MBA Business Analytics e Big Data

Análise Preditiva

Prof. Dr. João Rafael Dias

1º semestre - 2020

Agenda

Na aula de hoje...

Introdução e motivações
Feature engineering
Tratamento de variáveis
Transformação de variáveis
Arquivos de trabalho
Prática no RStudio

Aprendizagem supervisionada
Regressão e classificação
Formas de treino e validação
Bias-variance trade-off
Avaliação e comparação de modelos
Prática no RStudio

Regressão linear múltipla
Coeficiente de determinação
Regressão logística
Odds e log odds
Comparação entre as regressões
Multicolinearidade
Seleção de variáveis *step-wise*
Prática no RStudio

Estrutura de uma árvore de decisão
Intuição
Particionamento dos nós na regressão e classificação
Poda da árvores vs *overfitting*

Modelos de *ensemble*
Bootstrap
Random forest
Adaptive boosting
Prática no RStudio

Ensemble methods

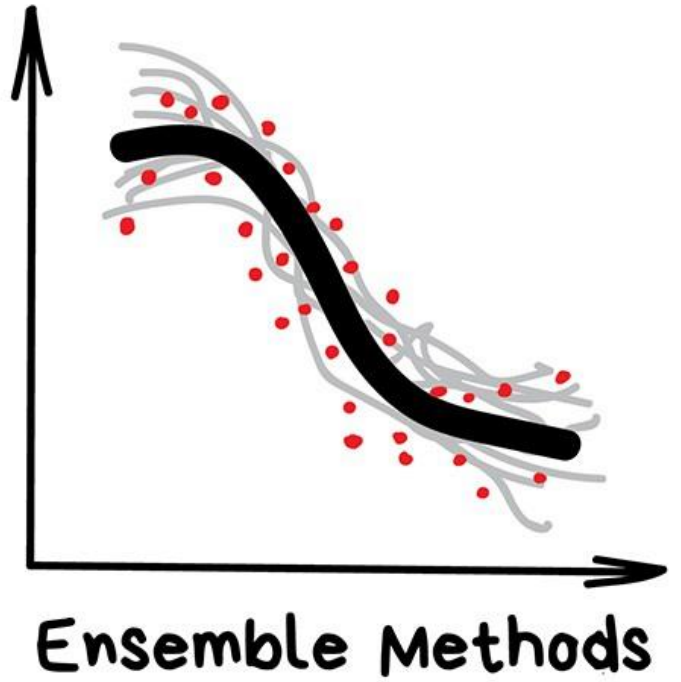
Ensemble methods

Overview

- Quando aplicamos um algoritmo de classificação em diferentes amostras da população, os resultados obtidos podem variar consideravelmente (grande variância)
- Um mesmo indivíduo que esteja em todas essas amostras, pode ser classificado de forma diferente pelas diversas regras obtidas
- Emprega-se as técnicas de *ensemble* para combinar o resultado de diversas regras de regressão e classificação mais simples para obter uma combinada com melhorar a acurácia

Definição

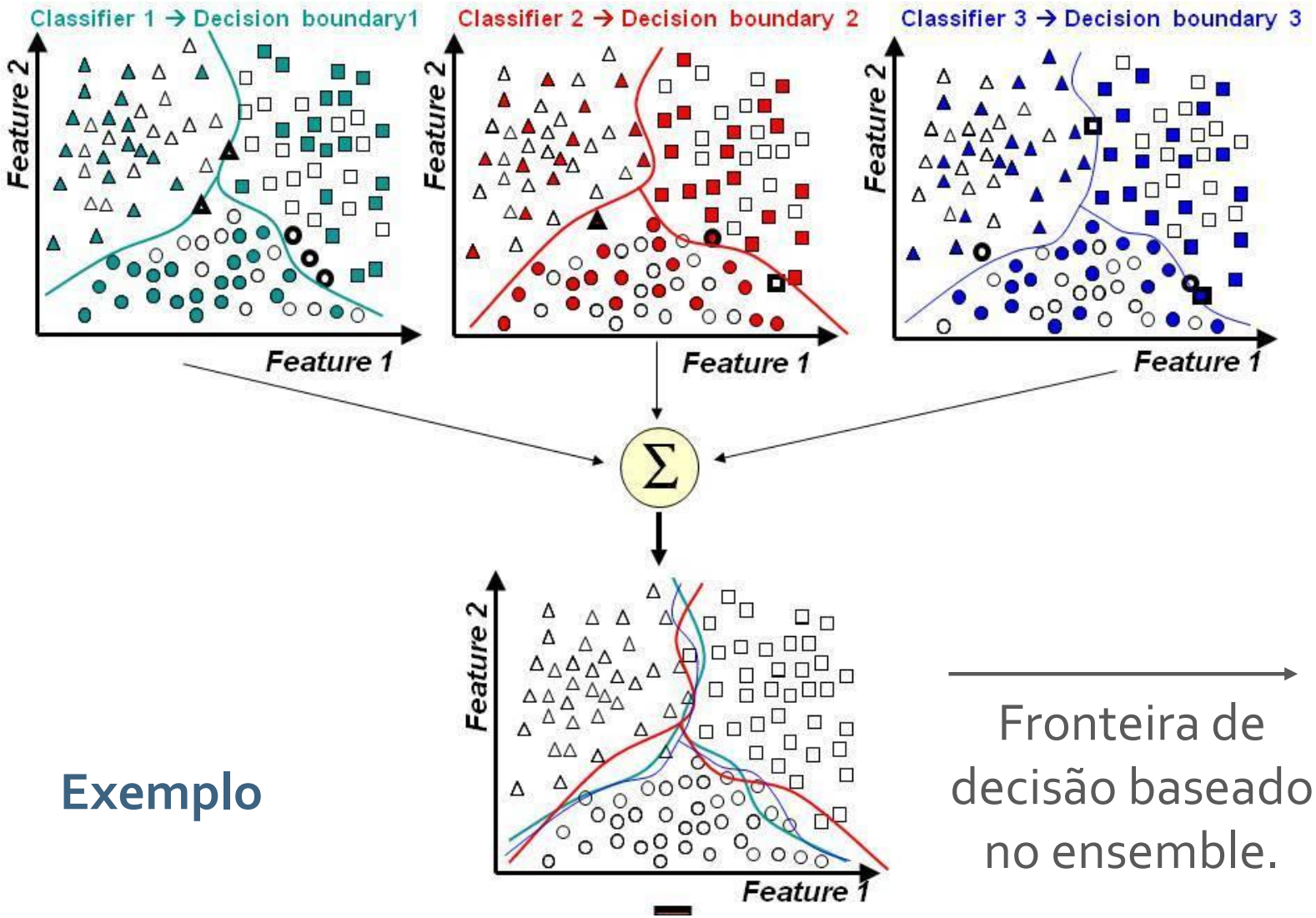
Ensemble Methods são algoritmos de aprendizado que constroem um conjunto de modelos e então preveem novas observações (i.e. novos dados) através da combinação (ponderada) das diversas previsões dada pelos modelos individuais.



https://vas3k.com/blog/machine_learning/

Ensemble methods

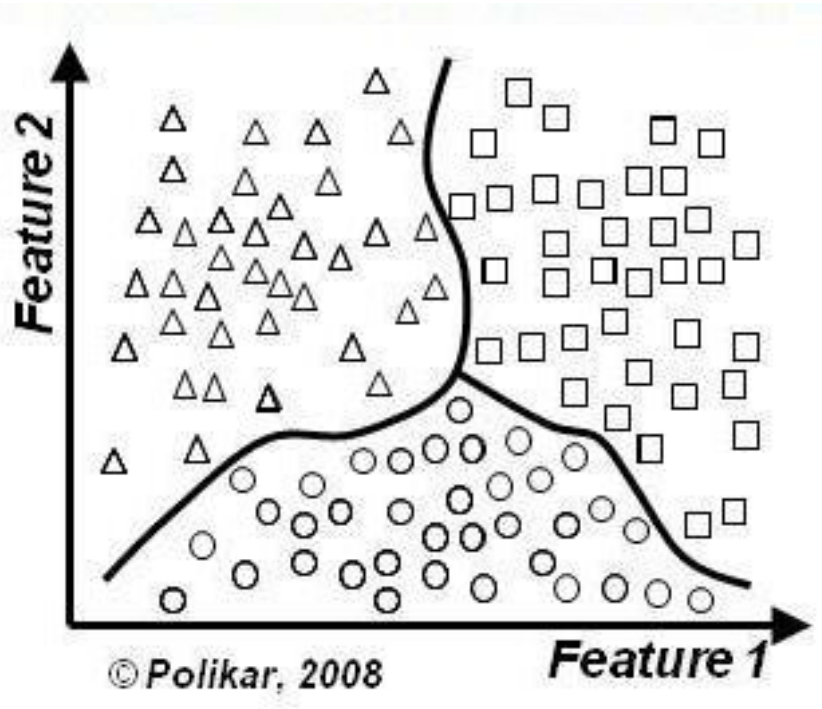
Overview



Exemplo

Fronteira de decisão baseado no ensemble.

