Técnicas de *Ensembles*

Hélio Pio

Programação das Aulas

Tópico 1: Introdução a Inteligência Artificial
Tópico 2: Agentes Inteligentes
Tópico 3: Fundamentos de Aprendizagem de Máquina
Tópico 4: Redes Neurais Artificiais
Tópico 5: Atividade em Aula – Primeira Avaliação
Tópico 6: Representação da Incerteza e Lógica Fuzzy
Tópico 7: Redes Bayesianas
Tópico 8: Support Vector Machines
Tópico 9: Atividade em Aula – Segunda Avaliação
Tópico 10: Resolução de Problemas por Meio de Busca e Otimização
Tópico 11: Técnicas de Ensemble
Tópico 12: Atividade em Aula – Terceira Avaliação

O que são técnicas de *Ensemble*?

Técnicas de *Ensemble*

O que são técnicas de *Ensemble*?

- Conjunto de técnicas que permite a construção de modelos mais robustos a partir da combinação de modelos básicos. Na maioria das vezes, esses modelos básicos apresentam um desempenho não tão bom por si próprios, porque eles têm um alto viés (modelos de baixo grau de liberdade, por exemplo) ou porque eles têm muita variância para serem robustos (modelos de alto grau de liberdade, por exemplo)

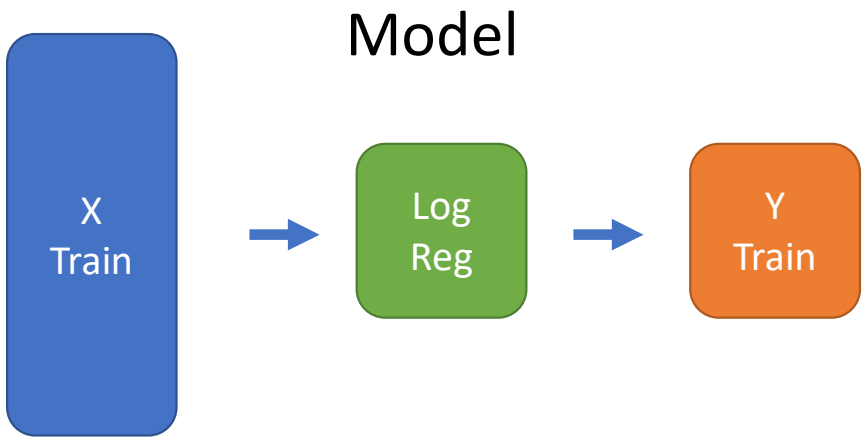
- Então, a ideia dos *ensembles* é tentar reduzir o viés e / ou a variância de aprendizes fracos, combinando vários deles para criar um aprendiz forte (ou modelo de *ensemble*) que obtenha melhores desempenhos.

Técnicas de *Ensemble*

O que são *Ensemble*?

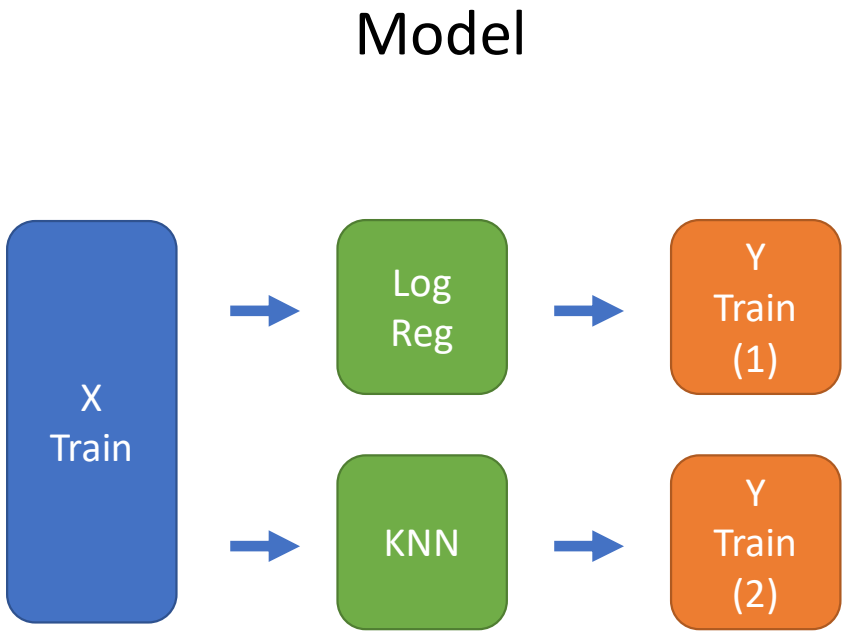
Técnicas de *Ensemble*

O que são *Ensemble*?



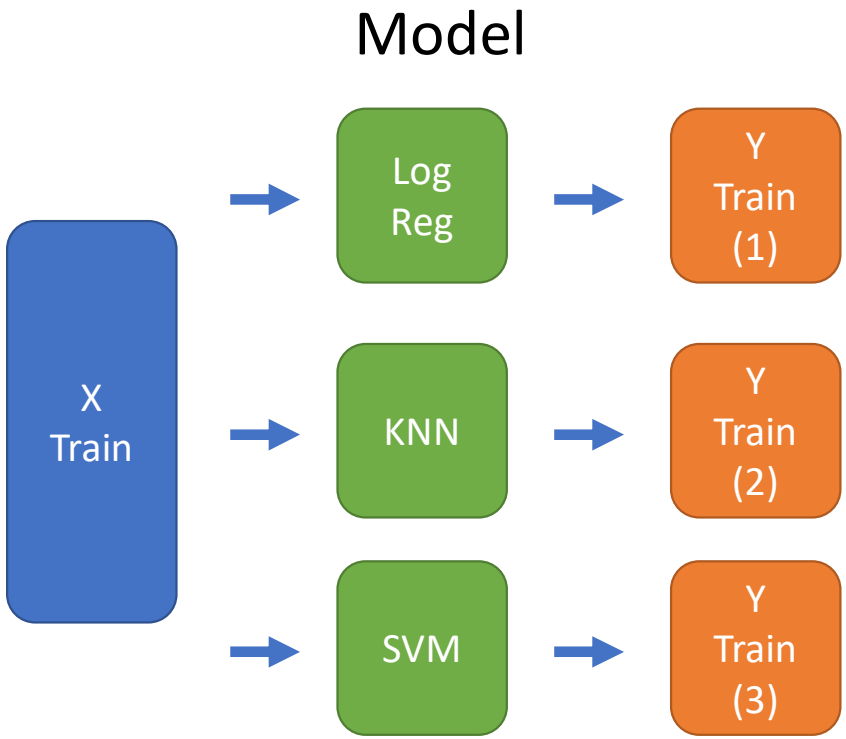
Técnicas de *Ensemble*

O que são *Ensemble*?



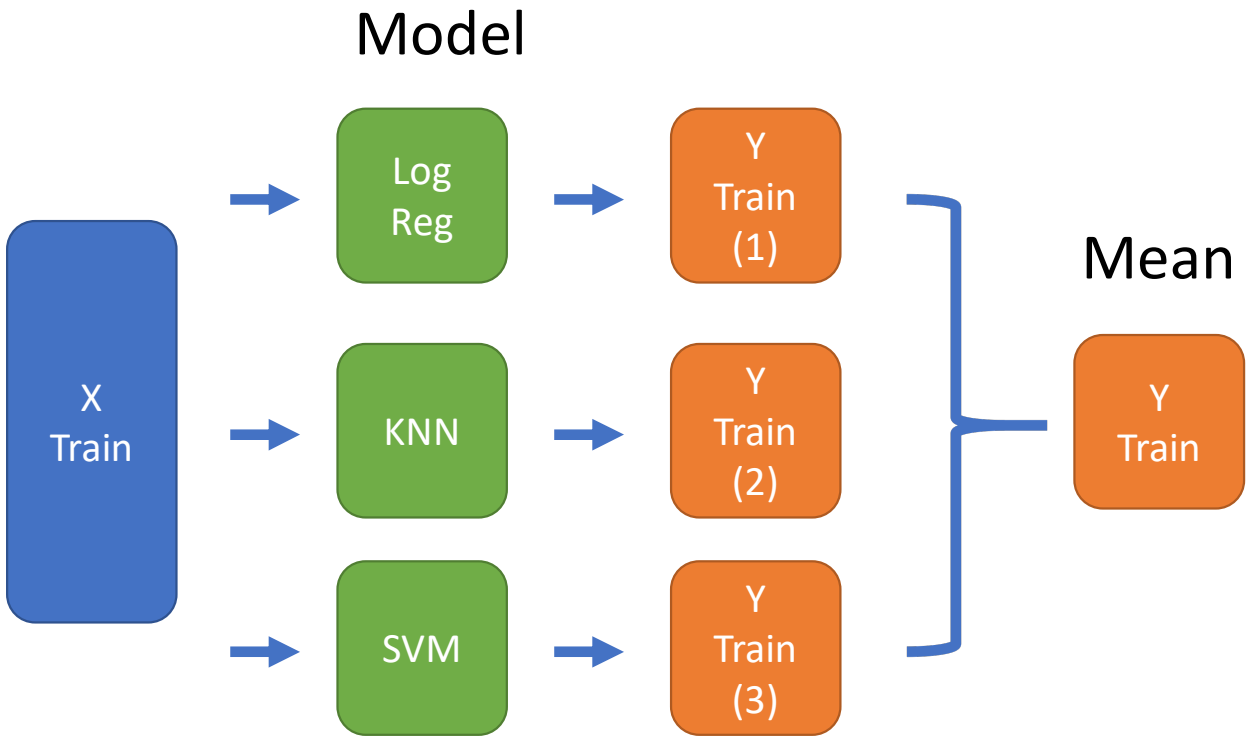
Técnicas de *Ensemble*

O que são *Ensemble*?



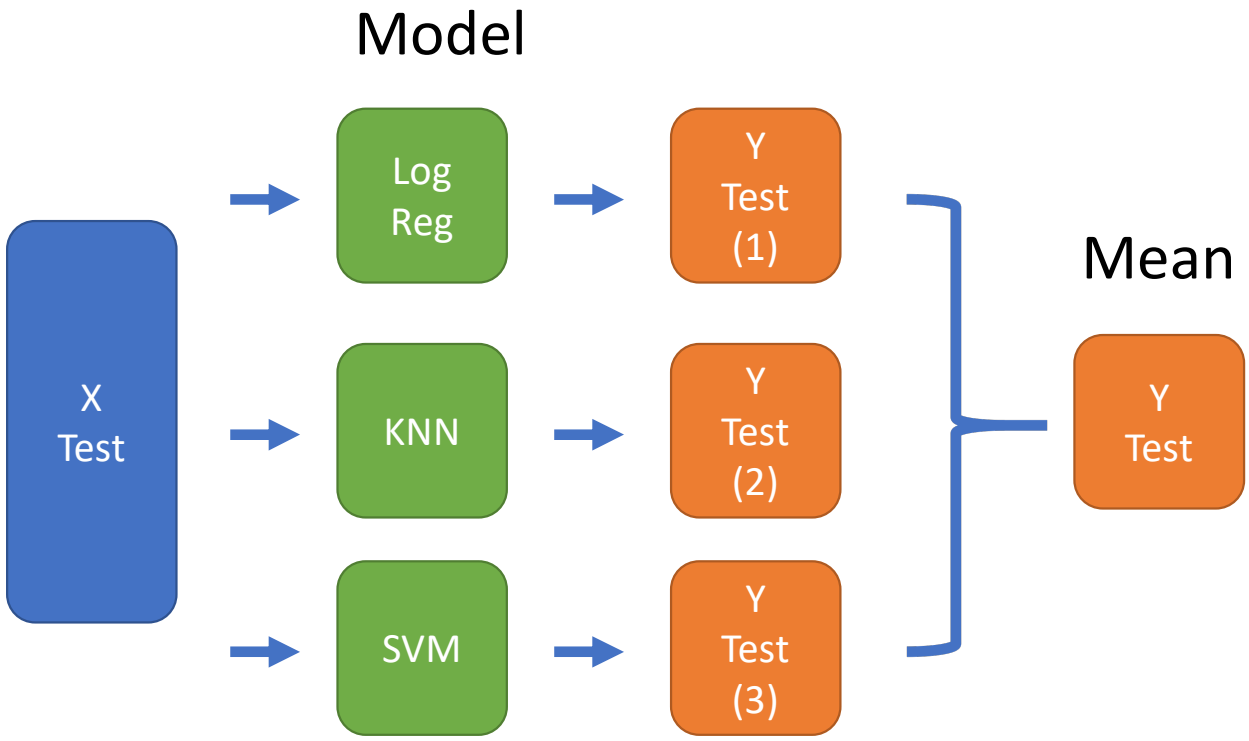
Técnicas de *Ensemble*

O que são *Ensemble*?



