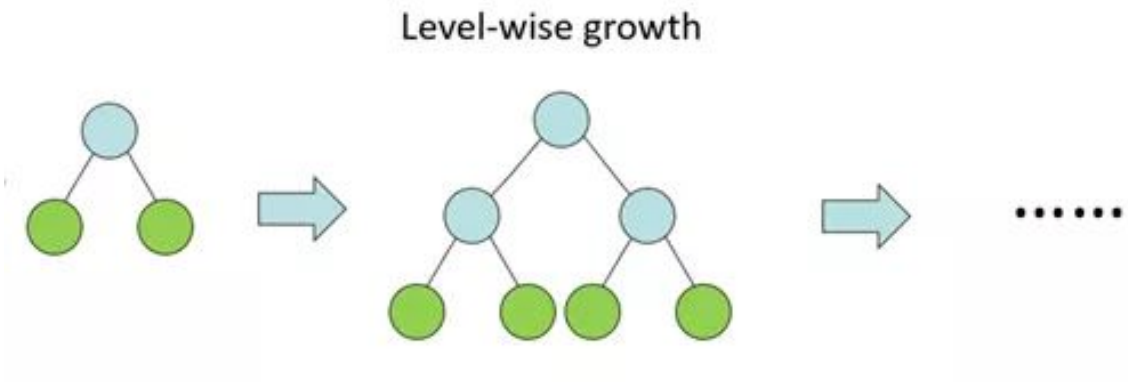


- Gradient-based One-Side Sampling

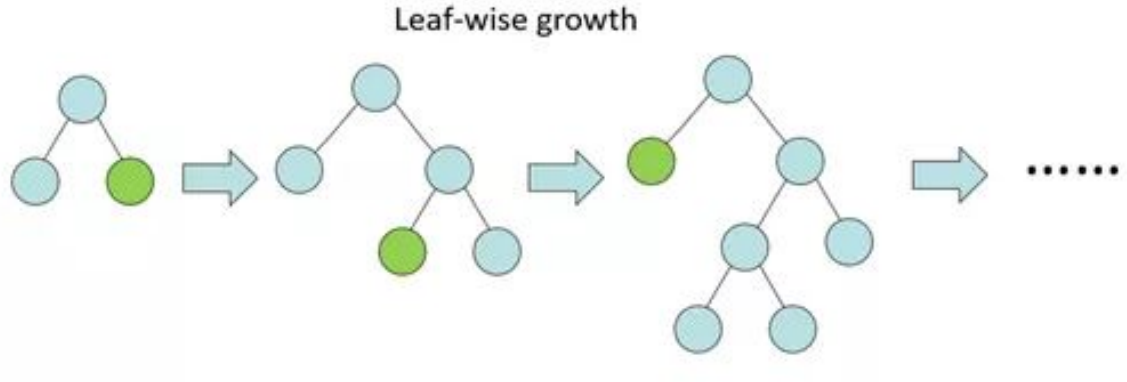


- Cresce as árvores por folha
- Separação por histograma, porém faz uma vez para todo o treinamento
- usa GOSS (Gradient Based One Side Sampling): faz downsampling do dataset:
  - 1. calcula o gradiente de cada linha
  - 2. seleciona todas as que tem um alto valor de gradiente
  - 3. faz seleção aleatória das que tem baixo valor