Classificação Final



Ensemble methods

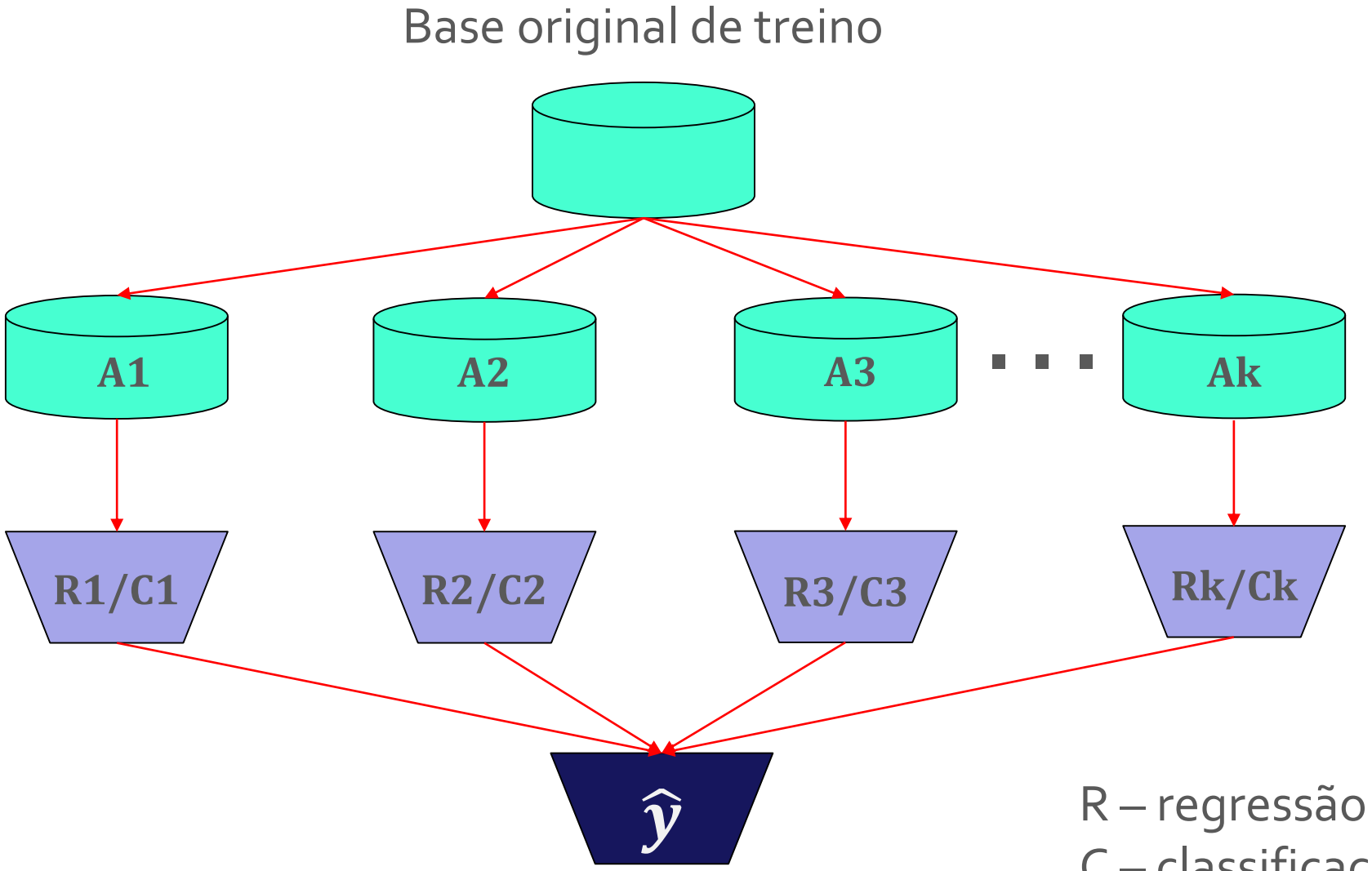
Estrutura geral

- De forma geral, temos:

PASSO 1:
Criação de múltiplas sub-amostras

PASSO 2:
Construção de múltiplos modelos

PASSO 3:
Combinação de previsões

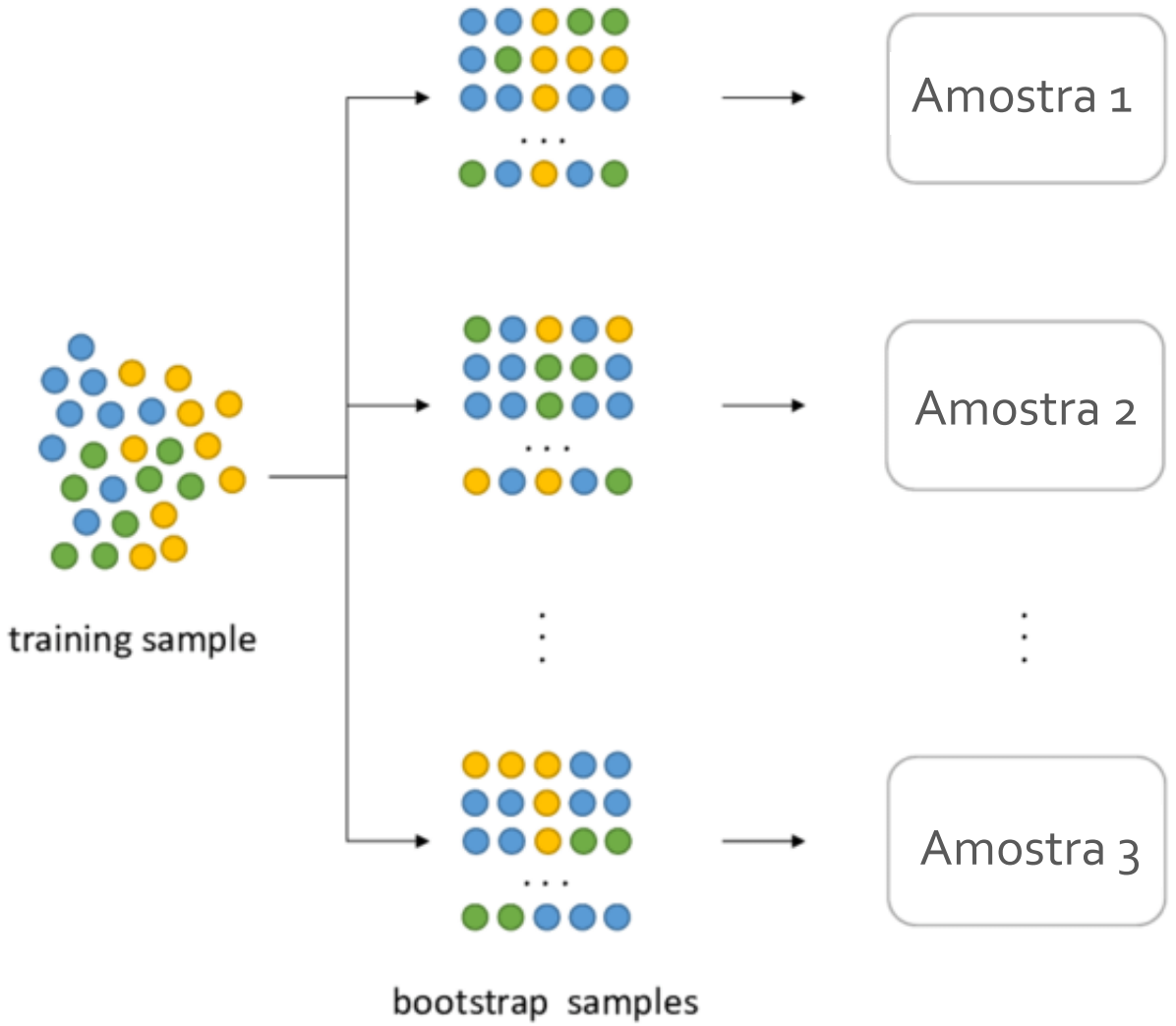


R – regressão
C – classificação

Ensemble methods

Bootstrapping

- Em linhas gerais, como a obtenção de inúmeras amostras independentes de uma população contendo o mesmo indivíduo é inviável, utilizamos um recurso denominado *bootstrapping*
- A técnica do *bootstrapping* consiste em selecionar sub-amostras, **com reposição**, de uma única amostra da população
- Em modelagem preditiva, mais especificamente na aplicação de *ensemble methods*, essas sub-amostras são utilizadas como amostra teste, ao passo que os casos que não são selecionados da amostra original (*out-of-bag*) para teste
- A amostra de *bootstrap* tem o mesmo tamanho da amostra original, sendo que no total cerca de 2/3 dos dados originais são representados nessas amostras ao passo que 1/3 são usados como *out-of-bag*



https://en.wikipedia.org/wiki/Bootstrap_aggregating

Random forest

Random forest

Overview

- O *random forest* é um método de *ensemble* que permite a construção de múltiplas árvores de regressão ou classificação formando uma "floresta" com árvores aleatória. Corresponde à um modificação do *bagging.*

Intuição

Variáveis fortes nos dados tendem se sobrepor à variáveis moderadamente mais fracas. Com isso, se fossem usadas todas as variáveis para o particionamento dos nós, as árvores seriam muito parecidas (correlacionadas)