Técnicas de *Ensemble*

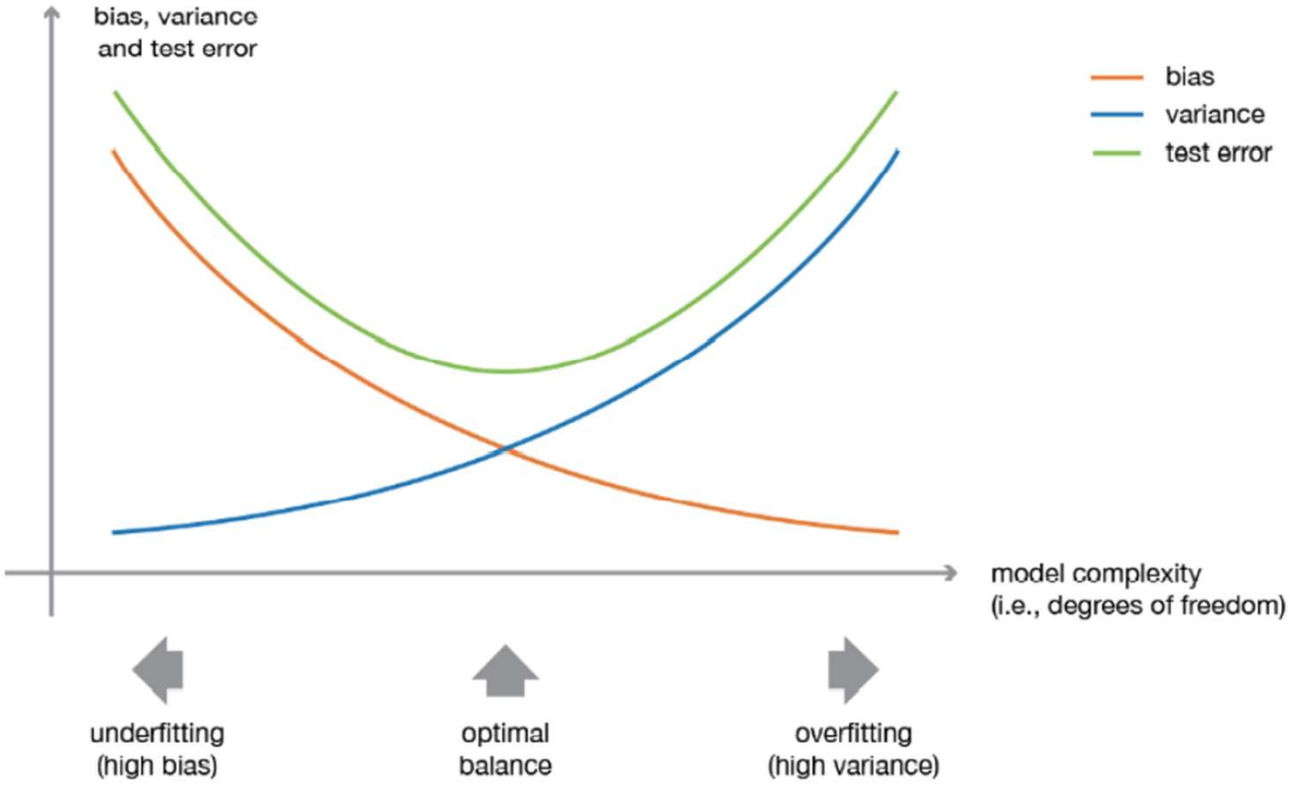
O que são *Ensemble*?



Técnicas de *Ensemble*

O que são *Ensemble*?

- Qual o erro para um classificador binário?
- Precisa ser menor que 50%.



Técnicas de *Ensemble*

Principais técnicas para combinação de modelos

- *Bagging* - Considera *weak learners* homogêneos, aprendendo independentemente um do outro em paralelo, seguindo algum tipo de processo determinístico de média para os combinar. Ex. *Random Forest* e *Extra Trees*.

- *Boosting*- Considera *weak learners* homogêneos, aprendendo sequencialmente de uma forma muito adaptativa (um modelo básico depende dos anteriores) e os combina seguindo uma estratégia determinística. Ex. *AdaBoost* (*Adaptive Boosting*), *Gradient Tree Boosting, XGBoost*.

- *Stacking*, Considera *weak learners* heterogêneos, *learn* em paralelo e os combina através do treinamento de um meta-modelo que tem como saída a predição baseada nos diferentes *weak learners*.

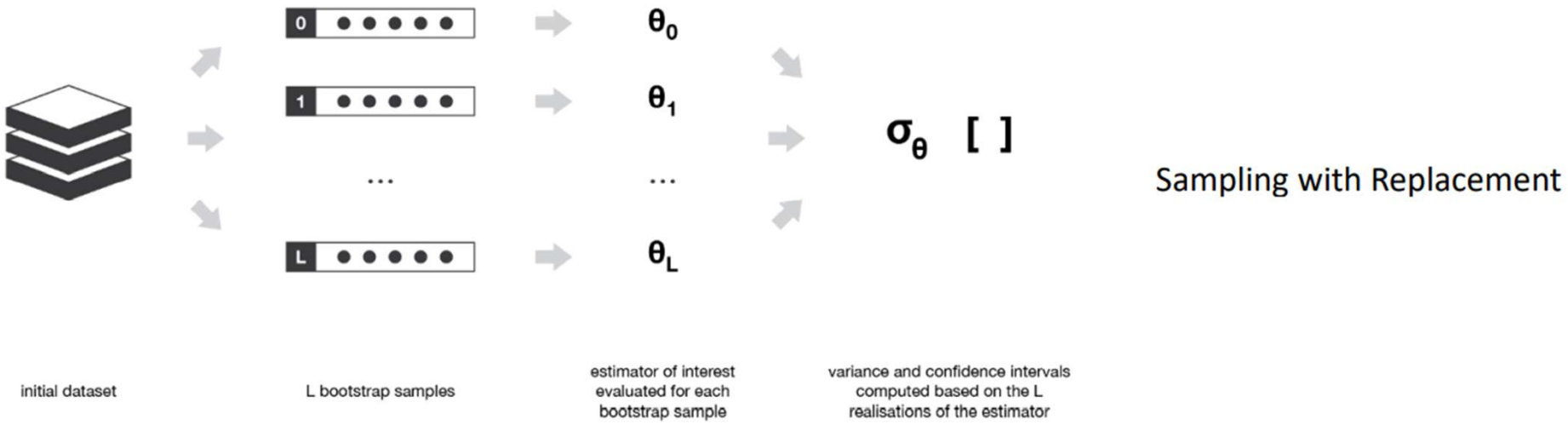
Técnicas de *Ensemble*

Bagging: Bootstrap – Aggregation

- *Bootstrapping* – é qualquer teste ou métrica que usa amostragem aleatória com substituição
    - Muito útil para avaliar a variância ou intervalos de confiança de estimadores estatísticos.
    - Representatividade - O tamanho do *dataset* inicial (N) deve ser grande o suficiente para capturar a complexidade da distribuição que ele representa.
    - Independência - O tamanho do N deve ser grande o suficiente quando comparado com o tamanho das amostras do *bootstrap* de forma que as amostras não sejam muito correlatas.

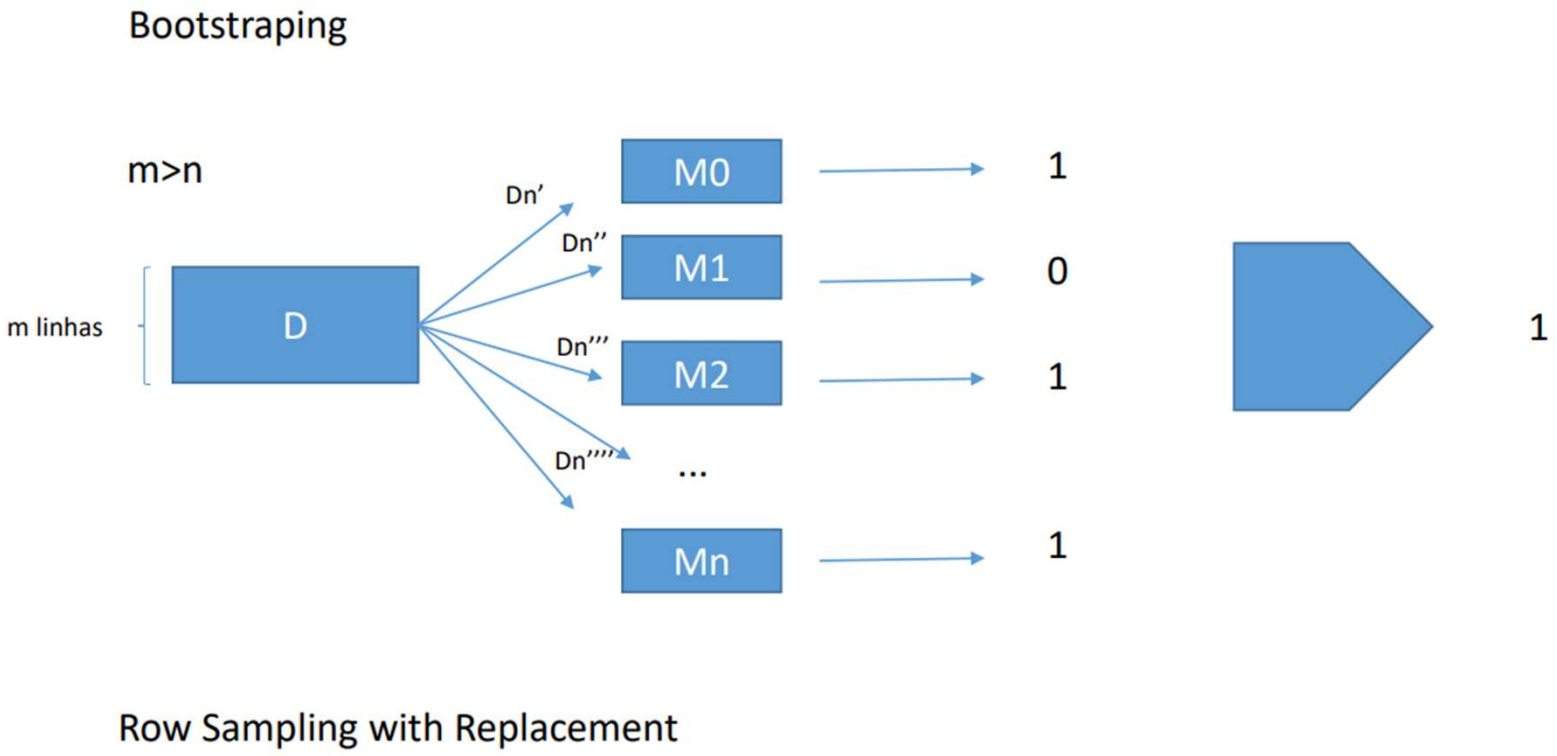
Técnicas de *Ensemble*

Bagging: Bootstrap – Aggregation



Técnicas de *Ensemble*

Bagging: Bootstrap – Aggregation



- Eleição por maioria - Classificação
- Ao agregar os modelos, o conjunto traz um resultado
- com menos variância que seus componentes.

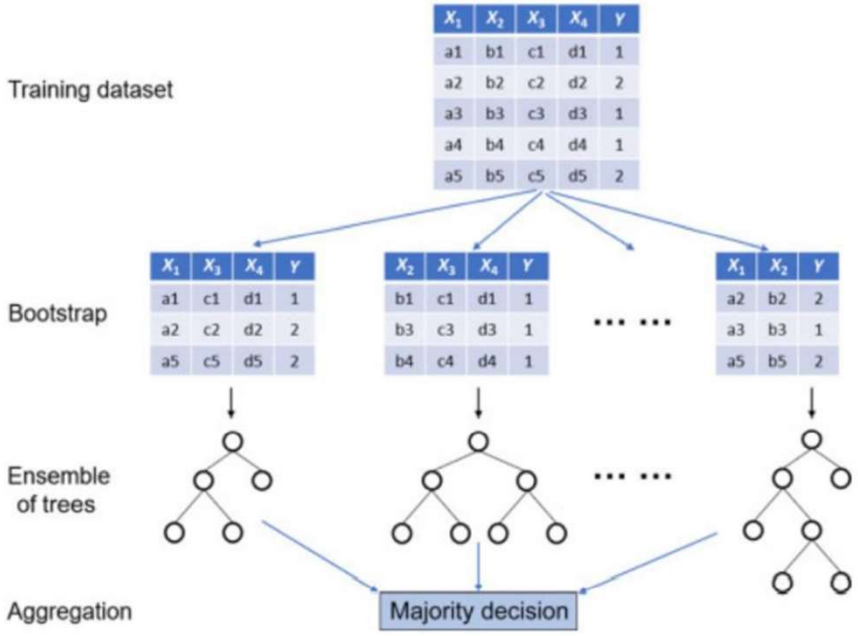
- E se fosse um problema de regressão?

Row Sampling with Replacement

Técnicas de *Ensemble*

Bagging: Bootstrap – Aggregation

- *Random Forest*
  - RF são compostas por várias árvores de decisão independentes que são treinadas independentemente em um subconjunto aleatório de Dados
  - *Row and Feature Sampling with Replacement*
  - Árvores de Decisão tem bias reduzido mas alta variância, tendência a *overfitting*.
  - Ao agregar os modelos, o conjunto traz um resultado com menos variância que seus componentes.

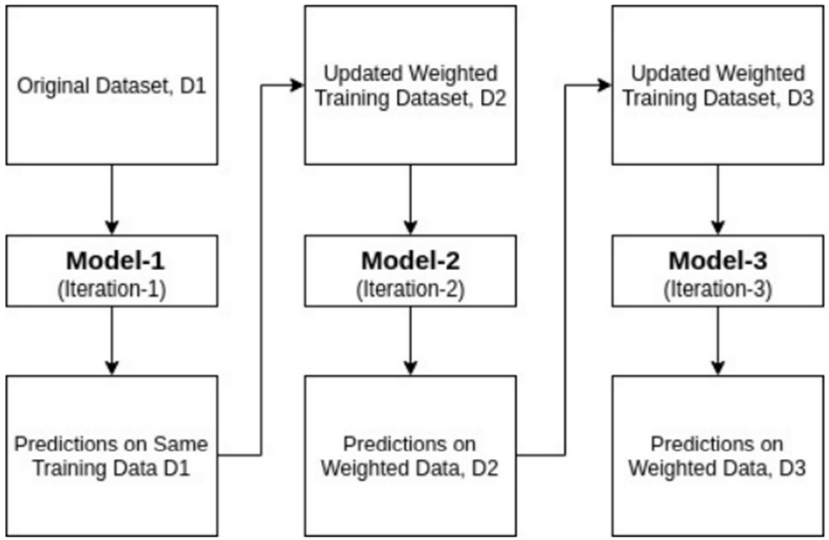


Técnicas de *Ensemble*

Boosting

- Combinação sequencial - A ideia é ajustar modelos iterativamente de forma que o treinamento do modelo em uma determinada etapa dependa dos modelos ajustados nas etapas anteriores.

