
Tera

## Módulo 4:
## Aprendizado de Máquina Supervisionado

# Aula 21: Random Forests & Ensembles

# Recapitulando…

- Aprendemos como alguns algoritmos de ML funcionam
- Aprendemos o fluxo geral de aprendizado supervionado e algumas etapas cruciais necessárias
- Hoje vamos investigar algumas limitação dos métodos que conhecemos
- E aprender métodos poderosos que mitigam alguns desses problemas!

<code> … </code>

# Árvores de Decisão

- São métodos flexíveis, intuitivos e interpretáveis
- Porém nem sempre geram os modelos mais precisos
- E sofrem muito de problemas de variância e overfitting
- São mais efetivos quando usados como base para métodos mais poderosos

# Ensembling

- Usamos uma técnica simples de **ensembling** para combinar múltiplos classificadores
- Esse técnica é chamada **bagging**
- Vamos entender melhor como e por que ela funciona com um cenário hipotético

# Cenário

- Você quer decidir se você deve **investir um startup promissora S**
- Basicamente você quer prever o sucesso dessa startup
- A Startup opera em uma área pouco familiar para você, então você **contrata 5 especialistas** para auxiliar na sua decisão

# Especialistas

Especialista em media social

Ex-CEO de uma startup adquirida

Operador da bolsa

Ex-funcionário da startup

Investidor da startup

# Especialistas

60% de acerto

70% de acerto

75% de acerto

65% de acerto

70% de acerto

# Especialistas

- Todos têm seus pontos **fortes** e **vieses**
- Você se sentiria mais confortável com a opinião de um deles ou de todos eles?
- Se você somente decidir por **unanimidade**, sua chance de acerto é maior que 99.9% !!
- Se você decidir por **maioria**, sua chance de acerto ainda é de 90%!