- Para contornar o efeito acima, o *random forest* minimiza a correlação das árvores nas tarefas de regressão ou classificação dentro da "floresta" através da randomização da quebra dos atributos:
- Constroem-se diversos modelos de árvore em amostras *bootstrap*, mas a cada partição um subconjunto de m *features* é escolhida (dos *p* totais) para ser usada na quebra [default: $m = \sqrt{p}$]

Random forest

Overview

- O *random forest* é um método de *ensemble* que permite a construção de múltiplas árvores de regressão ou classificação formando uma "floresta" com árvores aleatória. Corresponde à um modificação do *bagging.*

Intuição

Variáveis fortes nos dados tendem se sobrepor à variáveis moderadamente mais fracas. Com isso, se fossem usadas todas as variáveis para o particionamento dos nós, as árvores seriam muito parecidas (correlacionadas)

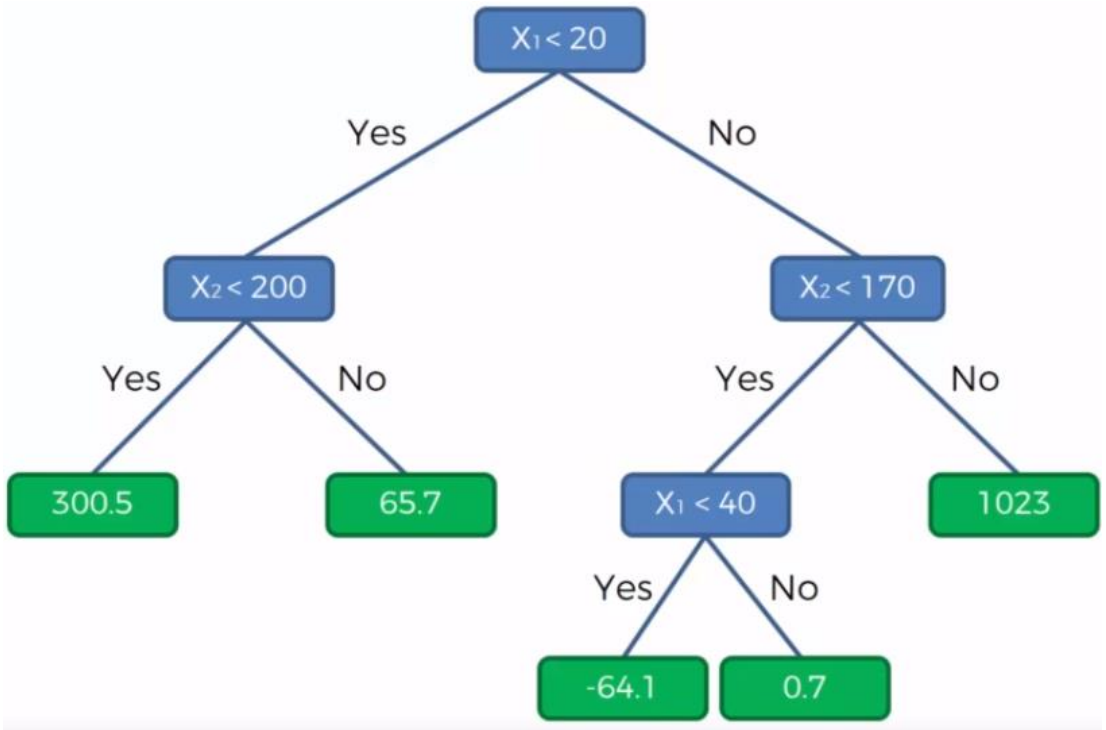
- Como as árvores não sofrem poda, elas podem apresentar internamente *overfitting*. Entretanto, sendo as sub-amostras distintas esses eventuais erros são desprezados na predição conjunta e não comprometem o resultado final
- Possuem parâmetros para *tuning*, sendo que os mais importantes são o número de árvores a serem construídas e o número de variáveis *m.* Na literatura encontramos diversas receitas, mas recomenda-se testar diversas configurações, focando sempre na melhor performance do modelo (AUC, RMSE, etc)

Random forest

Lembrando do particionamento da árvores

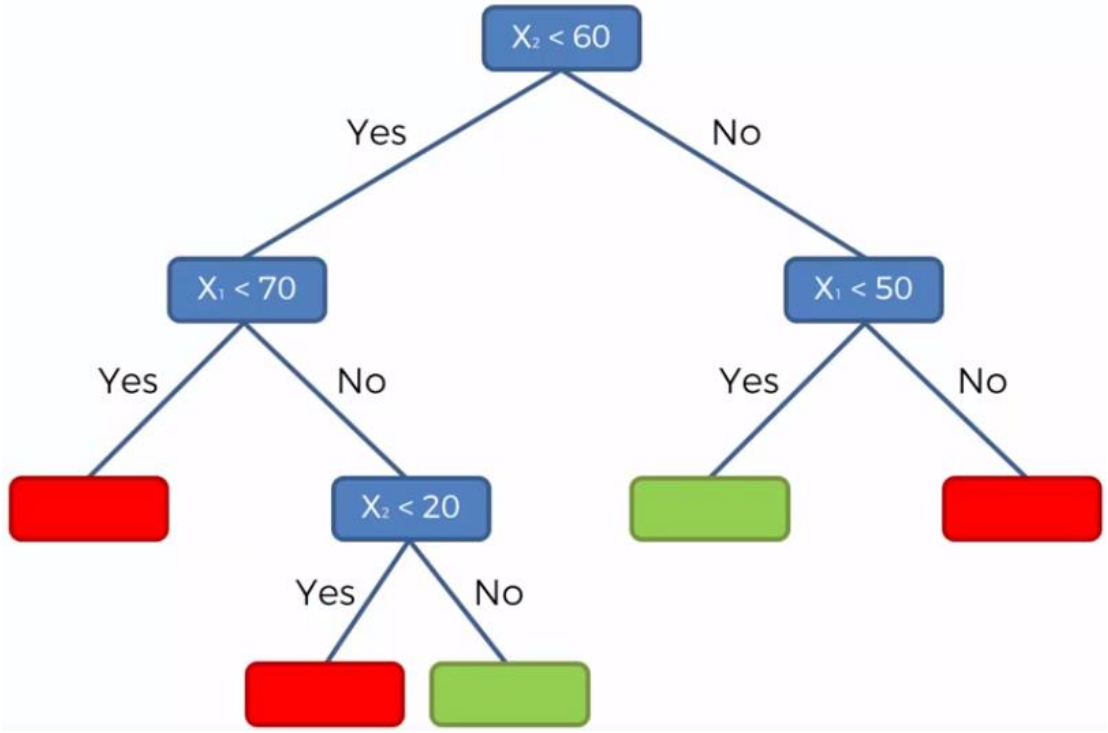
Regressão

O valor obtido nos nós finais é dado pela média das observações que estão contidas dentro daquele nó



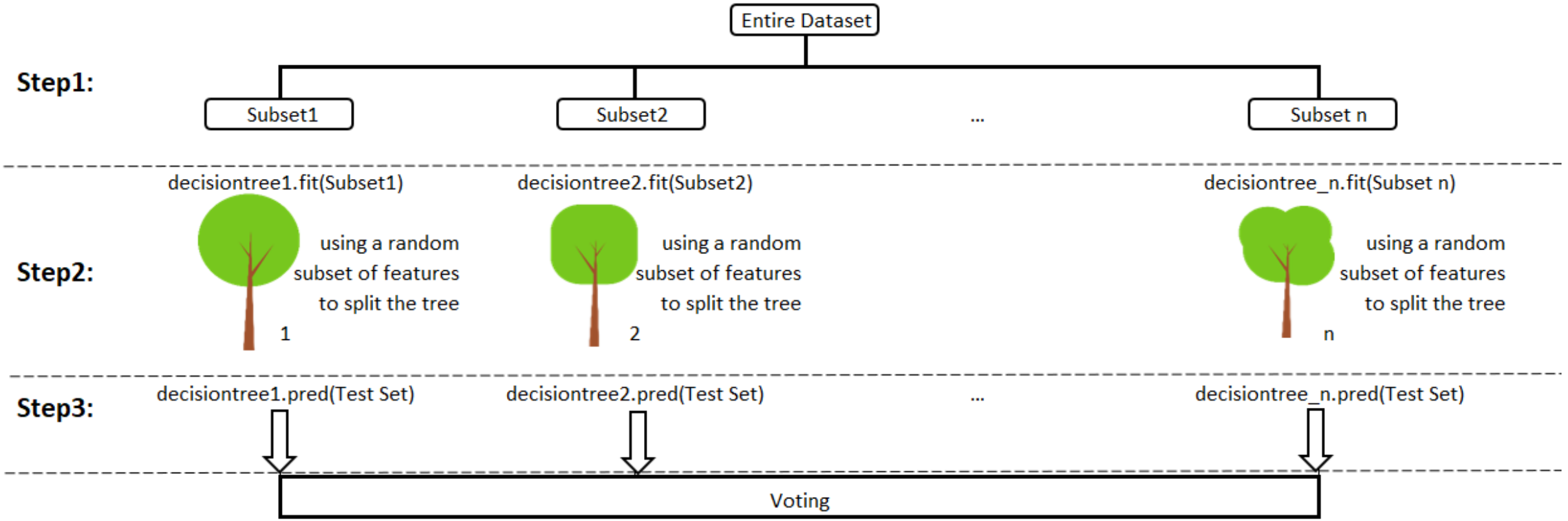
Classificação

O *label* nos nós finais é dado pela categoria dominante das observações dentro daquele nó