- Cada modelo na sequência é ajustado a dar mais importância às observações no conjunto de dados que foram mal avaliados pelos modelos anteriores.

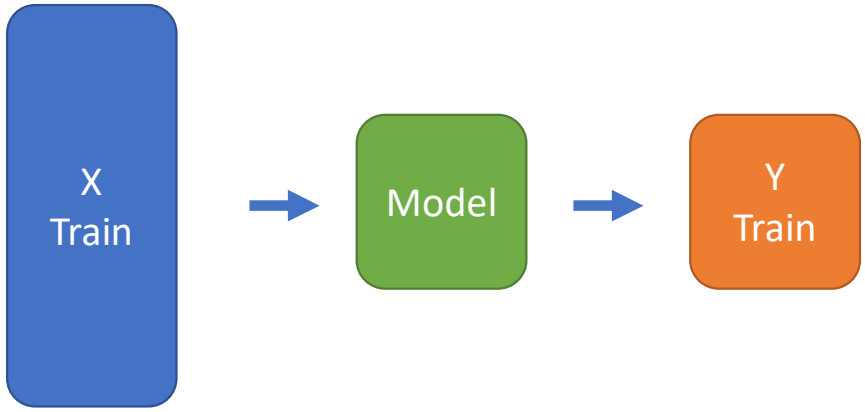


Técnicas de *Ensemble*

O que são *Ensemble*?

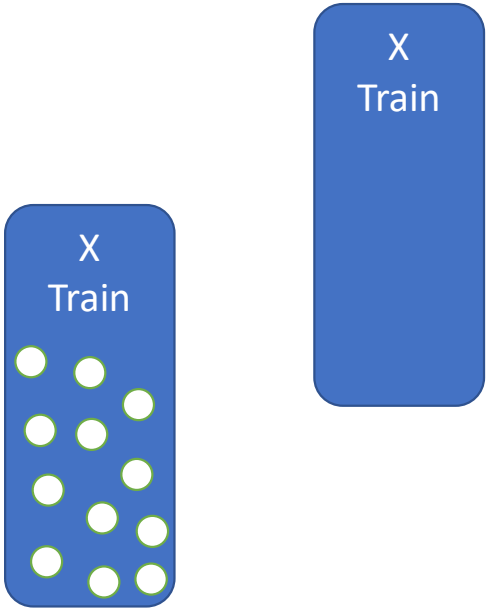
Técnicas de *Ensemble*

O que são *Ensemble*?



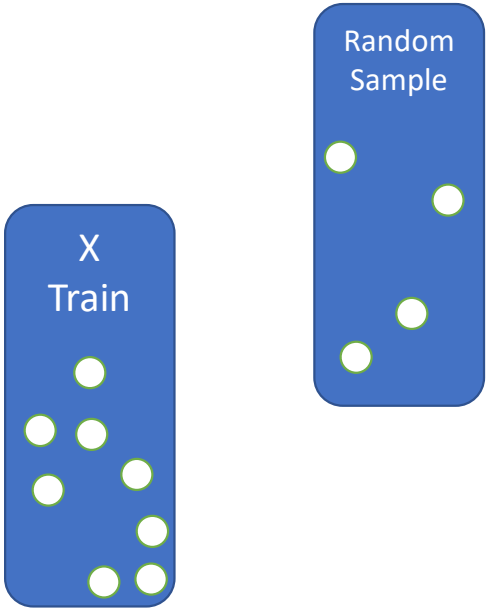
Técnicas de *Ensemble*

O que são *Ensemble*?



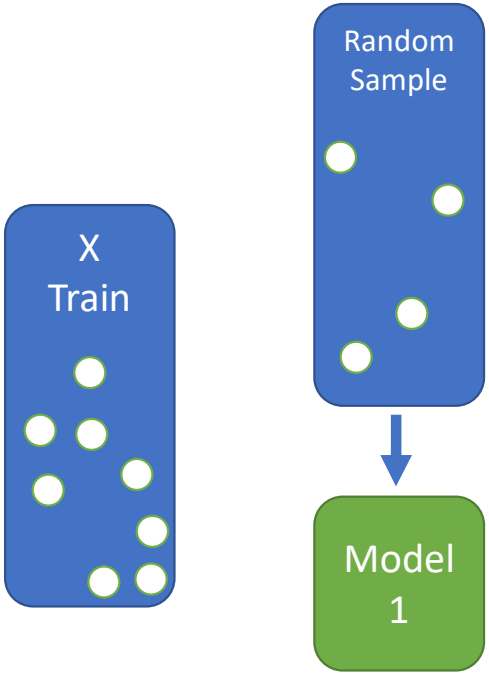
Técnicas de *Ensemble*

O que são *Ensemble*?



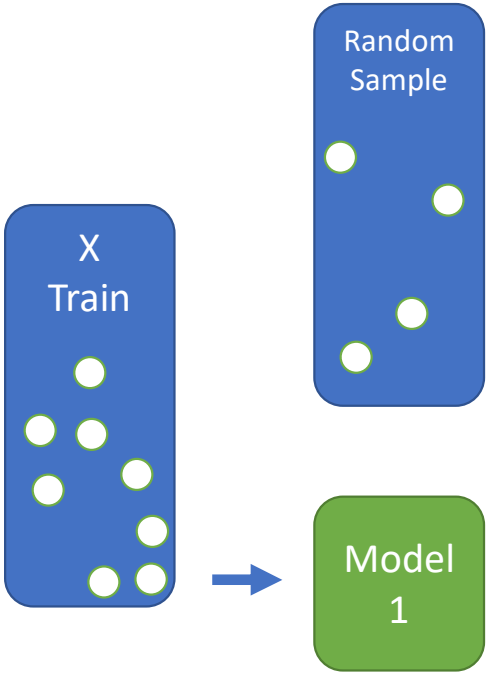
Técnicas de *Ensemble*

O que são *Ensemble*?



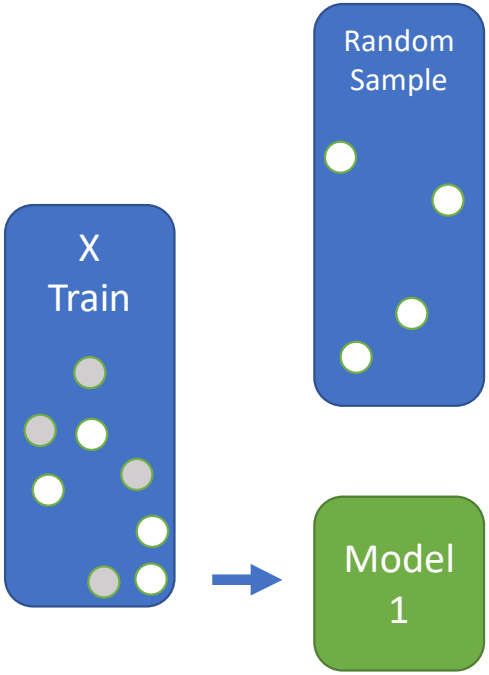
Técnicas de *Ensemble*

O que são *Ensemble*?



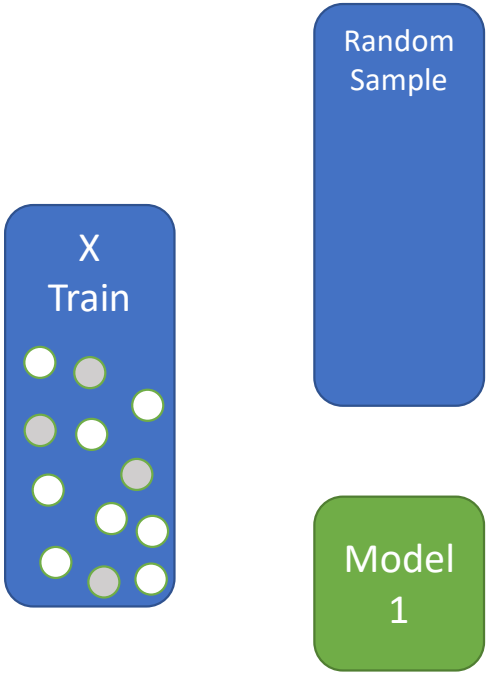
Técnicas de *Ensemble*

O que são *Ensemble*?



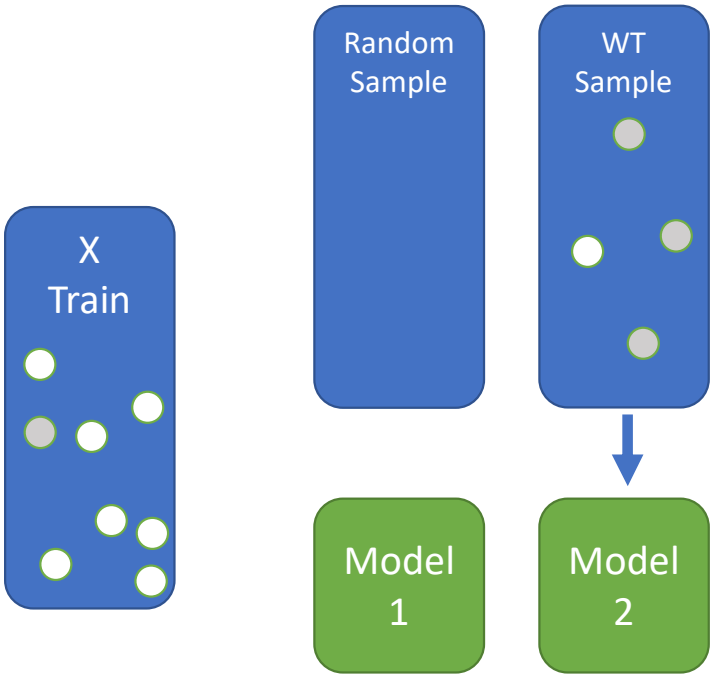
Técnicas de *Ensemble*

O que são *Ensemble*?



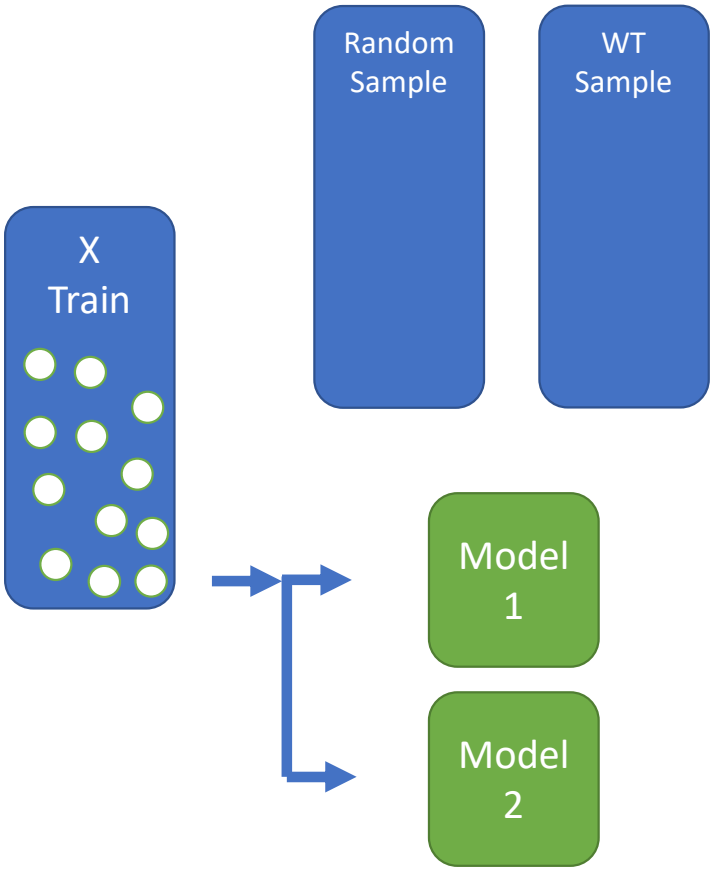
Técnicas de *Ensemble*

O que são *Ensemble*?