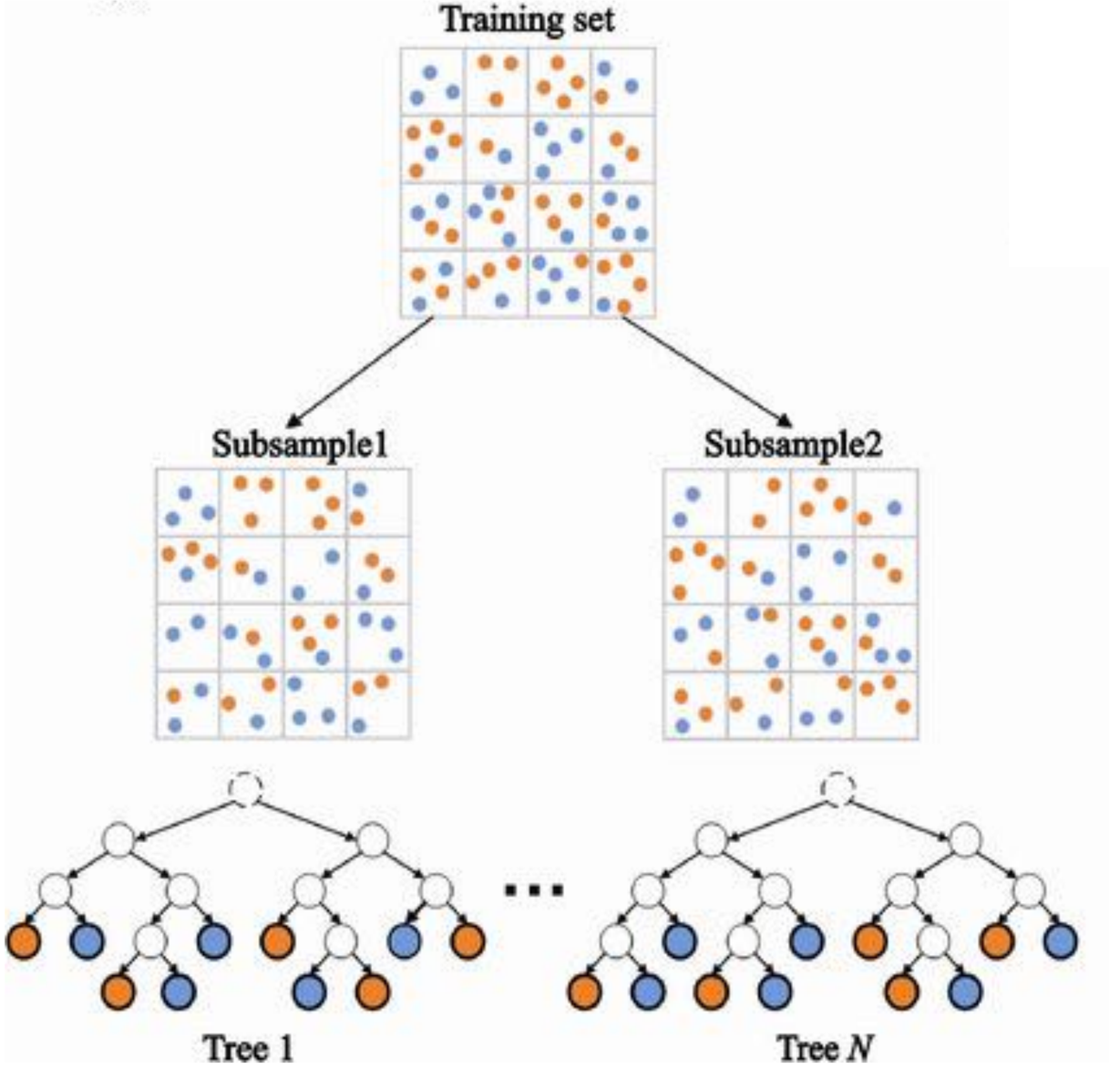
# Especialistas

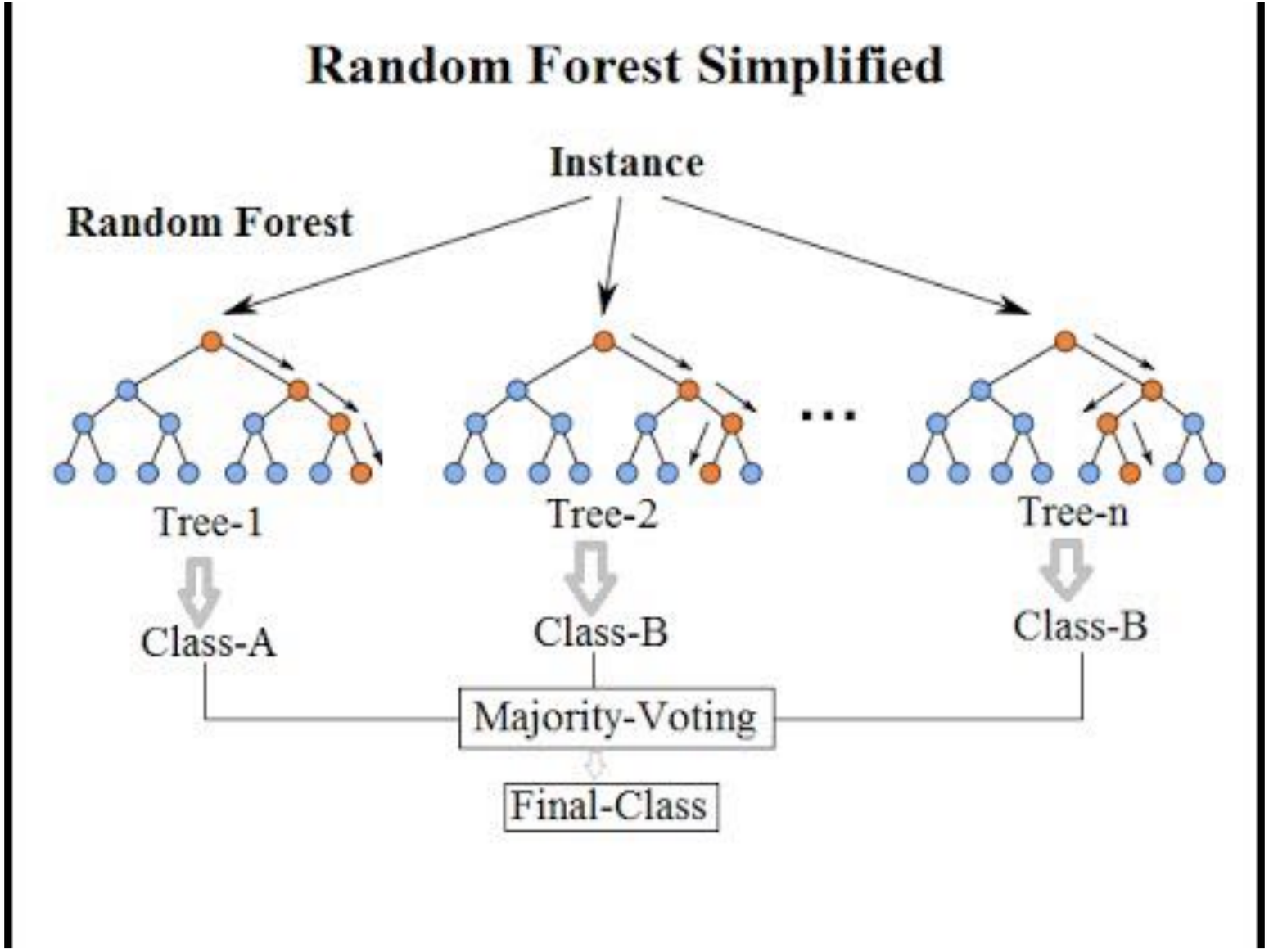
- Esses números assumem que as opiniões dos especialistas **são independentes**, o que raramente é verdade
- Intuitivamente, se todos especialistas fossem ex-funcionários da empresa S, sua confiança seria a mesma que no primeiro cenário?
- Diversidade é bom!
- Agora pense que ao invés de especialistas, temos cinco modelos treinados