XG Boosting vs Light GBM



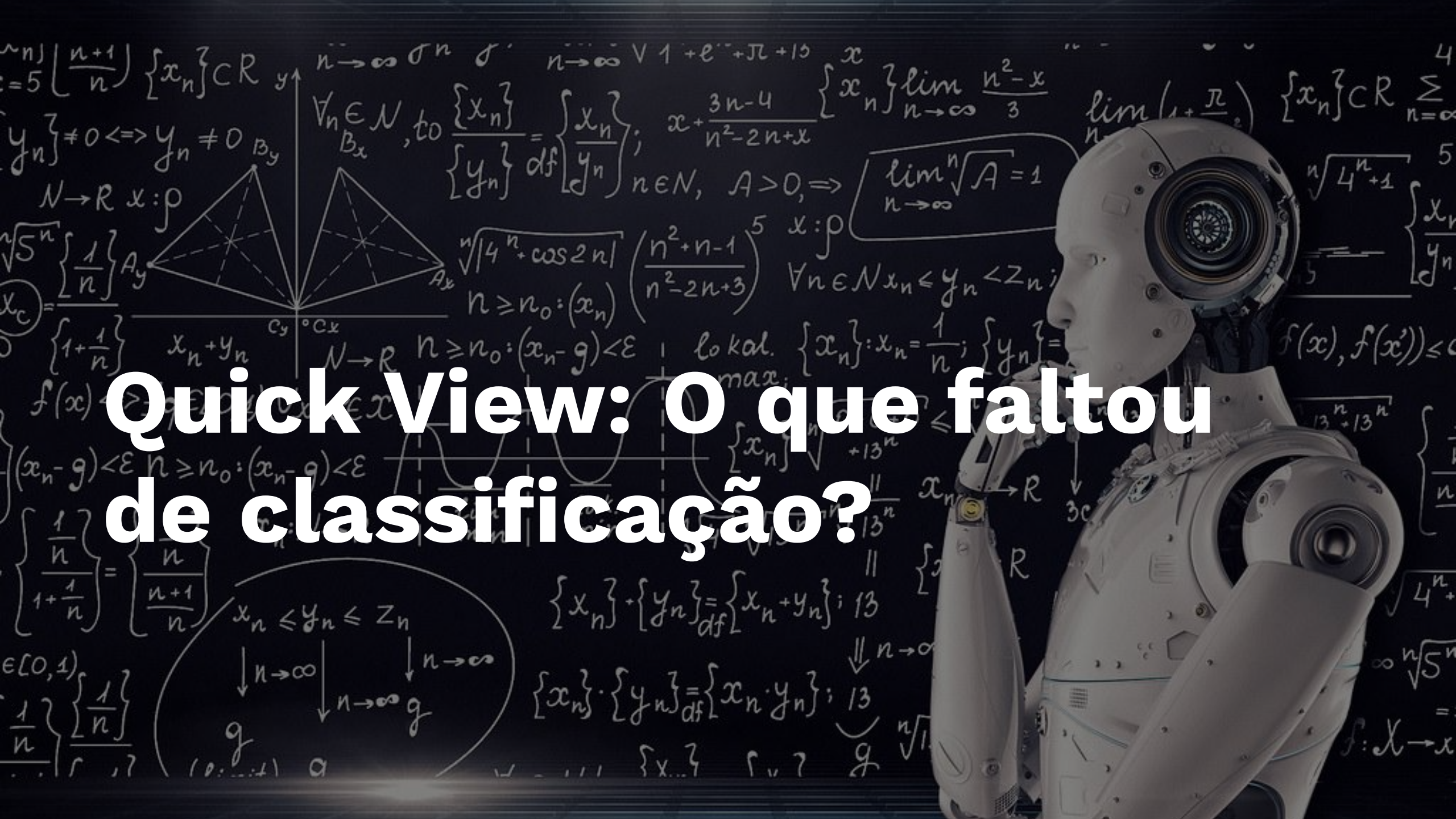
- Cresce as árvores em nível**
- Separação por histograma: cria bins (categorias) para as features contínuas.



- Cresce as árvores por folha
- Separação por histograma, porém faz uma vez para todo o treinamento
- usa GOSS (Gradient Based One Side Sampling)
- usa EFB (Exclusive Feature Bundling): diminui a quantidade de features juntando as que são esparsas de forma exclusiva, ou seja, onde está 0 em uma feature tem valor na outra.

Implementando Light GBM:

<Notebook>



Quick View: O que faltou de classificação?

Tópicos - Feature Engineering:

1. Algoritmos de redução da dimensionalidade:
    a. ReliefF (Feature selection using Relief algorithms with python example)
    b. PCA (Principal Component Analysis from Statistical and Machine Learning Perspectives)

São algoritmos que tentam diminuir a quantidade de features fazendo seleção das melhores ou combinação entre features. Como vimos os algoritmos recentes já fazem isso internamente.