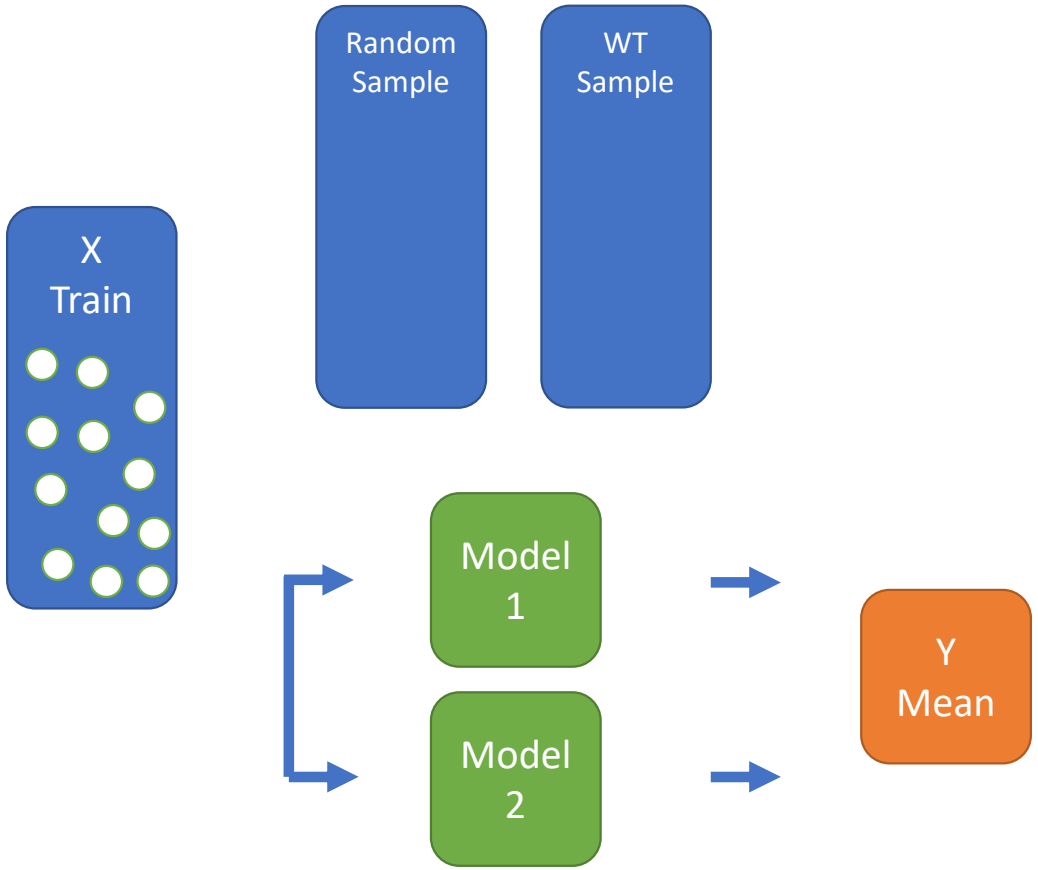
Técnicas de *Ensemble*

O que são *Ensemble*?



Técnicas de *Ensemble*

O que são *Ensemble*?

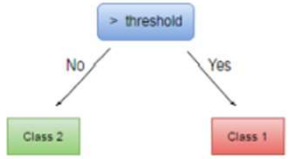


Técnicas de *Ensemble*

Boosting

- Cada novo modelo concentra seus esforços nas observações mais difíceis de se ajustar até o momento, para que obtenhamos, ao final do processo, um aprendiz forte (*Strong Learner*) com menor viés.

- Mais adequado a modelos que tenham baixa variância mas alto viés
  - Ex. Se usarmos árvores de decisão como nossos modelos básicos, escolheremos árvores de decisão rasas – *Stump Decision Tree*
  - Busca-se modelos menos dispendiosos em termos computacionais. Como os cálculos para ajustar os diferentes modelos não podem ser feitos em paralelo pode ser muito custoso ajustar sequencialmente vários modelos complexos.



Stump

Técnicas de *Ensemble*

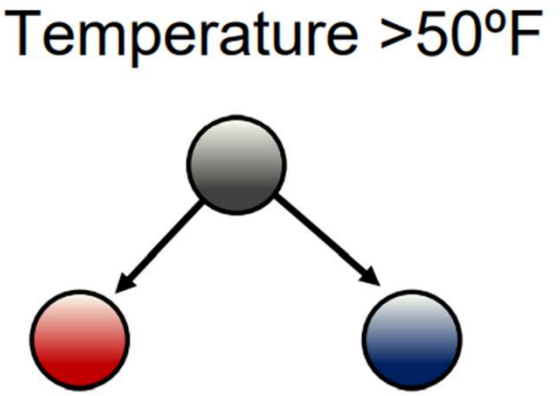
Adaboosting – Adaptative Boosting

- O *Adaboost* deve atender a duas condições:
  - O classificador deve ser treinado interativamente em vários exemplos de treinamento (com peso).
  - Em cada iteração, ele tenta fornecer um ajuste excelente para esses exemplos, minimizando o erro de treinamento.

- Qualquer algoritmo de aprendizado de máquina pode ser usado como classificador base desde que aceite pesos no conjunto de treinamento.
- Quais informações dos modelos anteriores levamos em consideração para ajustar o modelo atual?
- Como eles serão agregados como agregamos o modelo atual aos anteriores?

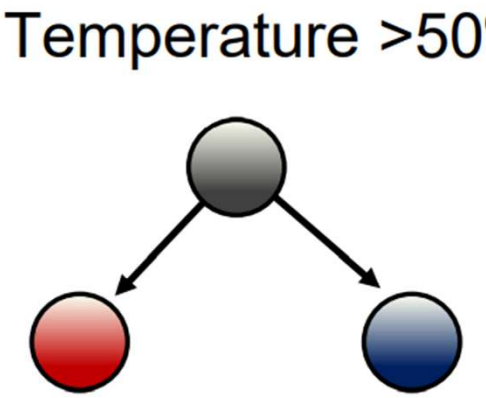
Técnicas de *Ensemble*

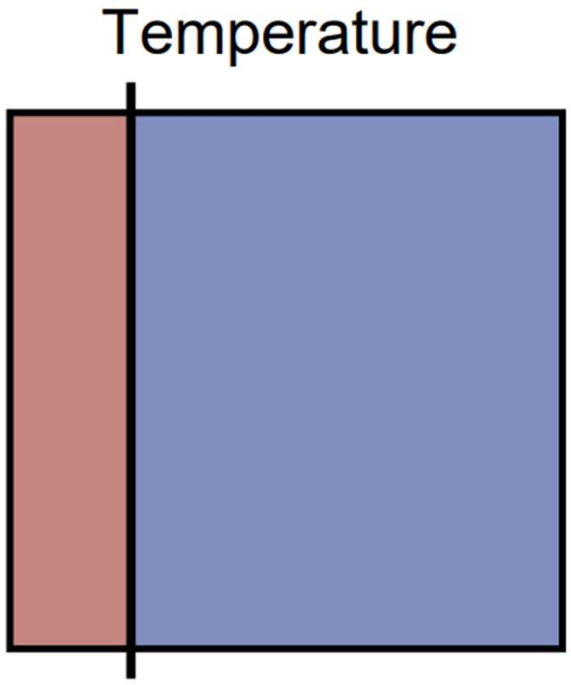
Decision Stump: the Boosting Base Learner



Técnicas de *Ensemble*

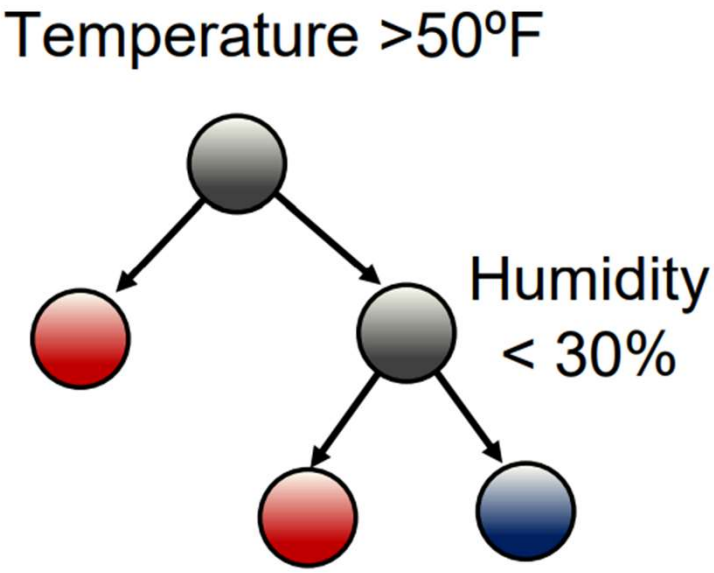
Decision Stump: the Boosting Base Learner

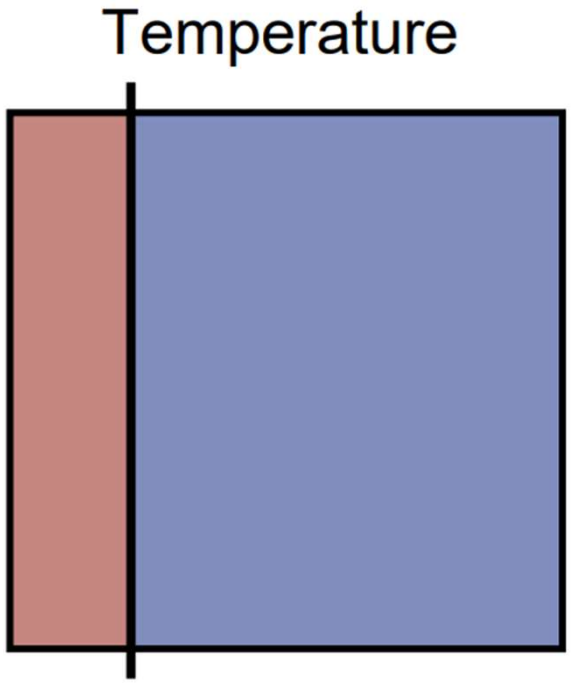




Técnicas de *Ensemble*

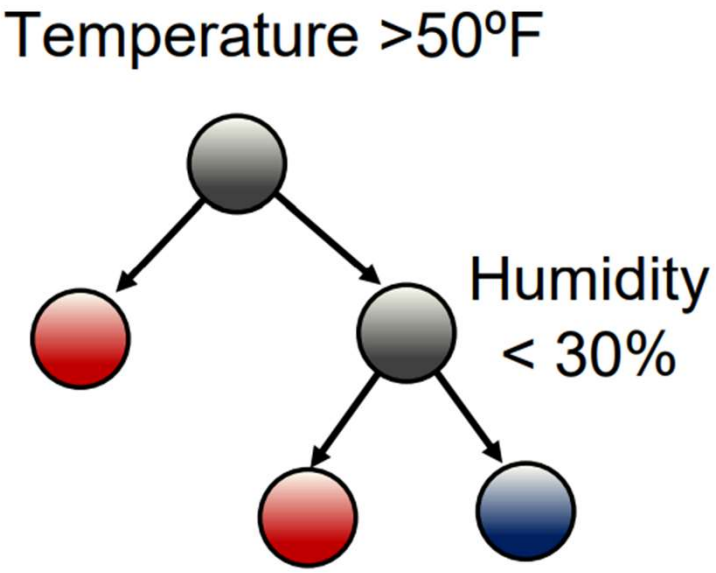
Decision Stump: the Boosting Base Learner