SHOW
do MILHÃO
sbt

# Florestas Aleatórias

- Podemos usar as **decisões de várias árvores** para fazer uma predição
- A diversidade entre as árvores é promovida usando **dados e atributos diferentes** pra treinar cada árvore
- Por fim, devemos combinar as decisões das árvores de alguma forma


Training set
Subsample1
Subsample2
Tree 1
Tree N


Random Forest Simplified
Instance
Random Forest
Tree-1
Tree-2
...
Tree-n
Class-A
Class-B
Class-B
Majority-Voting
Final-Class

<code> … </code>

# Ensembles

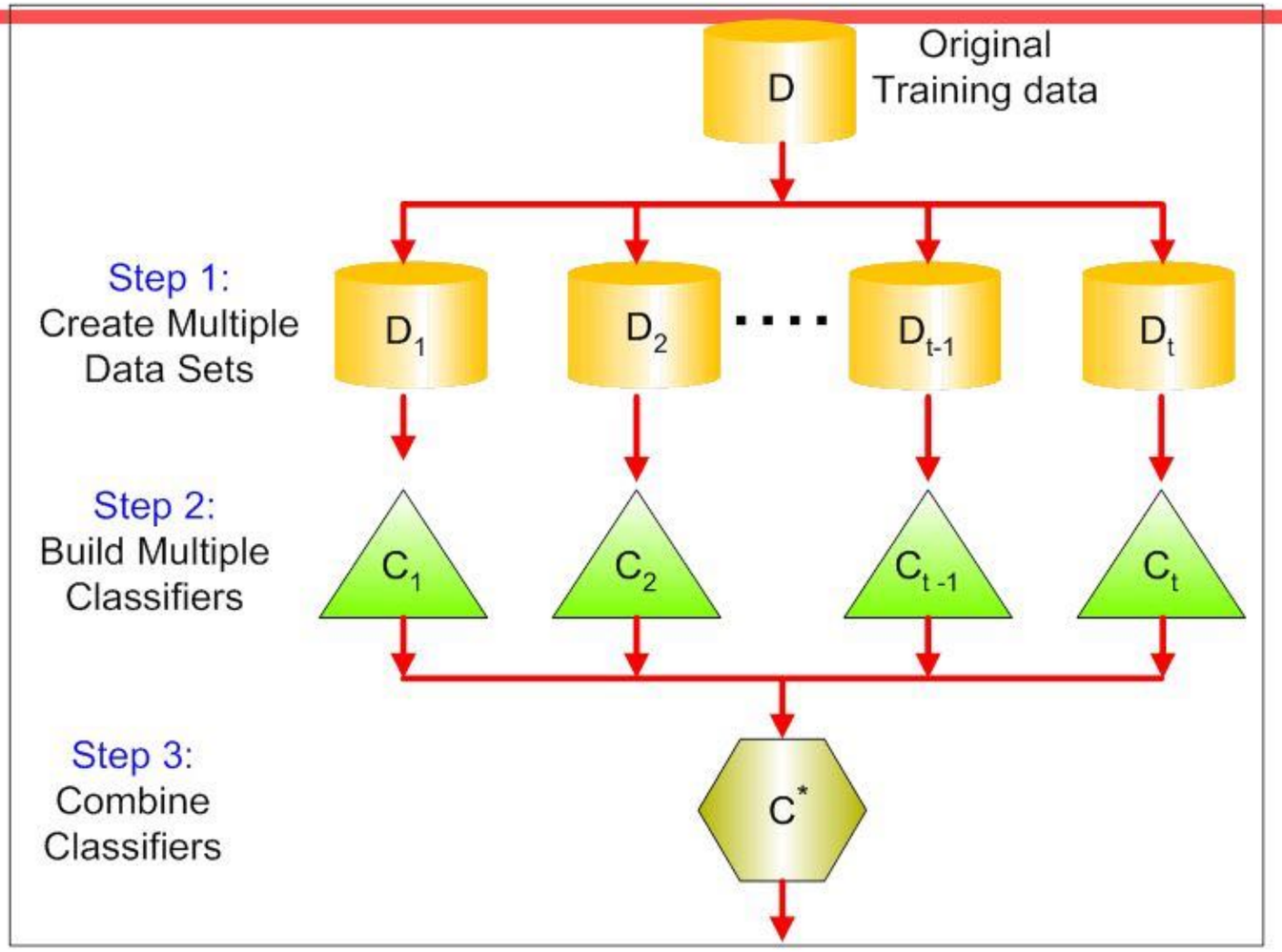
- *Random forests* são possivelmente os ensembles mais populares, porém o conceito vai muito além delas
- Se podemos combinar resultados de árvores de decisão diferentes, por que não **combinar quaisquer modelos diferentes**?
- Não só podemos, como existem inúmeras formas de fazê-lo!

***Ensembles*** *são métodos que combinam* ***múltiplos modelos*** *para obter predições que os modelos constituintes não seriam capaz de obter*

# Bagging

- RandomForest pode ser considerada uma forma **Bagging**
- Bagging consiste em combinar o resultado de *n* classificadores *[$C_1$, $C_2$, …, $C_n$]*, treinados respectivamente em *n* datasets *[$D_1$, $D_2$, …, $D_n$]*
- Cada dataset $D_i$ é uma amostra, com reposição, de *m* elementos do dataset original D

# General Idea



# Bagging

- Cada modelo $C_i$ é considerado um **weak learner**
- Com reposição significa que os $D_i$ podem conter elementos em comum e até mesmo repetidos!
- O resultado dos modelos $C_i$ podem ser combinados de diversas formas:
  - **Regressão**: média das predições
  - **Classificação**: classe majoritária das predições (hard voting), média das probabilidades das predições (soft voting)

<code> … </code>

# O que mais?

- Em bagging, treinamos classificadores com amostras diferentes do dataset, porém com o mesmo algoritmo
- Por que não usar **diferentes algoritmos**?
- Podemos obter diversidade entre os *weak learners* utilizando diferentes:
  - Populações
  - Algoritmos
  - Parametrizações
  - Sementes aleatórias

# Voting

- Quaisquer modelos podem ser combinados através de qualquer estratégia
- **VotingClassifier** do SKLearn pode ser usado para isso
- E essa combinação pode ser também usada em um novo *ensemble*!
- Sua criatividade é o limite