Random forest

Estrutura geral



Aprendizagem

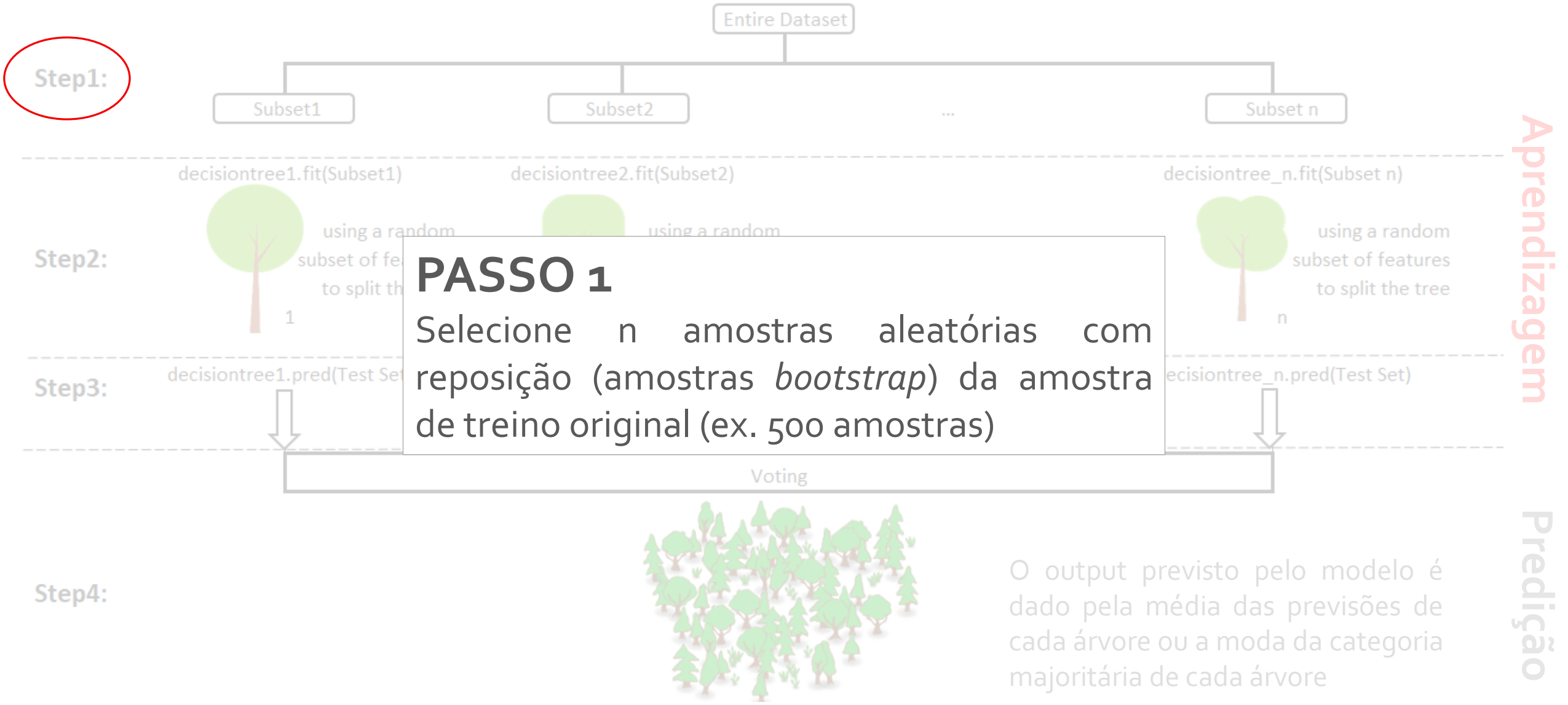
Predição

O output previsto pelo modelo é dado pela média das previsões de cada árvore ou a moda da categoria majoritária de cada árvore

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Random forest

Estrutura geral



PASSO 1

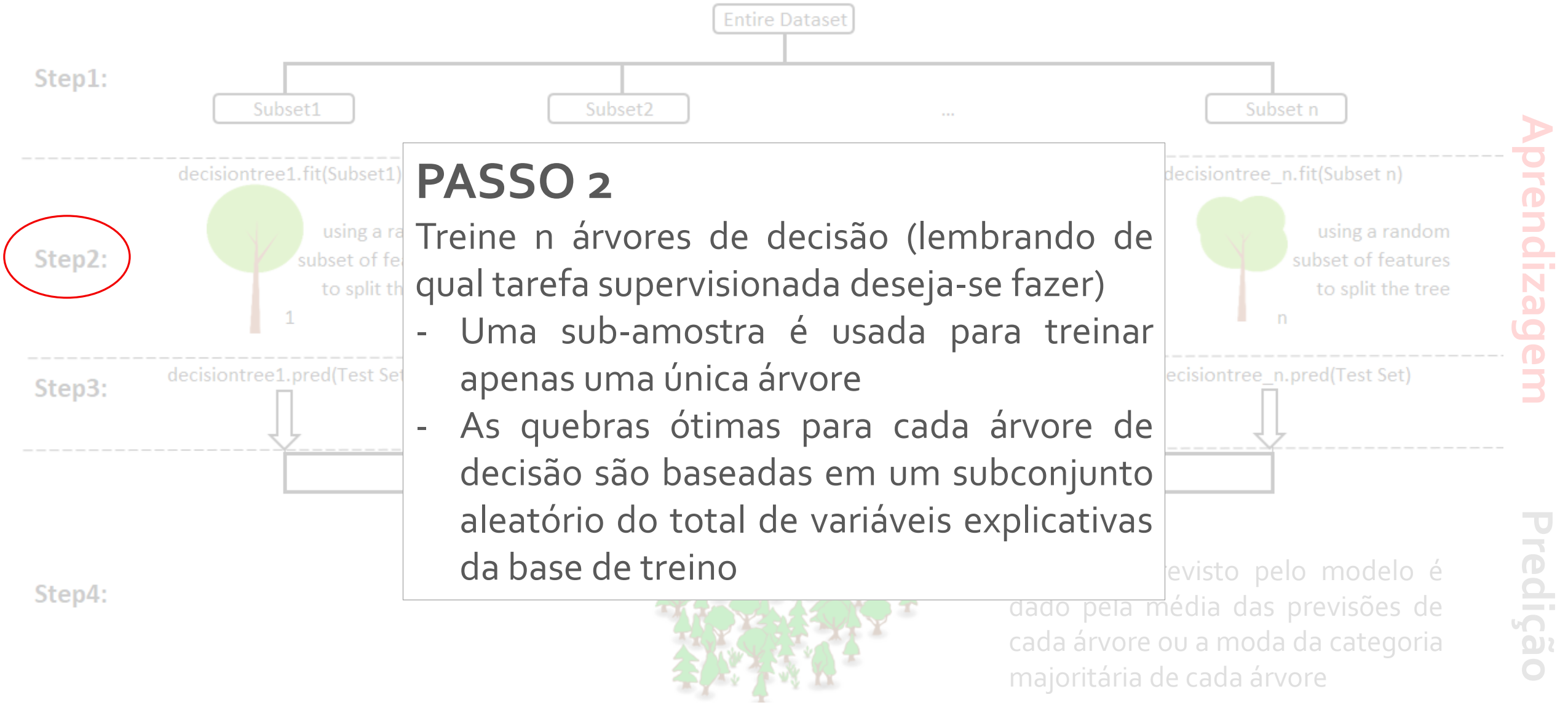
Selecione n amostras aleatórias com reposição (amostras *bootstrap*) da amostra de treino original (ex. 500 amostras)

O output previsto pelo modelo é dado pela média das previsões de cada árvore ou a moda da categoria majoritária de cada árvore