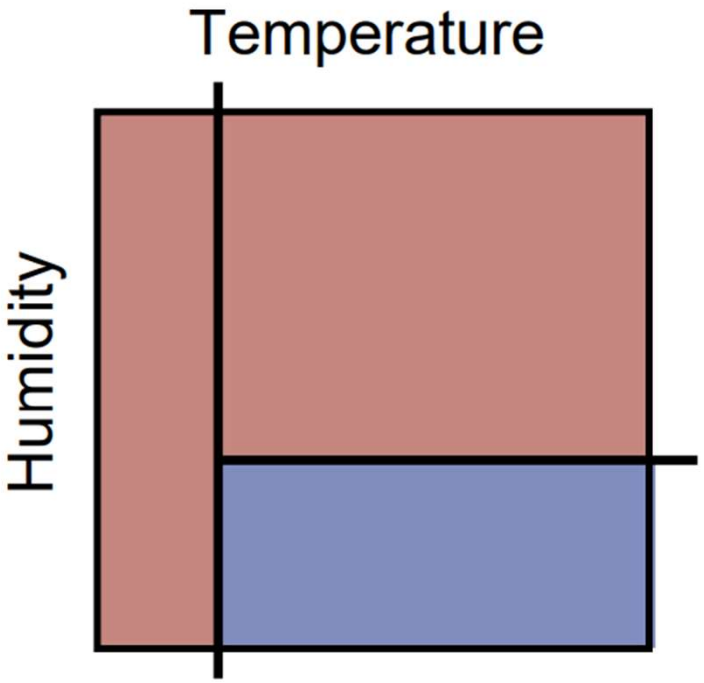




Técnicas de *Ensemble*

Decision Stump: the Boosting Base Learner

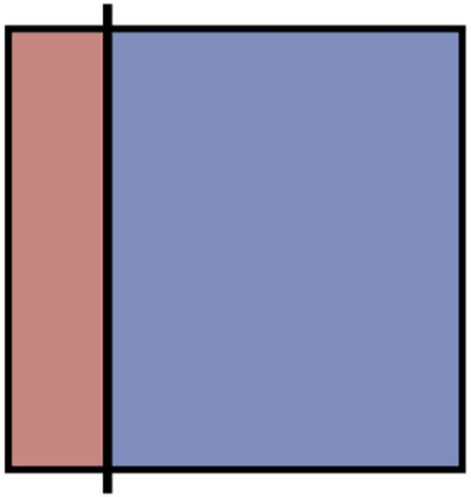




Técnicas de *Ensemble*

Overview of Boosting

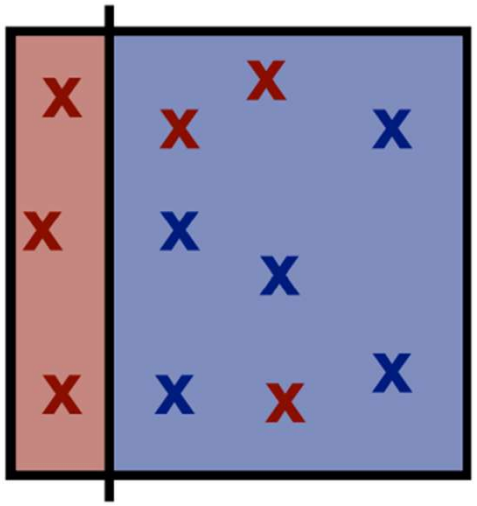
Create initial decision stump



Técnicas de *Ensemble*

Overview of Boosting

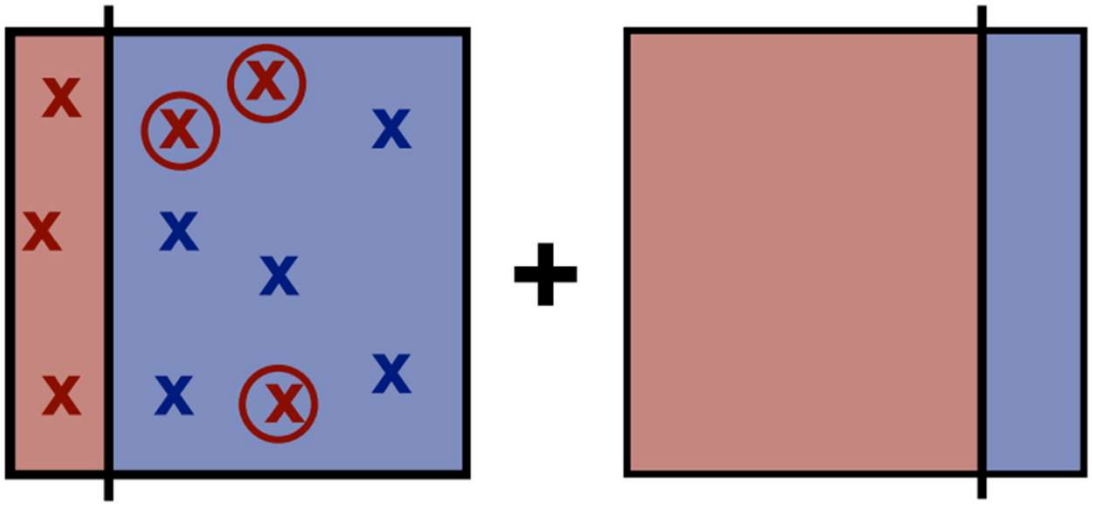
Fit to data and calculate residuals



Técnicas de *Ensemble*

Overview of Boosting

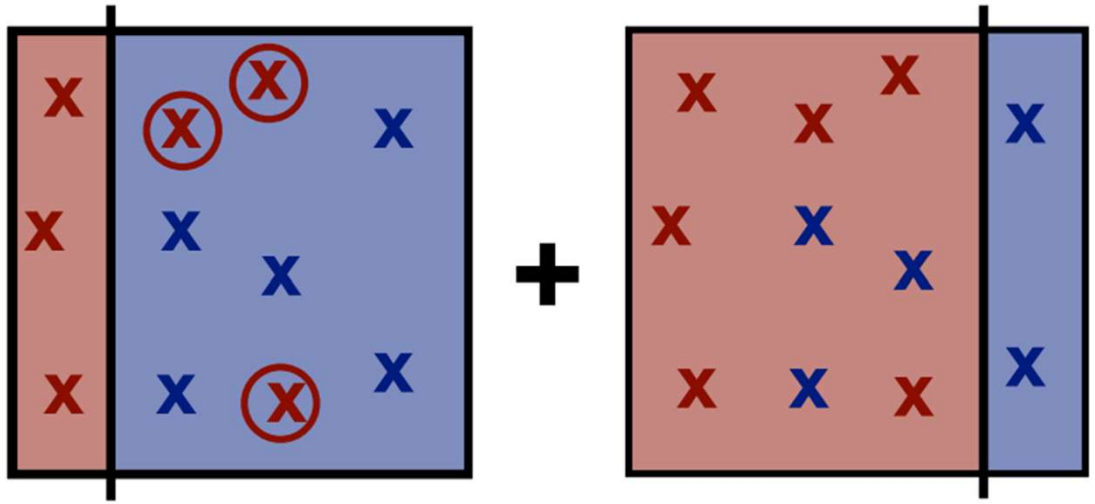
Find new decision stump to fit weighted residuals



Técnicas de *Ensemble*

Overview of Boosting

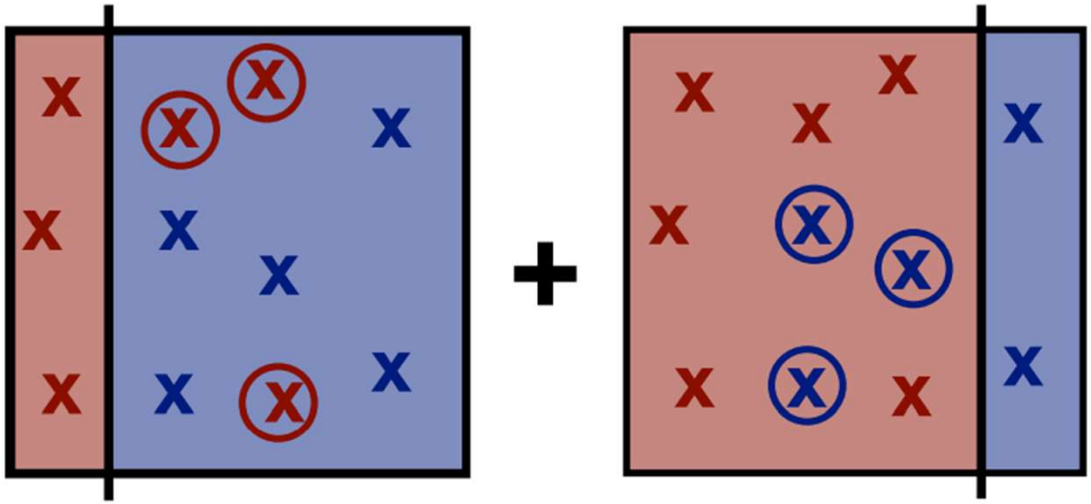
Fit new decision stump to current residuals