Tópicos - Modelagem:

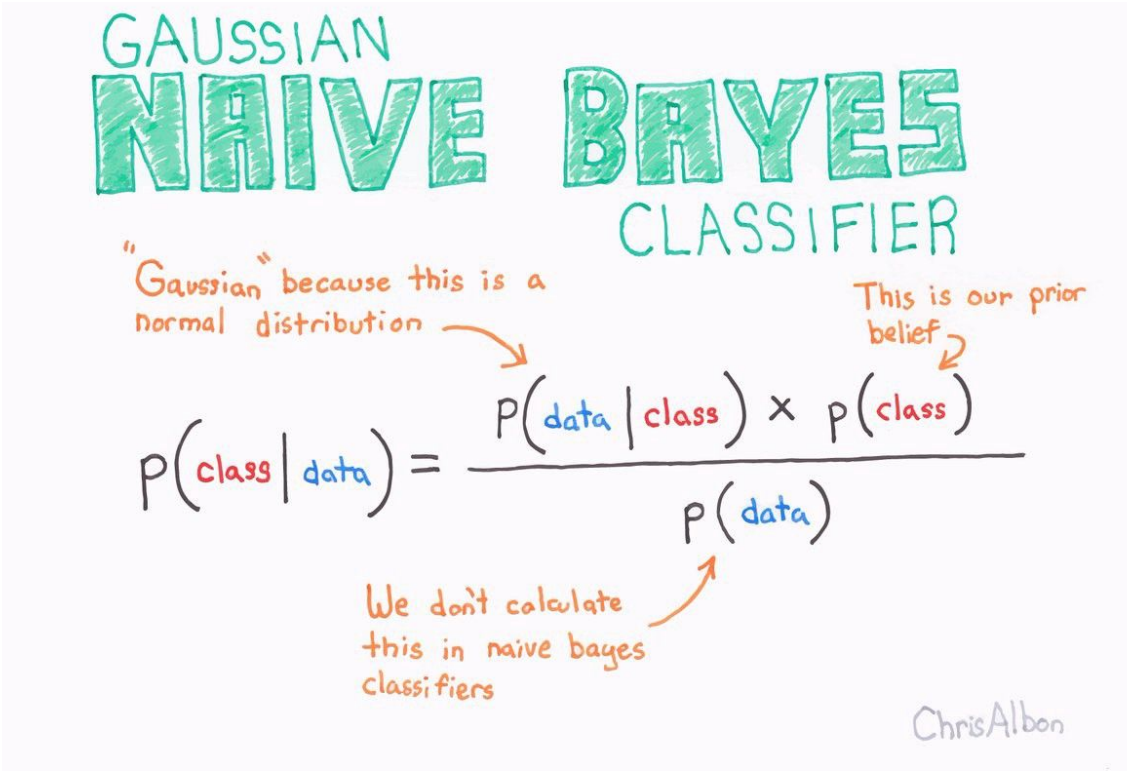
2. Mais alguns algoritmos de classificação:

   a. Naive Bayes (Naive Bayes):

   -Algoritmo probabilístico baseado no Teorema de Bayes.