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Random forest

Estrutura geral



PASSO 2

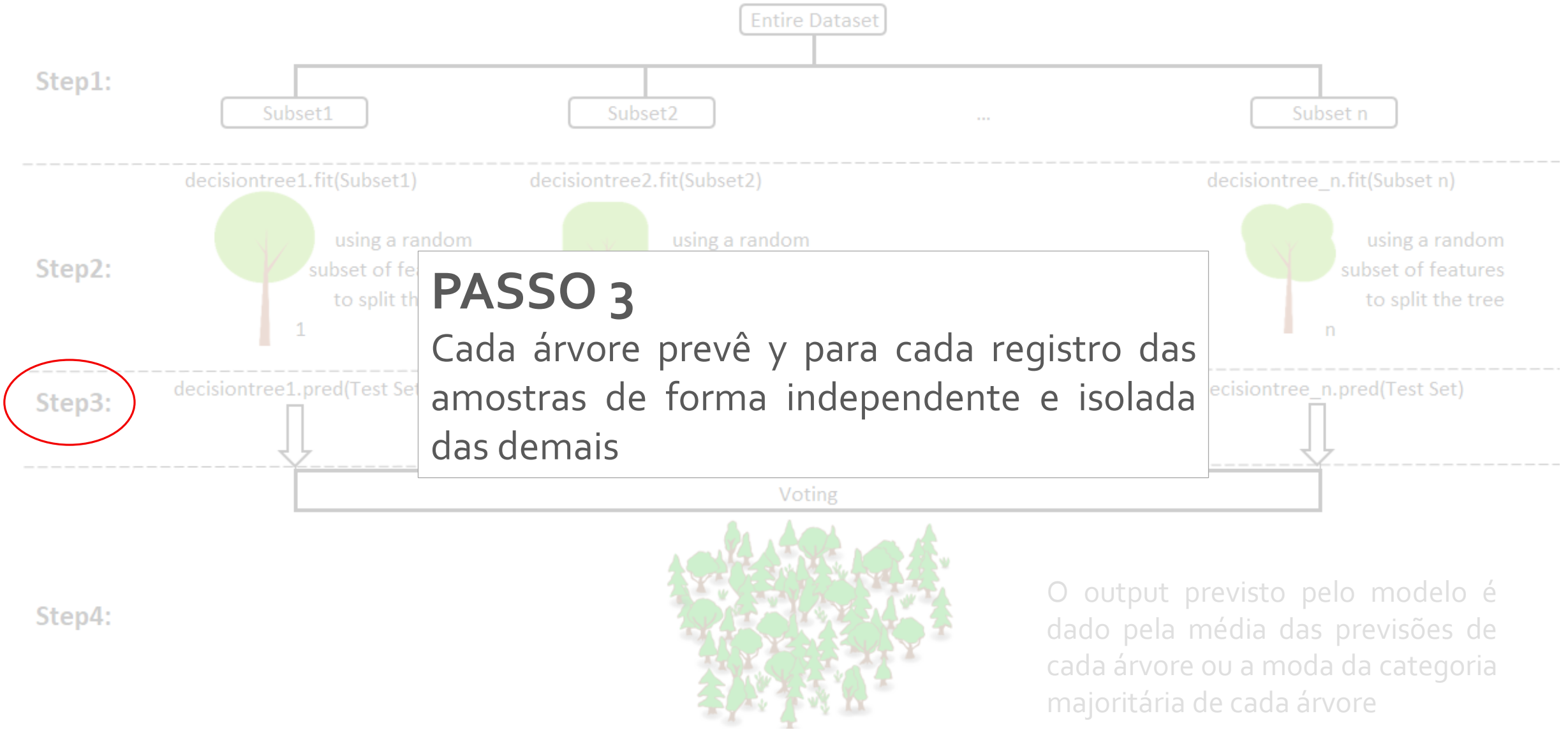
Treine n árvores de decisão (lembrando de qual tarefa supervisionada deseja-se fazer)

- Uma sub-amostra é usada para treinar apenas uma única árvore
- As quebras ótimas para cada árvore de decisão são baseadas em um subconjunto aleatório do total de variáveis explicativas da base de treino

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Random forest

Estrutura geral



PASSO 3

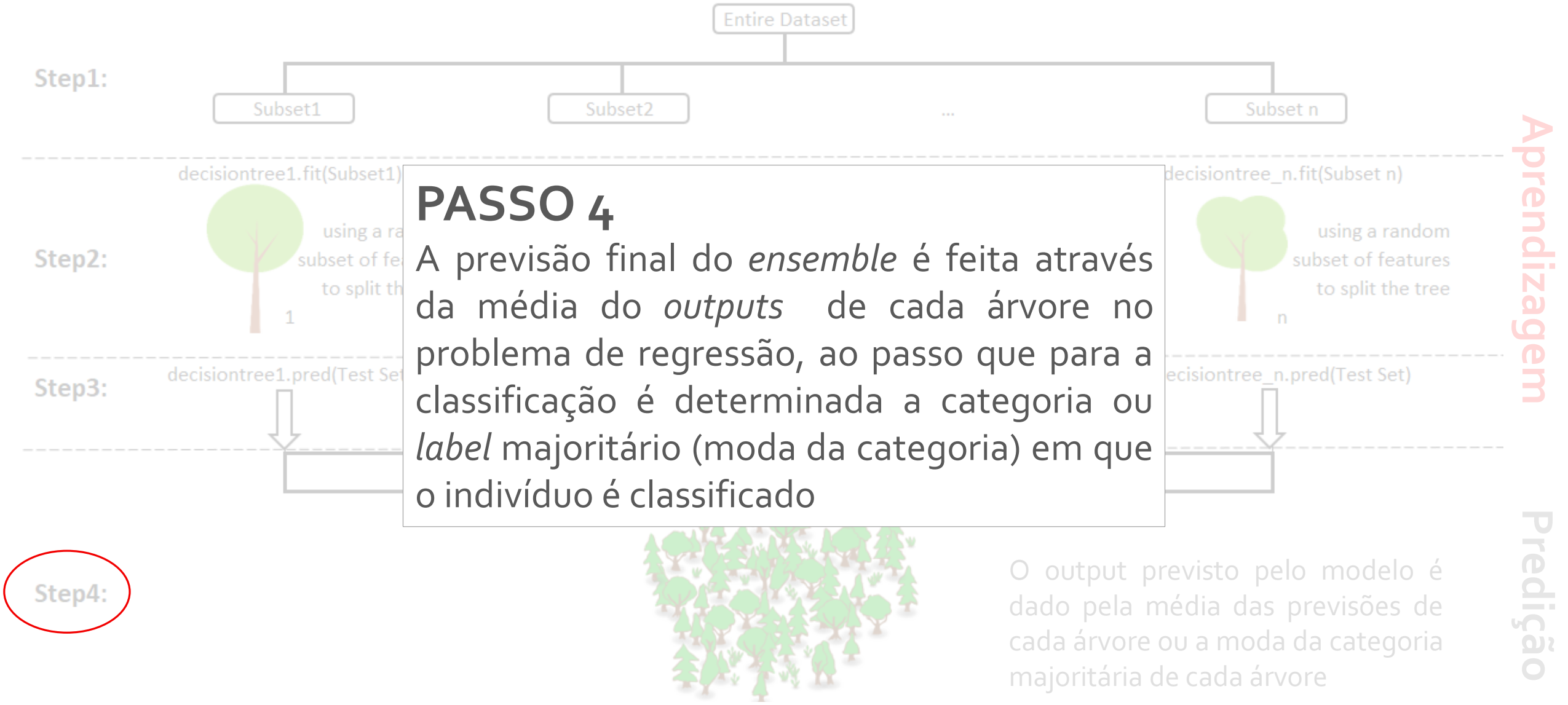
Cada árvore prevê y para cada registro das amostras de forma independente e isolada das demais

O output previsto pelo modelo é dado pela média das previsões de cada árvore ou a moda da categoria majoritária de cada árvore

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Random forest

Estrutura geral



PASSO 4

A previsão final do *ensemble* é feita através da média do *outputs* de cada árvore no problema de regressão, ao passo que para a classificação é determinada a categoria ou *label* majoritário (moda da categoria) em que o indivíduo é classificado

O output previsto pelo modelo é dado pela média das previsões de cada árvore ou a moda da categoria majoritária de cada árvore

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Random forest

Prós e contras do algoritmo

Prós

Pode apresentar resultados com boa acurácia

Consegue lidar bem com grande conjunto de dados com alta dimensionalidade

Fornece a importância das variáveis usadas na quebra dos nós

Como para a árvore, requer menos *feature engineering*

Baixo impacto de *outliers*

Consegue lidar com dados desbalanceados (um *label* é muito menos representativo)

Contras

Caso o dado seja ruidoso, pode apresentar *overfitting*

As variáveis utilizadas precisam ter poder preditivo senão o algoritmo não entrega bons resultados

Diferentemente de uma única árvore de classificação, os resultados são difíceis de se interpretar (modelo *black box*)

É necessário realizar o *tuning* dos parâmetros para uma melhor performance

Adaboost

Adaboost

Overview

- Como um método de *ensemble*, o *boosting* (“alavancagem”) combina a previsão de diversos modelos para formar a classificação final de cada indivíduo presente na amostra.
- Entretanto aqui, o *boosting* obtém modelos de previsão com boa acurácia a partir de regras moderadamente menos acuradas.

Intuição

O algoritmo gera uma sequencia de classificadores fracos, onde a cada iteração registros que são classificados erroneamente recebem mais peso. Dessa forma a cada iteração o algoritmo aprende mais, focando-se em casos mais difíceis.

- No *boosting*, as amostras não são selecionadas de forma independente e a classificação final não é por voto majoritária pois cada rodada do *boosting* depende dos resultados das rodadas anteriores. As árvores são desenvolvidas sequencialmente: cada árvore é criada usando informação da árvore do passo anterior

Adaboost

Overview

- Como um método de *ensemble*, o *boosting* (“alavancagem”) combina a previsão de diversos modelos para formar a classificação final de cada indivíduo presente na amostra.
- Entretanto aqui, o *boosting* obtém modelos de previsão com boa acurácia a partir de regras moderadamente menos acuradas.