Técnicas de *Ensemble*

Overview of Boosting

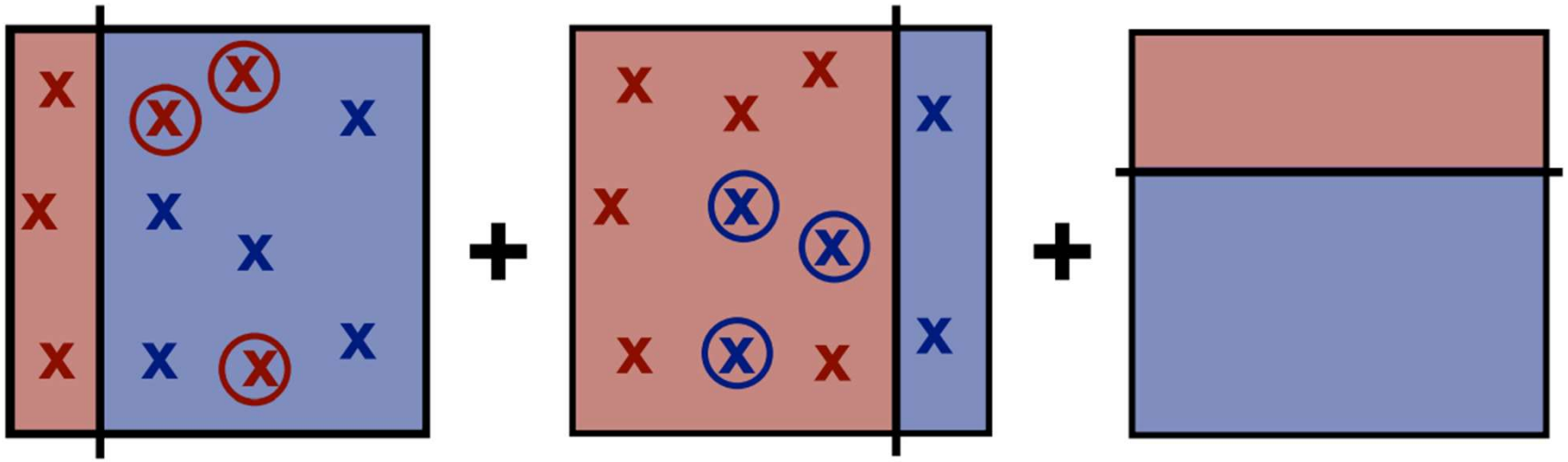
Calculate errors and weight data points



Técnicas de *Ensemble*

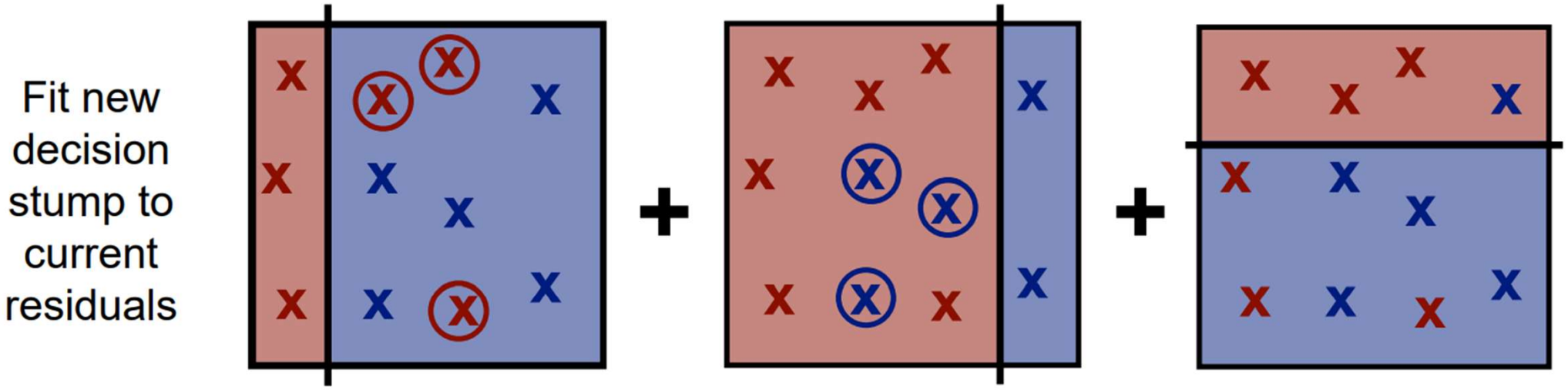
Overview of Boosting

Find new decision stump to fit weighted residuals

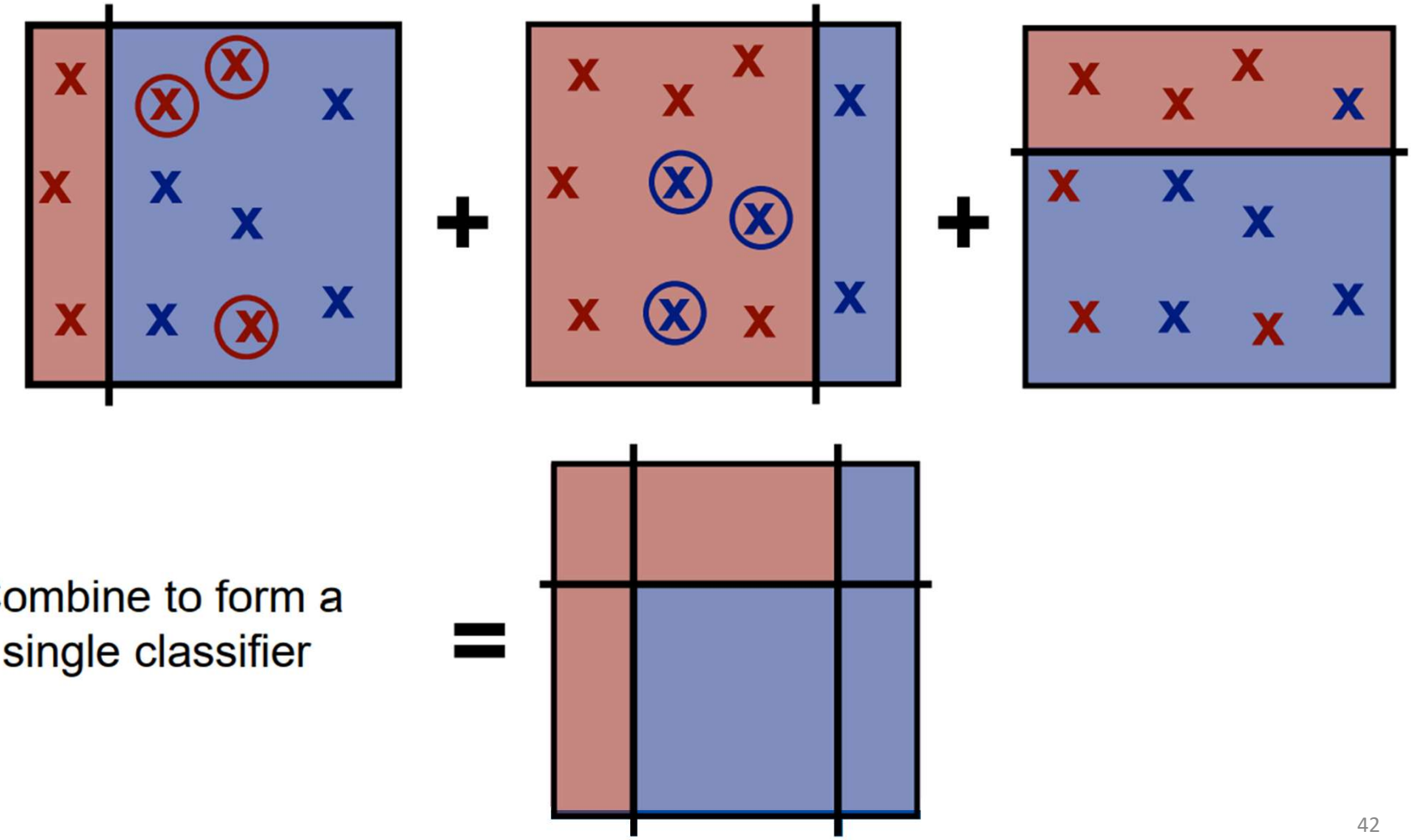


Técnicas de *Ensemble*

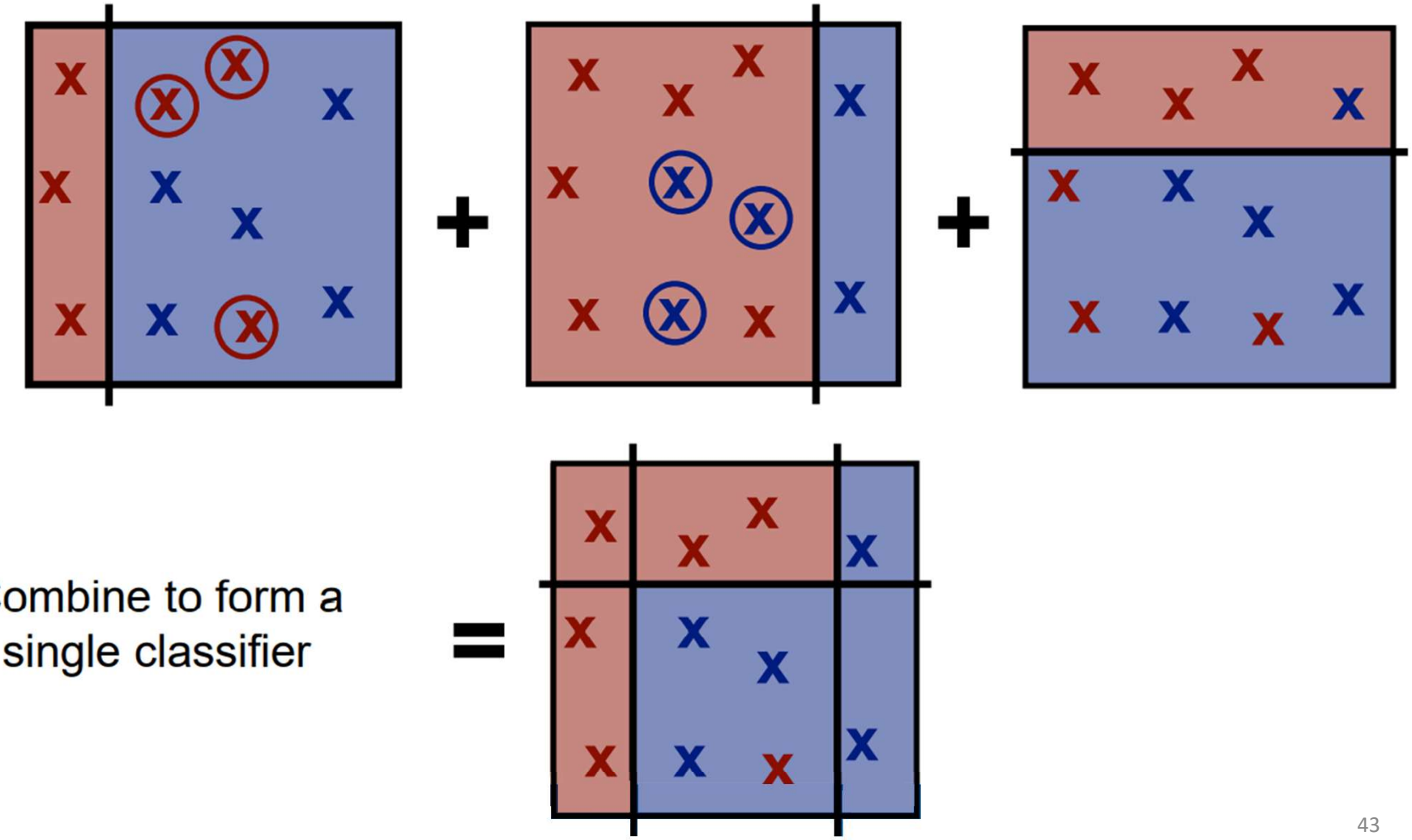
Overview of Boosting



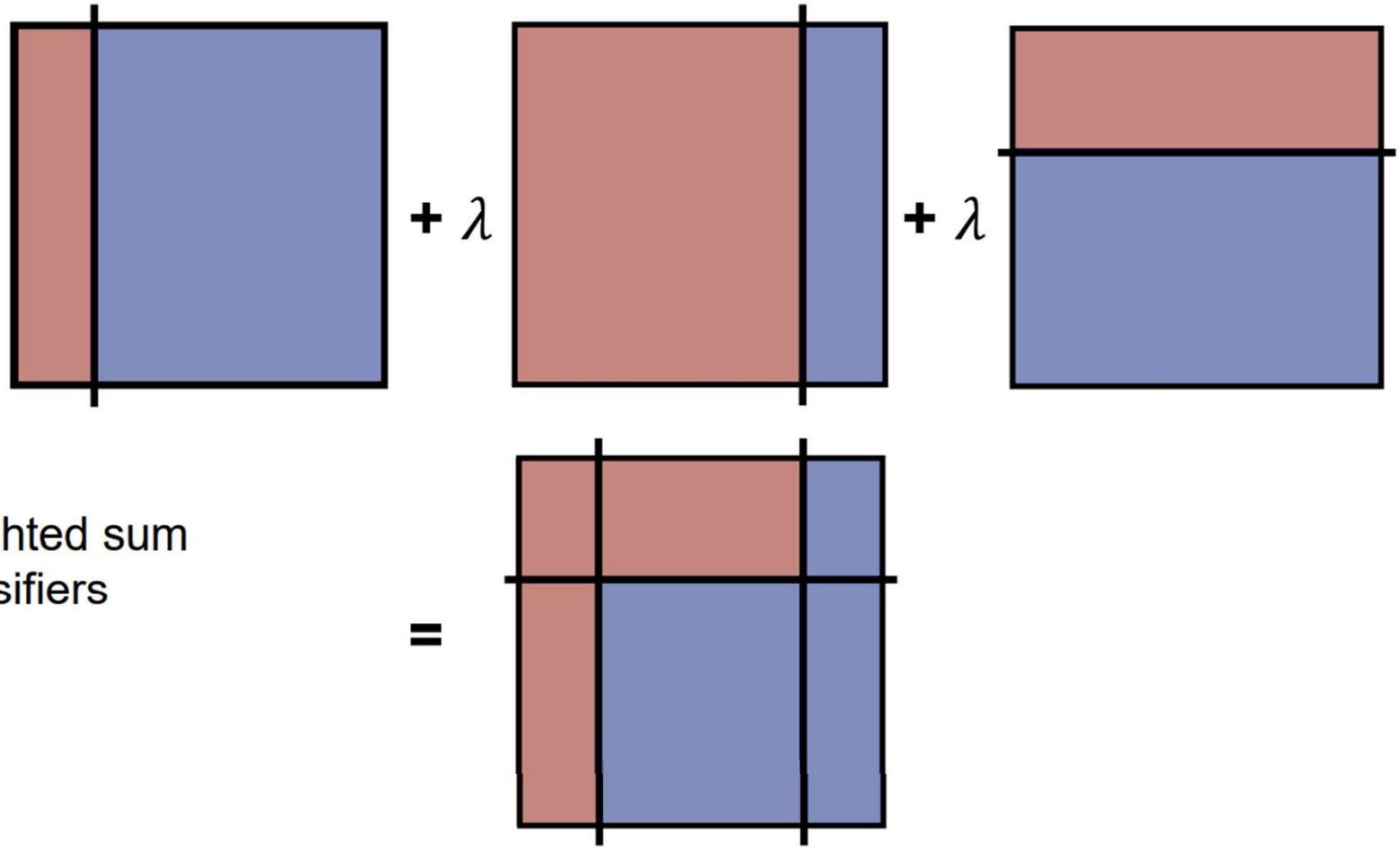
Técnicas de *Ensemble*



Técnicas de *Ensemble*

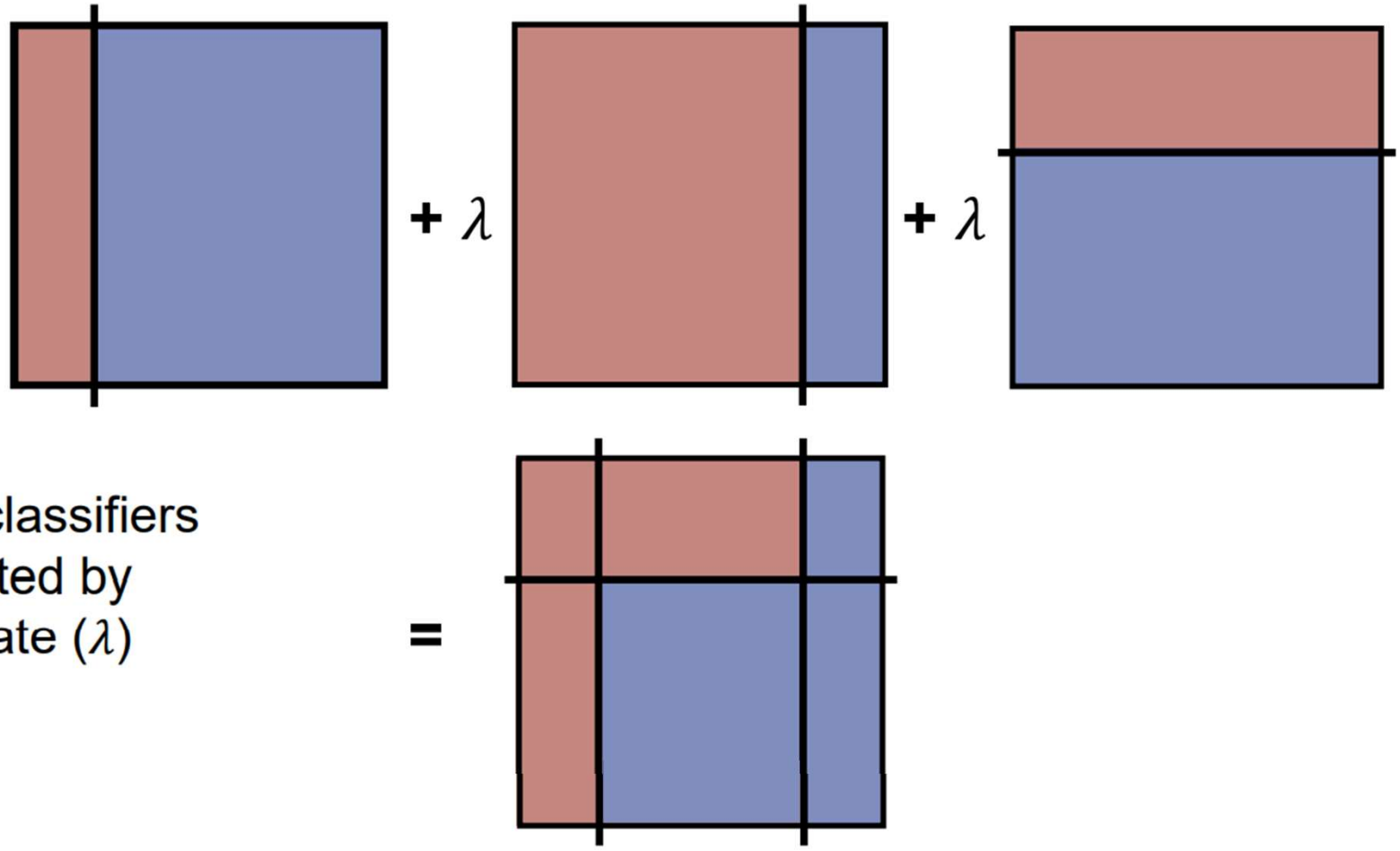


Técnicas de *Ensemble*



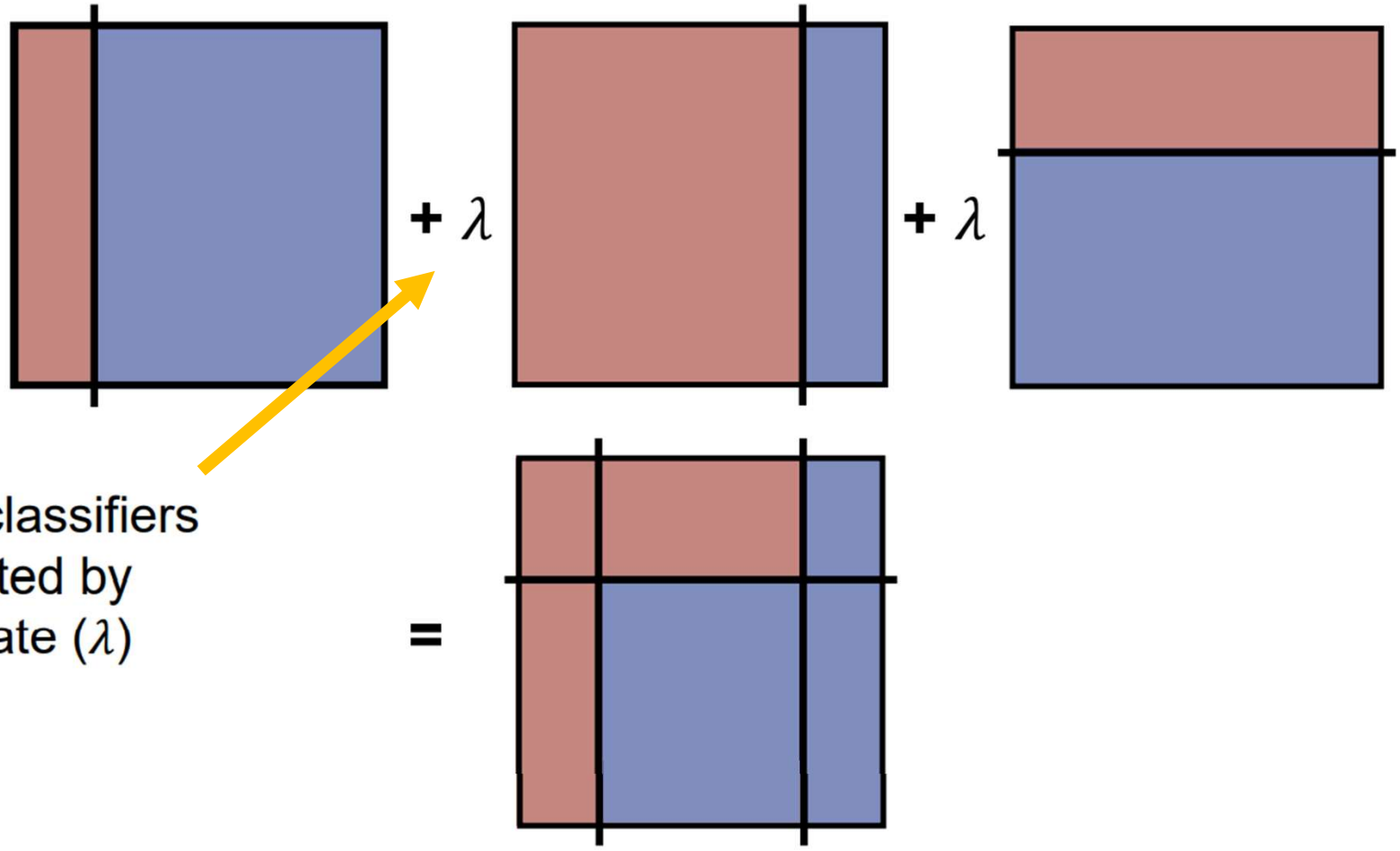
Result is weighted sum
of all classifiers

Técnicas de *Ensemble*



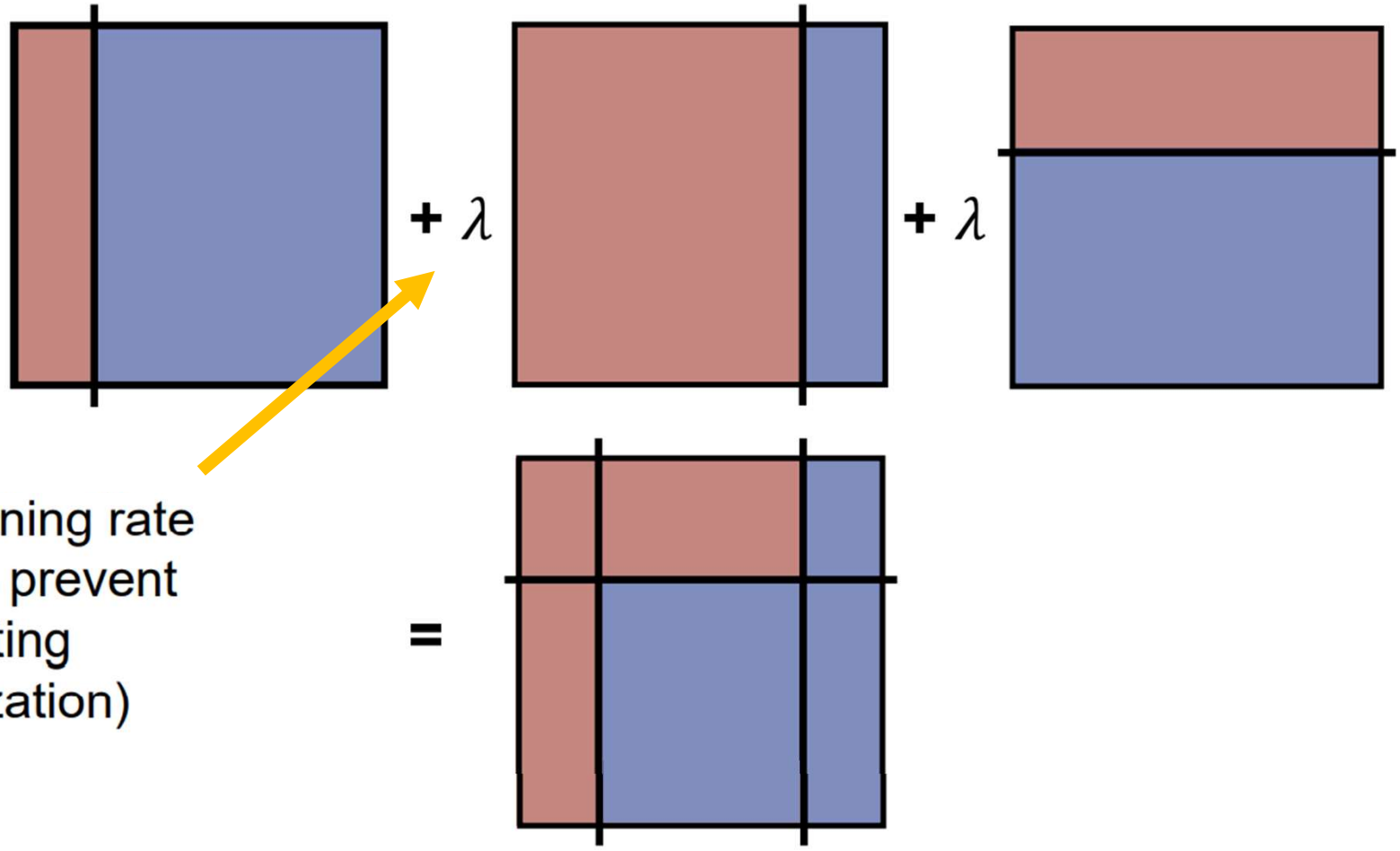
Successive classifiers are weighted by learning rate (λ)

Técnicas de *Ensemble*