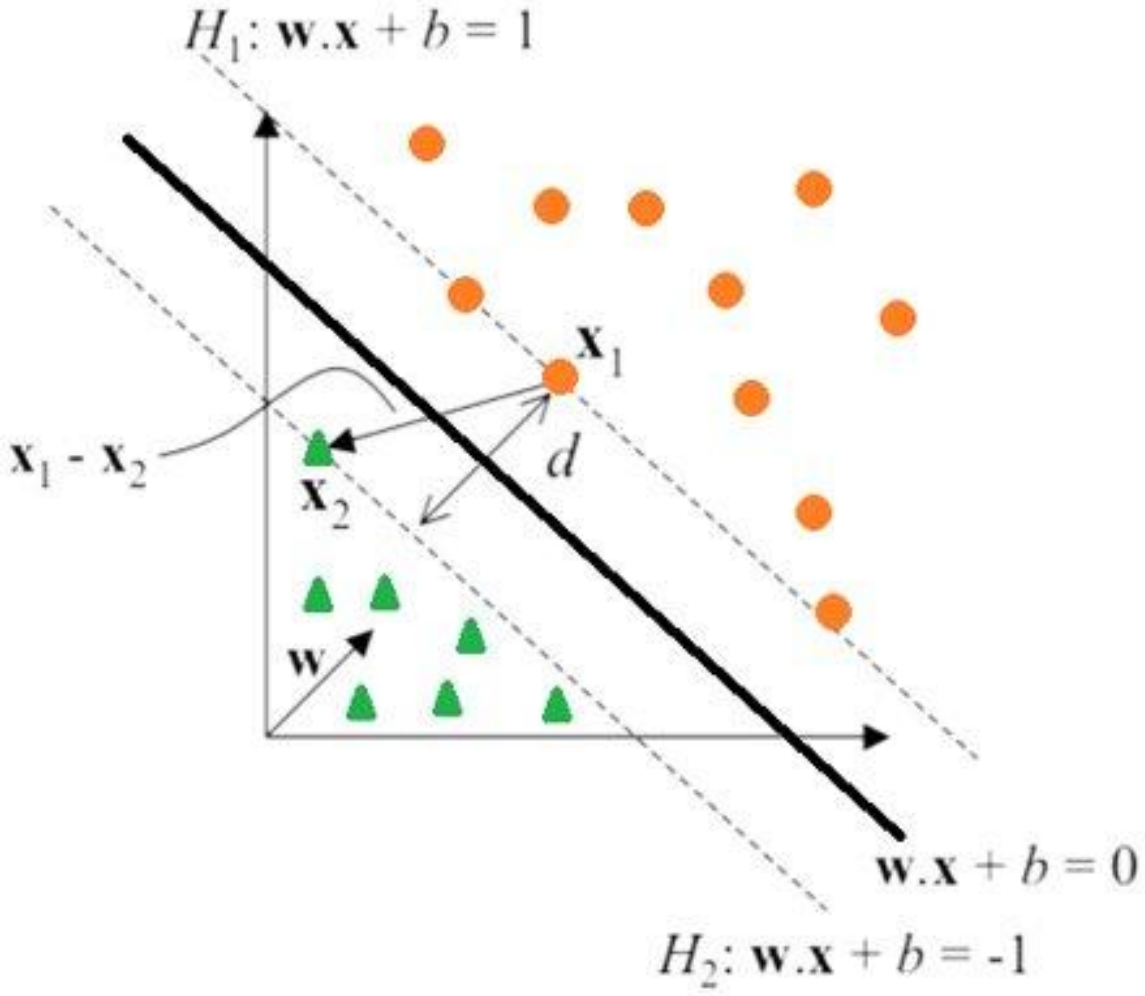
   - É naive (ingênuo) porque desconsidera a relação entre as variáveis.

   - É rápido, porém normalmente não atinge altos resultados.



Tópicos - Modelagem:

2. Mais alguns algoritmos de classificação:
    b. SVM (SVM)
    - Determina o hiperplano que separa duas classes;
    - Kernel pode ser linear ou não-linear
    - Pode atingir ótimos resultados, mas é preciso encontrar os parâmetros ideias



Tópicos - Molelagem:

3. Hiperparametrização de algorítmos:
    a. GridSearch (GridSearch na tua cara)
    b. AutoML: TPot (TPOT)

Formas de encontrar os parâmetros ideias para um algoritmo. Os algoritmos atuais costumam funcionar bem com os parâmetros default.

Tópicos - Avaliação:

4. Validação:
    a. Validação cruzada (k-fold cross validation):

       Geralmente utilizado para garantir que os parâmetros escolhidos para o algoritmo não estão causando Overfitting.