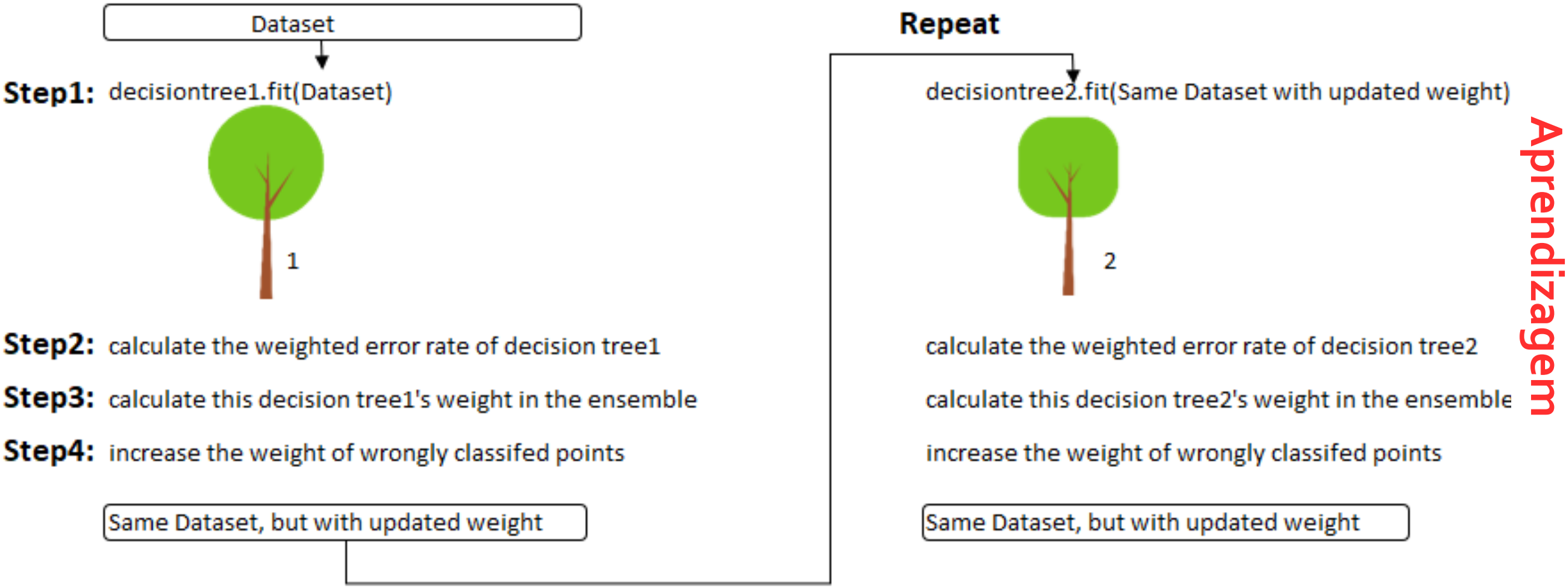
Intuição

O algoritmo gera uma sequencia de classificadores fracos, onde a cada iteração registros que são classificados erroneamente recebem mais peso. Dessa forma a cada iteração o algoritmo aprende mais, focando-se em casos mais difíceis.

- Existem diversos algoritmos diferentes para o *boosting* como *Gradient Boosting*, *Xtreme Gradient Boosting*, *Lightboost*, *Catboost* . Aqui focamos no *Adaboost* (*Adaptive Boosting*), que foi o pioneiro na década de 90.
- O desenvolvimento de modelos do tipo *boosting* requer o ajuste de diversos parâmetros (*tuning*) e por isso tendem a ser mais complicado de refinar a escolha dos parâmetros

Adaboost

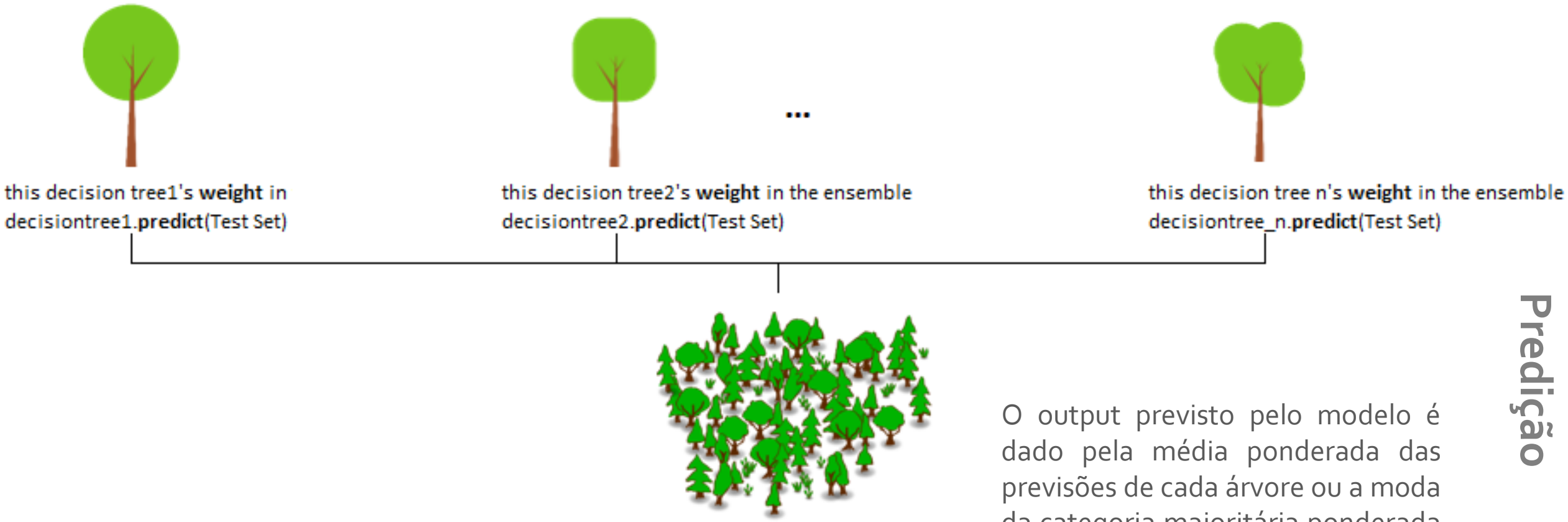
Estrutura geral



https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Adaboost

Estrutura geral



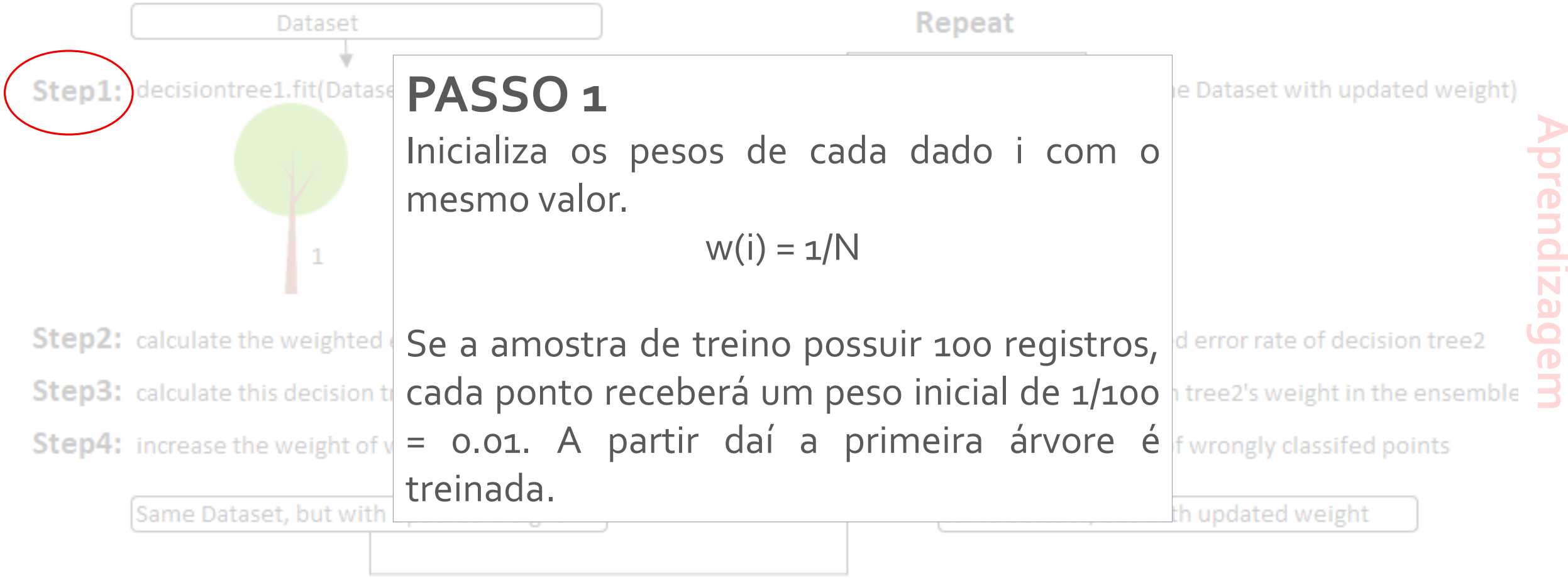
O output previsto pelo modelo é dado pela média ponderada das previsões de cada árvore ou a moda da categoria majoritária ponderada de cada árvore

Predição

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Adaboost

Estrutura geral



PASSO 1

Inicializa os pesos de cada dado i com o mesmo valor.