Successive classifiers are weighted by learning rate (λ)

Técnicas de *Ensemble*



Using a learning rate < 1.0 helps prevent overfitting (regularization)

Técnicas de *Ensemble*

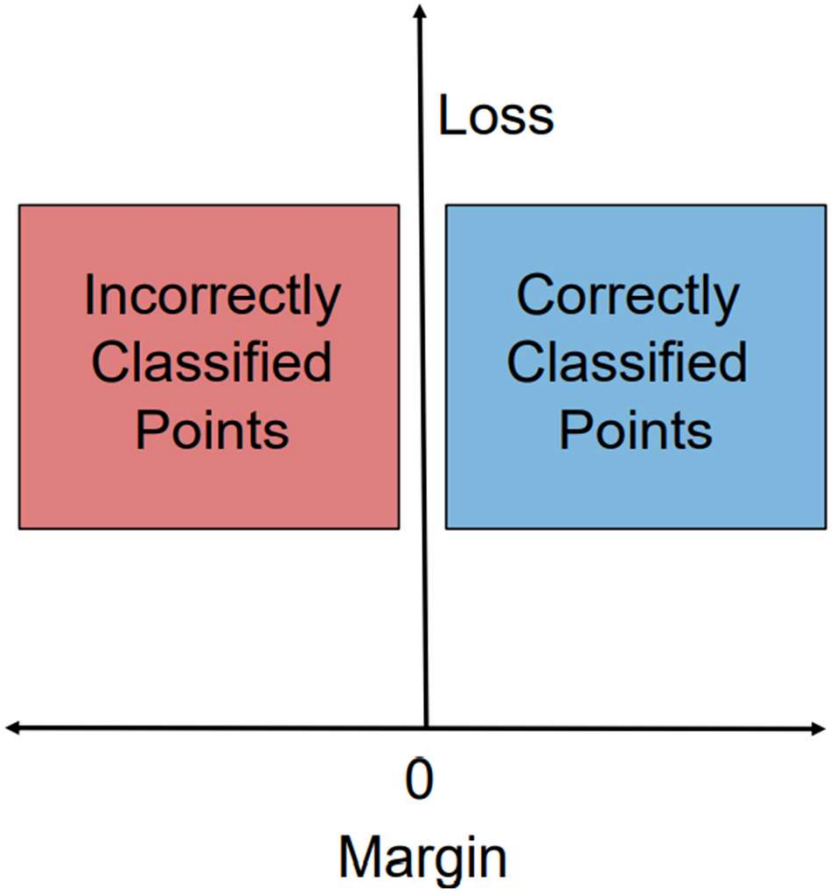
Overview of Boosting

- *Boosting* utiliza diferentes funções de erro
- A cada estado, a margem é determinada por cada ponto.
- Função de erro é a função que resulta na penalidade de cada residual.
- Residual é y-F(x), isto é a diferença entre valor predito e observado.

Técnicas de *Ensemble*

Overview of Boosting

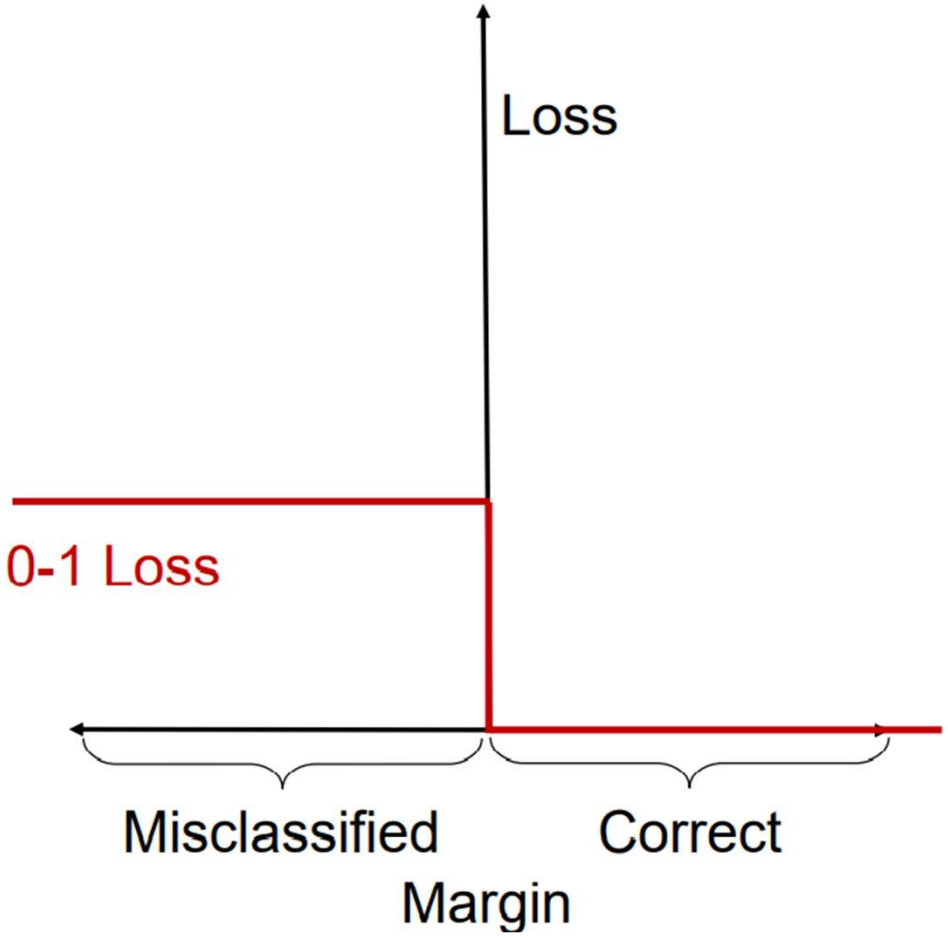
- *Boosting* utiliza diferentes fuções de erro.
- A cada estágio a margem é determinada por cada ponto.
- Margem é positive para pontos classificados corretamente e, negativa, para classificados erroneamente.
- O valor da função de erro é calculada a partir da margem.