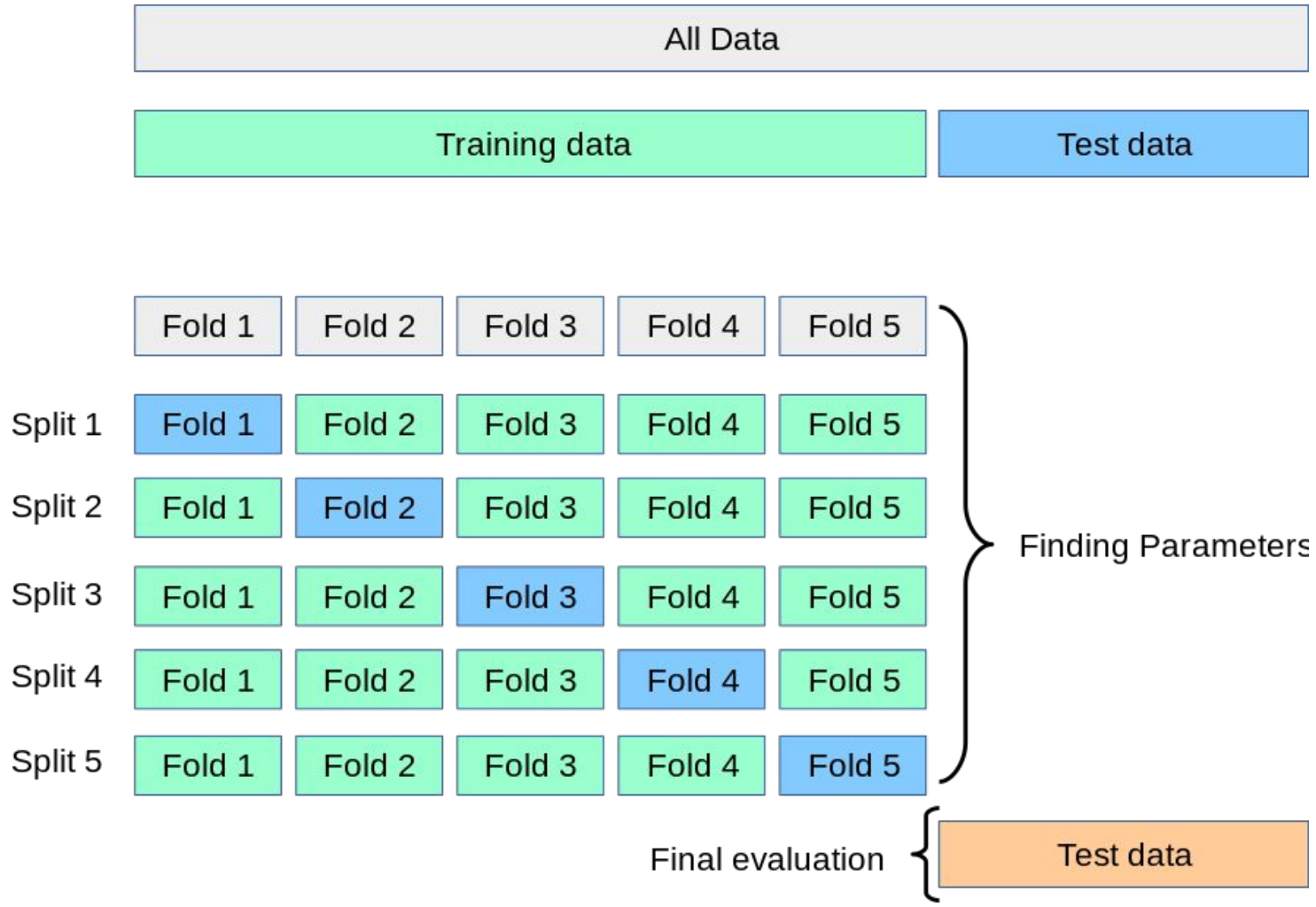
       Divide-se os dados em uma determinada quantidade de blocos (folds), e em cada rodada um desses blocos é usado para teste.