$$w(i) = 1/N$$

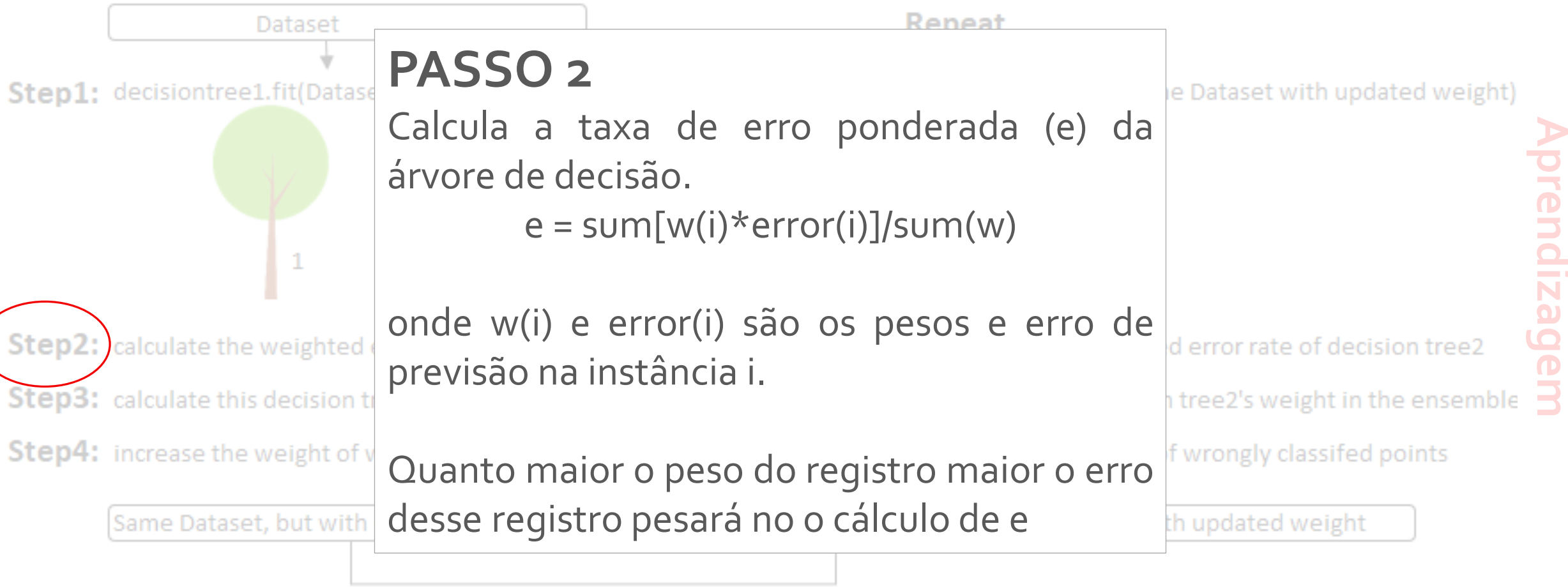
Se a amostra de treino possuir 100 registros, cada ponto receberá um peso inicial de 1/100 = 0.01. A partir daí a primeira árvore é treinada.

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Aprendizagem

Adaboost

Estrutura geral



PASSO 2

Calcula a taxa de erro ponderada (e) da árvore de decisão.

$$e = sum[w(i)*error(i)]/sum(w)$$

onde w(i) e error(i) são os pesos e erro de previsão na instância i.

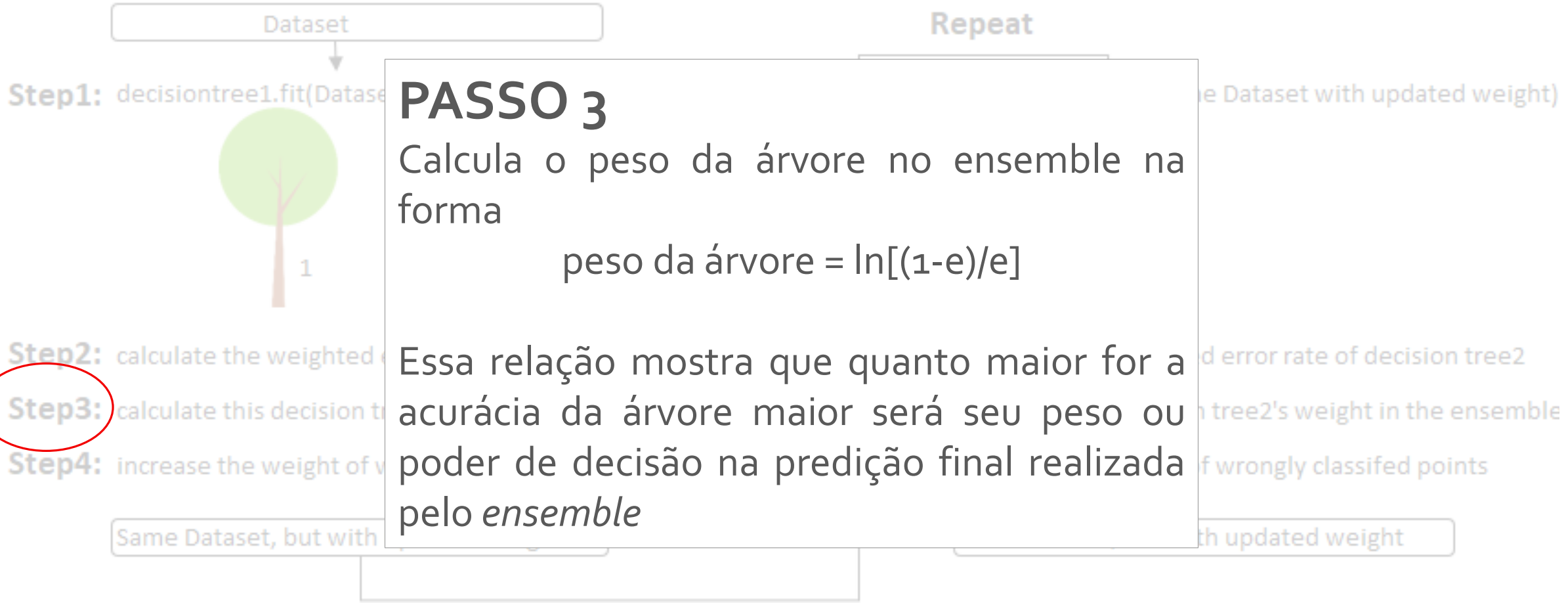
Quanto maior o peso do registro maior o erro desse registro pesará no o cálculo de e

Aprendizagem

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Adaboost

Estrutura geral



PASSO 3

Calcula o peso da árvore no ensemble na forma

$$\text{peso da árvore} = \ln[(1-e)/e]$$

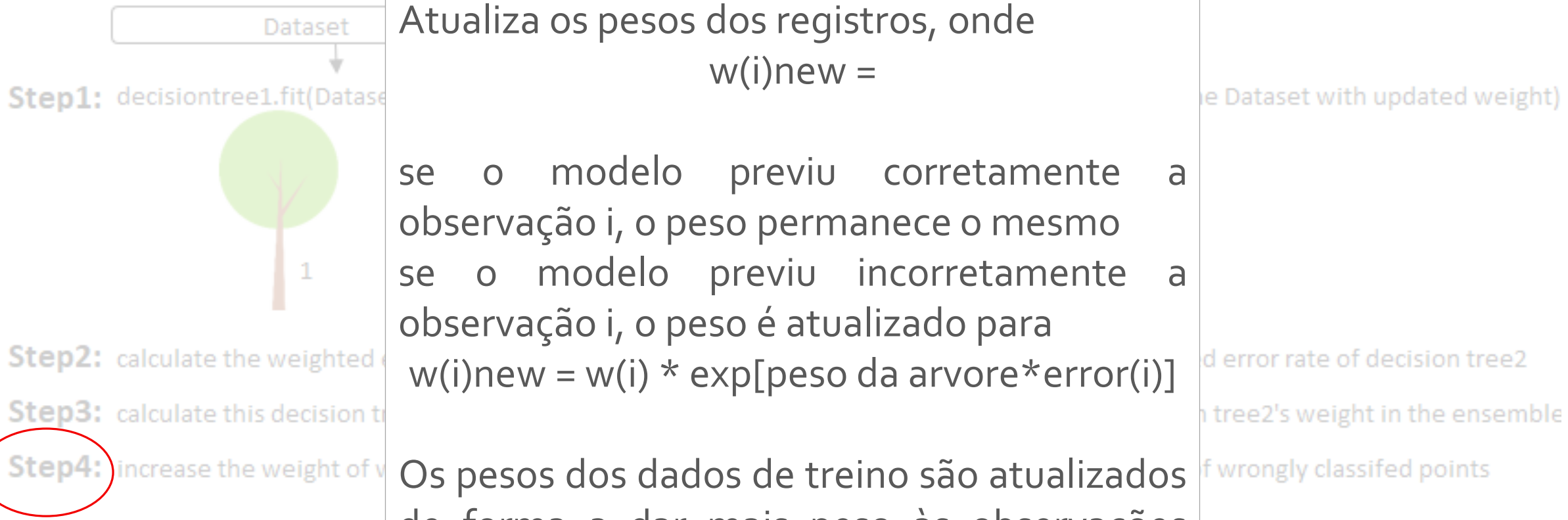
Essa relação mostra que quanto maior for a acurácia da árvore maior será seu peso ou poder de decisão na predição final realizada pelo *ensemble*