Técnicas de *Ensemble*

Overview of Boosting

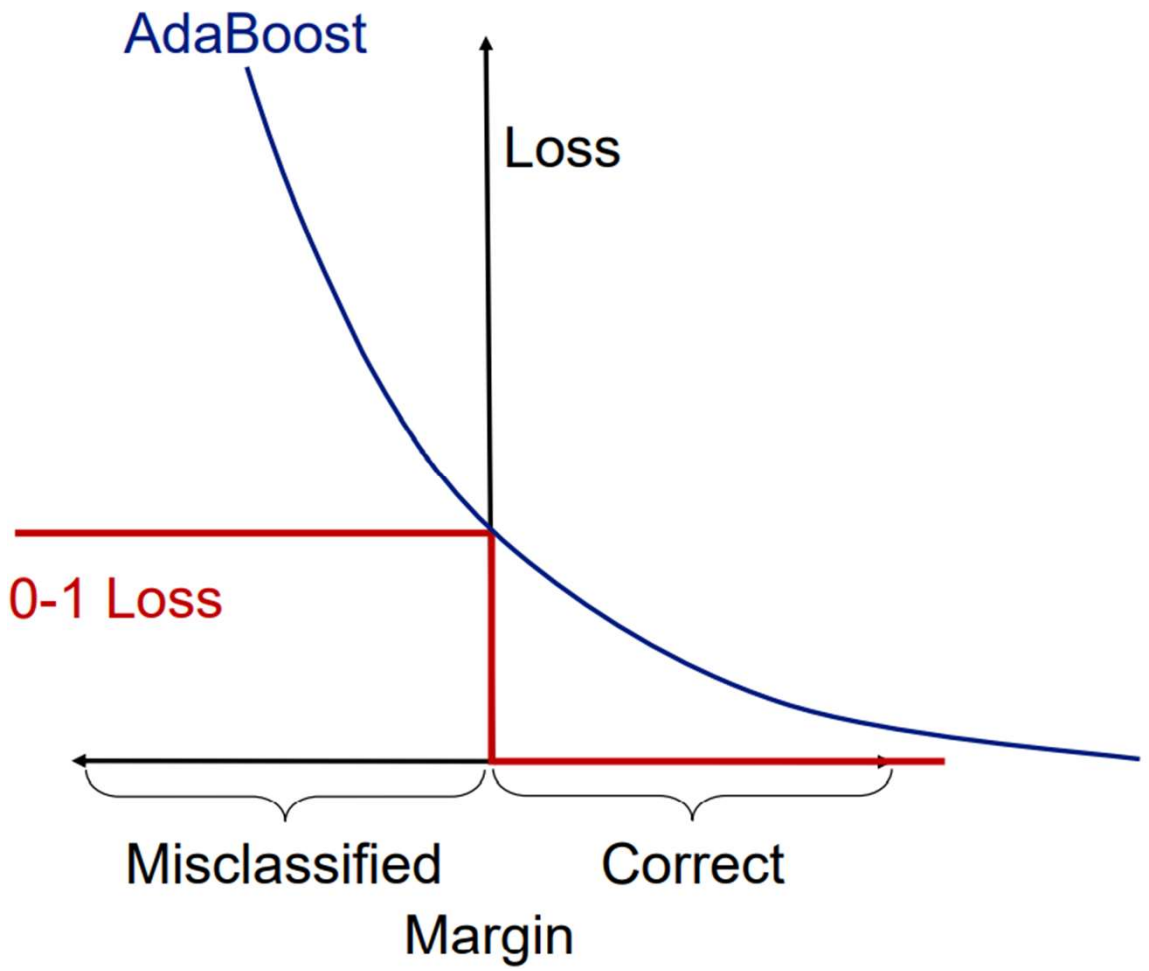
- 0 - 1 Função de erro
- O 0 – 1 erro multiplica os pontos erroneamente classificados por 1.
- Os pontos corretos são ignorados.
- Em tese é a função de erro “ideal”.
- Dificuldade de otimizar *non-smooth* e *non-convex.*



Técnicas de *Ensemble*

Overview of Boosting

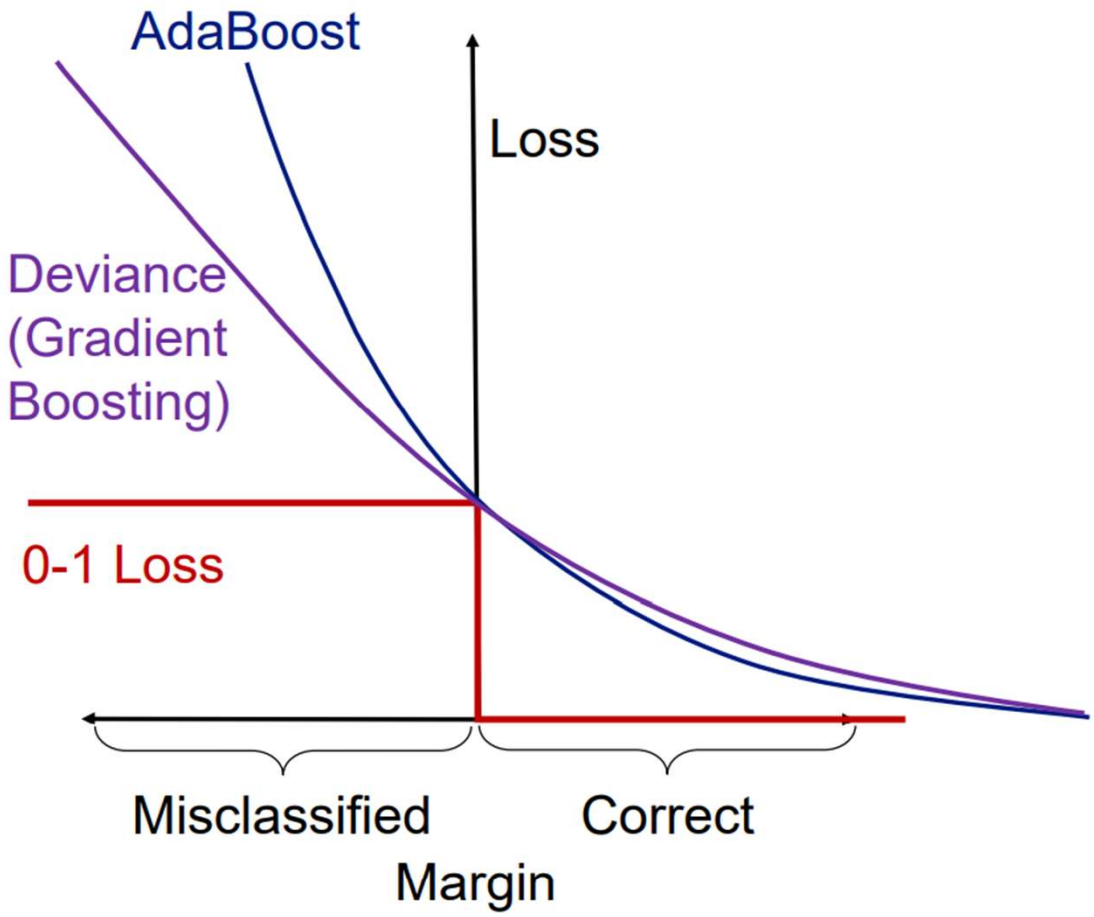
- *AdaBoost* Função de erro
- *AdaBoost* = *Adaptive Boosting*
- Função de erro é exponencial: exp(-margin).
- Mais sensível a *outliers*.



Técnicas de *Ensemble*

Overview of Boosting

- *Gradient Boosting* Função de erro
- Método generalizado de Boosting que usa diferentes funções de erro.
- Implementação mais comum usa função de erro log binomial: log(1 + exp(-margin)).
- Mais robusto a *outliers* quando comparado ao *AdaBoost*.



Técnicas de *Ensemble*

Overview of Boosting

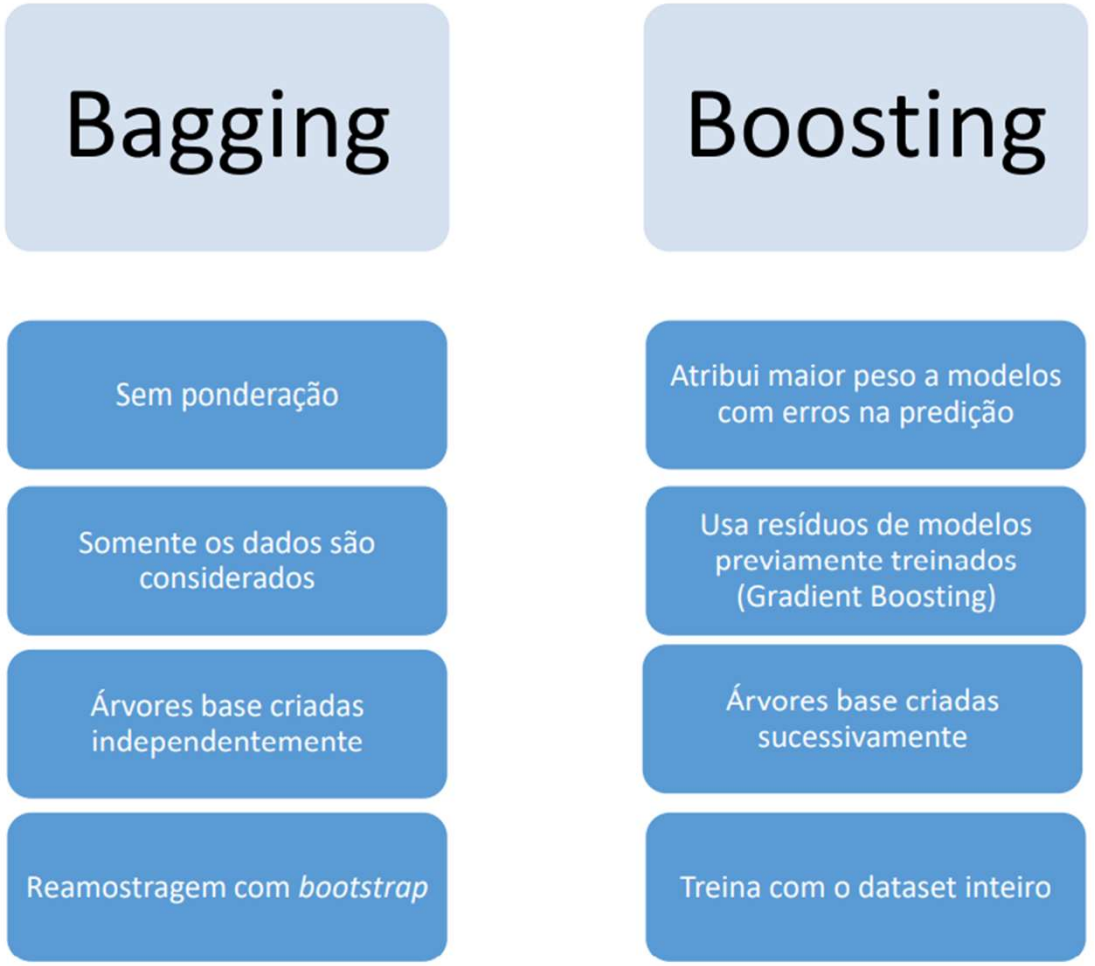
- XGBOOST: significa *eXtreme Gradient Boosting*.

- Técnicas de aumento de gradiente são geralmente muito lentas na implementação por causa do treinamento do modelo sequencial. Portanto, eles não são muito escaláveis.

- *Gradient Boosting* turbinado:
  - Computação distribuída para treinar modelos muito grandes usando um cluster de máquinas.
  - Computação *Out-of-Core* para conjuntos de dados muito grandes que não cabem na memória.
  - Otimização de cache de estruturas de dados e algoritmo para fazer o melhor uso do hardware.

Técnicas de *Ensemble*

Bagging vs Boosting



Técnicas de *Ensemble*

Stacking

- *Stacking*: considera aprendizes fracos e heterogêneos, aprendendo em paralelo e combinando através do treinamento de um meta-modelo para gerar uma previsão com base nas diferentes previsões dos modelos fracos.

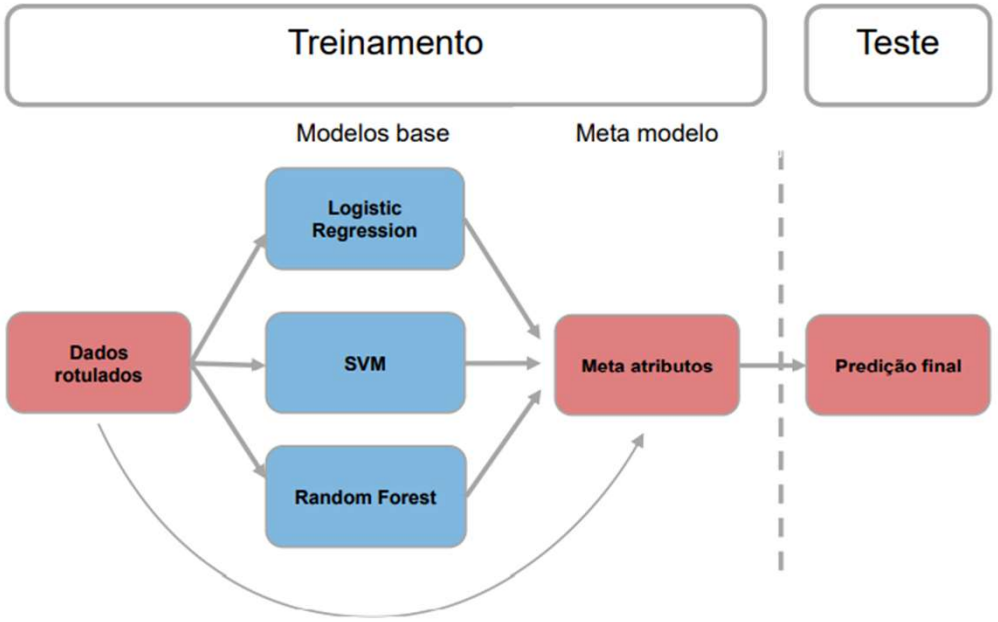
- Difere de *boosting* e *bagging* principalmente em dois pontos:
  1. Considera aprendizes fracos heterogêneos, em que diferentes algoritmos de aprendizado são combinados, enquanto que *boosting* e *bagging* consideram principalmente aprendizes fracos homogêneos.
  2. Aprende a combinar os modelos base usando um meta modelo, ao passo que os outros combinam aprendizes fracos seguindo algoritmos determinísticos.

- Ex: StackingClassifier, StackingRegressor

Técnicas de *Ensemble*

Stacking

- A produção dos modelos de base pode ser combinada por maioria de votos ou ponderada.
- Dados adicionais de retenção são necessários se parâmetros de meta modelos forem usados.
- Natural aumento da complexidade do modelo.



E agora?

Comentários