Tópicos - Avaliação:

Documentação de validação cruzada do scikit-learn.

```python
import numpy as np
from sklearn.model_selection import KFold

X = ["a", "b", "c", "d"]
kf = KFold(n_splits=2)
for train, test in kf.split(X):
    print("%s %s" % (train, test))
```

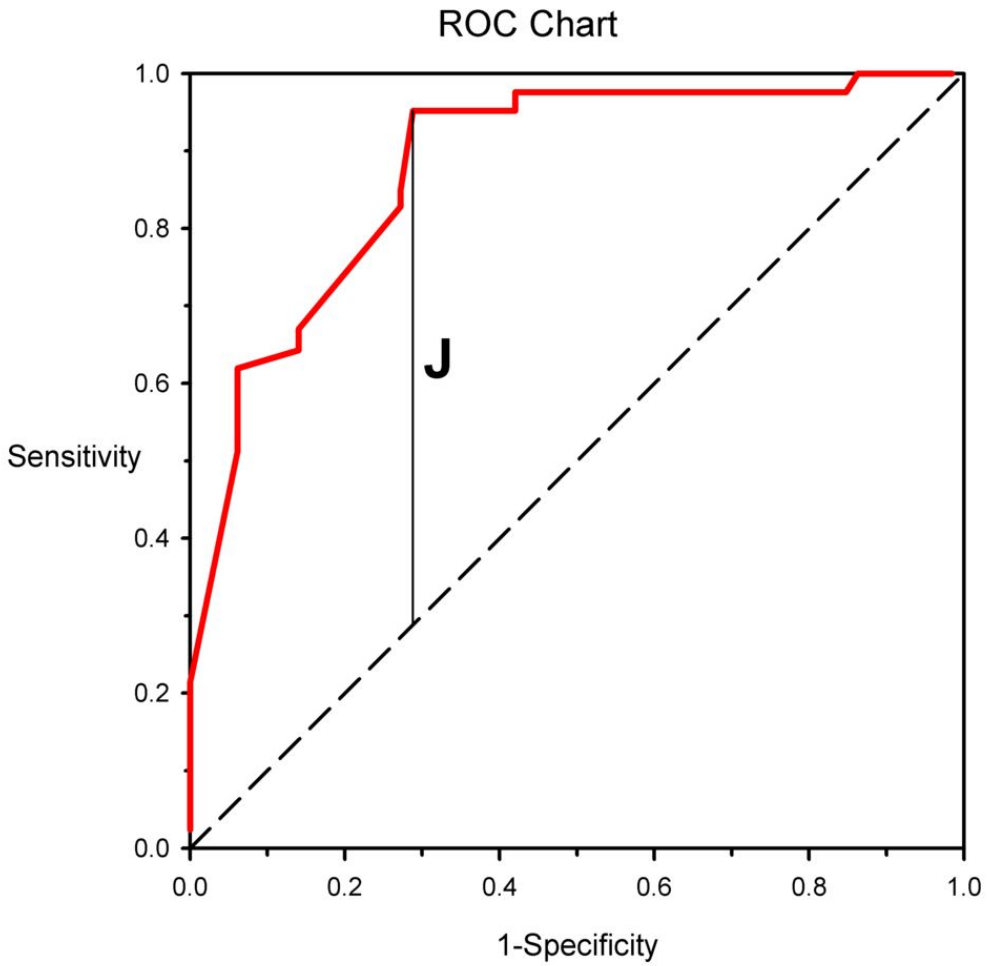


Tópicos - Avaliação:

4. Validação:

    b. Cálculo de Threshold:

    Geralmente assumimos que se a probabilidade for maior do que 0.5, o dado pertence a classe. Mas isso nem sempre é uma boa prática, há fórmulas para definir o melhor threshold baseado na curva ROC. Uma delas é o Índice de Youden.



Exercícios para casa: Pratiquem!!

1 - Tenho um dataset em que o minha tarefa é dizer o preço de um produto:

a) Que tipo de algoritmo devo usar? Supervisionado ou não-supervisionado? De classificação ou de regressão?
b) Posso usar uma Random Forest para essa Tarefa?
c) Se no meu dataset tiver a variável “valor por kg” eu posso usar como entrada para o meu algoritmo? Por quê?

Lembram do desafio do titanic?

2- Tentem resolvê-lo com cada um dos algoritmos vistos hoje.

No notebook tentem ir explicando por passos o que estão fazendo, tanto a parte de Feature Engineering quando a Modelagem em si.

Aproveitem para colocar no github e usar como início de portfólio :D

Obrigada!

Dúvidas?
Podem nos procurar! :D