Aprendizagem

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Adaboost

Estrutura geral



PASSO 4

Atualiza os pesos dos registros, onde

w(i)new =

se o modelo previu corretamente a observação i, o peso permanece o mesmo

se o modelo previu incorretamente a observação i, o peso é atualizado para

w(i)new = w(i) * exp[peso da arvore*error(i)]

Os pesos dos dados de treino são atualizados de forma a dar mais peso às observações incorretamente previstas. Nas implementação esses pesos permitem selecionar mais vezes os casos mais difíceis

Aprendizagem

Adaboost

Estrutura geral

this decision tree1's weight in
decisiontree1.predict(Test Set)

his decision tree n's weight in the ensemble
ecisiontree_n.predict(Test Set)

PASSO 5

Repetir o processo de construção das árvores, atualização dos pesos e recalculo da ponderação até atingir um número pré-determinado de iterações

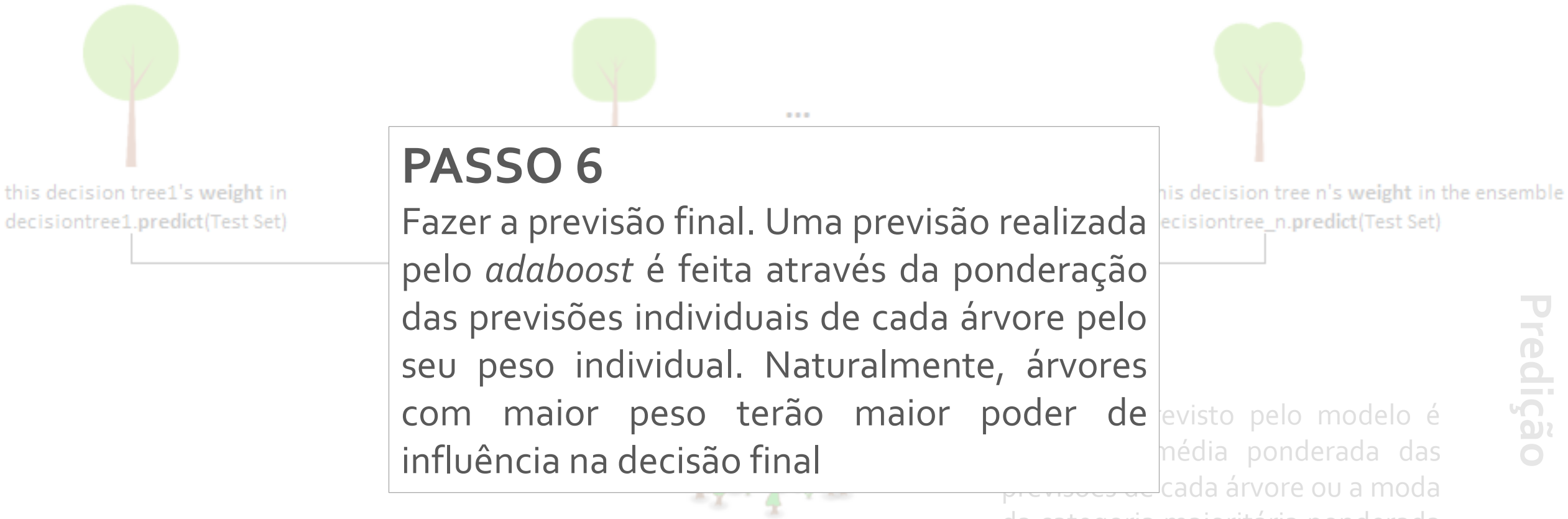
evisto pelo modelo é
dado pela média ponderada das
previsões de cada árvore ou a moda
da categoria majoritária ponderada
de cada árvore

Predição

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Adaboost

Estrutura geral



PASSO 6

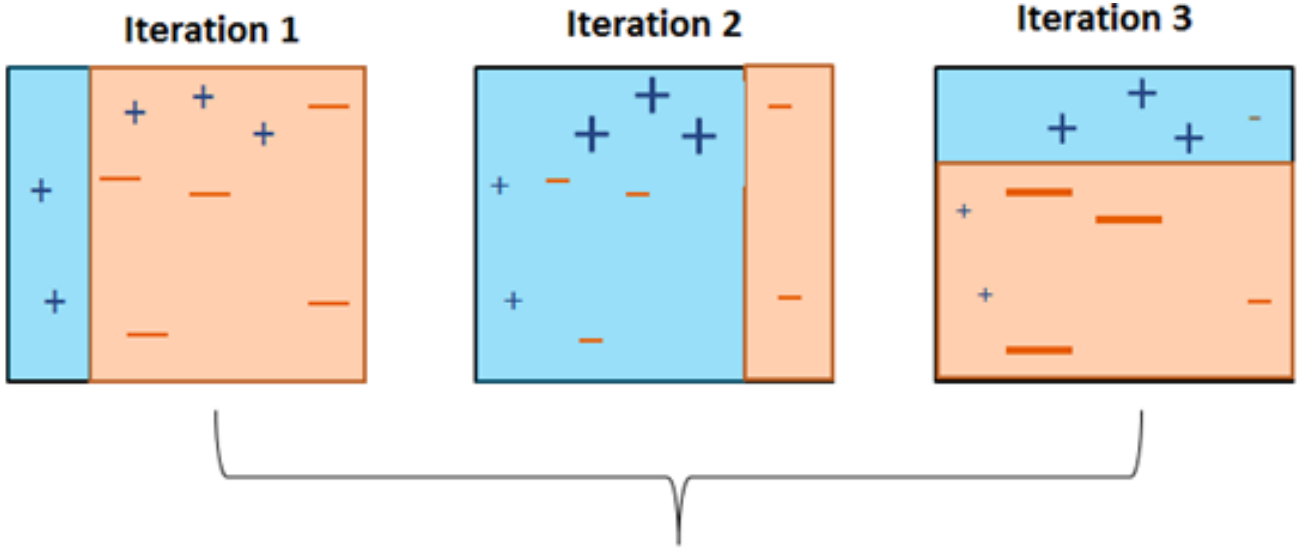
Fazer a previsão final. Uma previsão realizada pelo *adaboost* é feita através da ponderação das previsões individuais de cada árvore pelo seu peso individual. Naturalmente, árvores com maior peso terão maior poder de influência na decisão final

https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

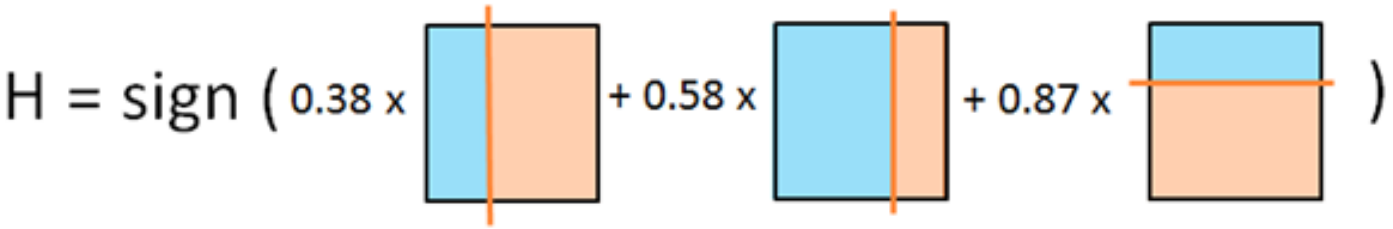
Adaboost

Exemplo ilustrativo

Os elementos incorretamente classificados ganham mais atenção na próxima iteração para o desenvolvimento da árvore



A previsão final é feita com a combinação das previsões individuais de cada árvore porém ponderadas pelo peso de cada árvore



Exemplo de classificação feita com o *adaptive boosting* utilizando árvores de classificações simples (*weak learners*)

https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781789953633/2/ch02lvl1sec22/adaboost-classifier

Adaboost

Prós e contras do algoritmo

Prós

Pode apresentar resultados com boa acurácia

Diferentemente do *random forest* e do *bagging*, o *adaboost* considera os pesos de cada árvore.

Constrói as árvores sequencialmente aprendendo com os erros das iterações anteriores

Diferente algoritmos podem ser usados como *weak learners*

Pode lidar tanto com variáveis qualitativas quando quantitativas

Contras

Caso o dado seja ruidoso, pode apresentar *overfitting*

Pode apresentar *overfitting* se forem selecionados *learners* menos complexos ou se o número de iterações é muito grande

Pode ter um desempenho computacional ruim quando comparado à outros tipos de *boosting* como o *XGBoosting*

Precisa de ajuste fino dos parâmetros

Não possui implementação para regressão no R

Prática no RStudio

Prof. Dr. João Rafael Dias Pinto

...foco de hoje

- **Treinando os algoritmos de *random forest* e *adaboost***

  Criando as amostras de treino e teste. Ajuste dos algoritmos, aprimoramento dos resultados, seleção do número de árvores e iterações. Avaliações dos *outputs* dos modelos