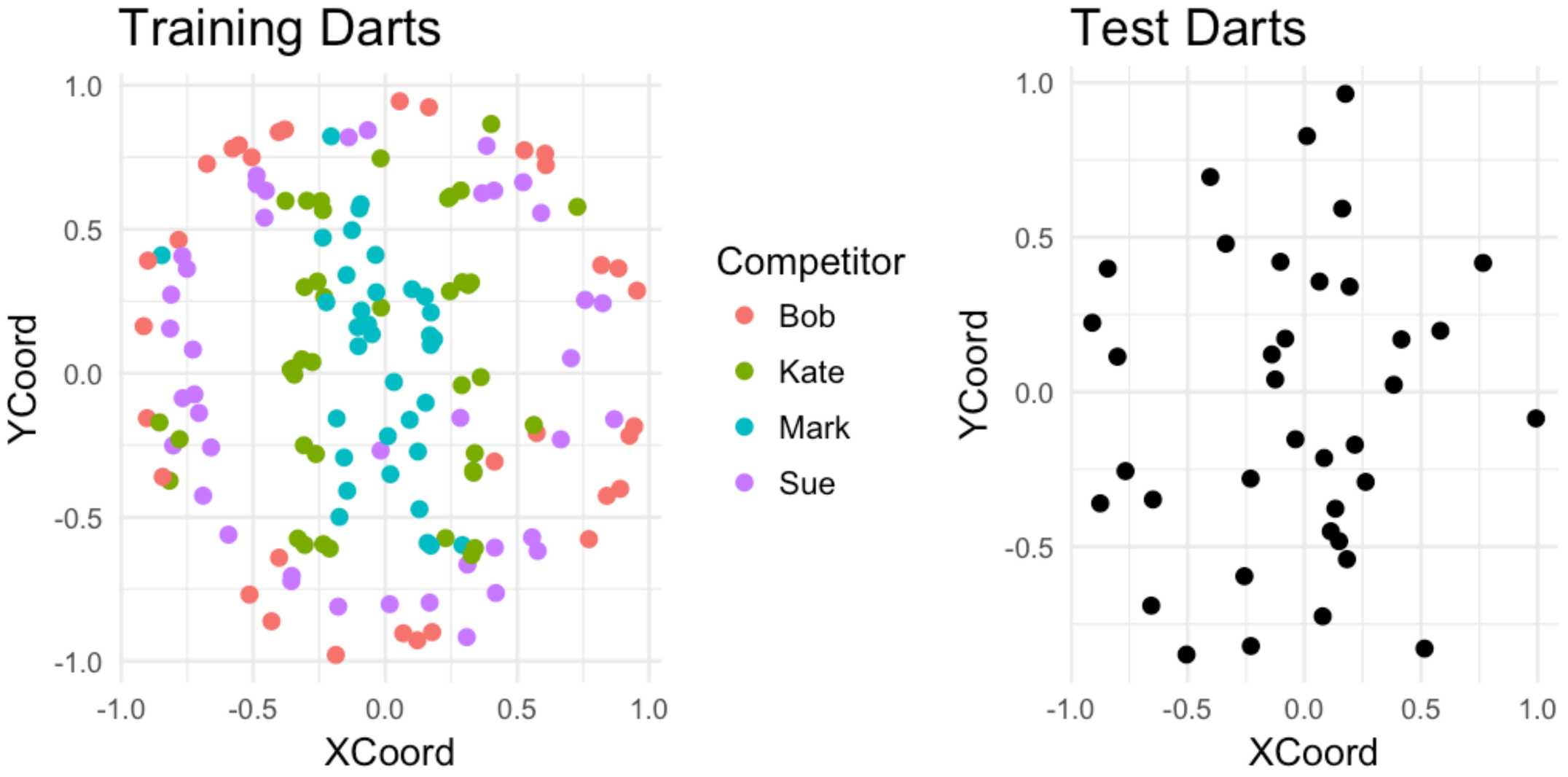
<code> … </code>

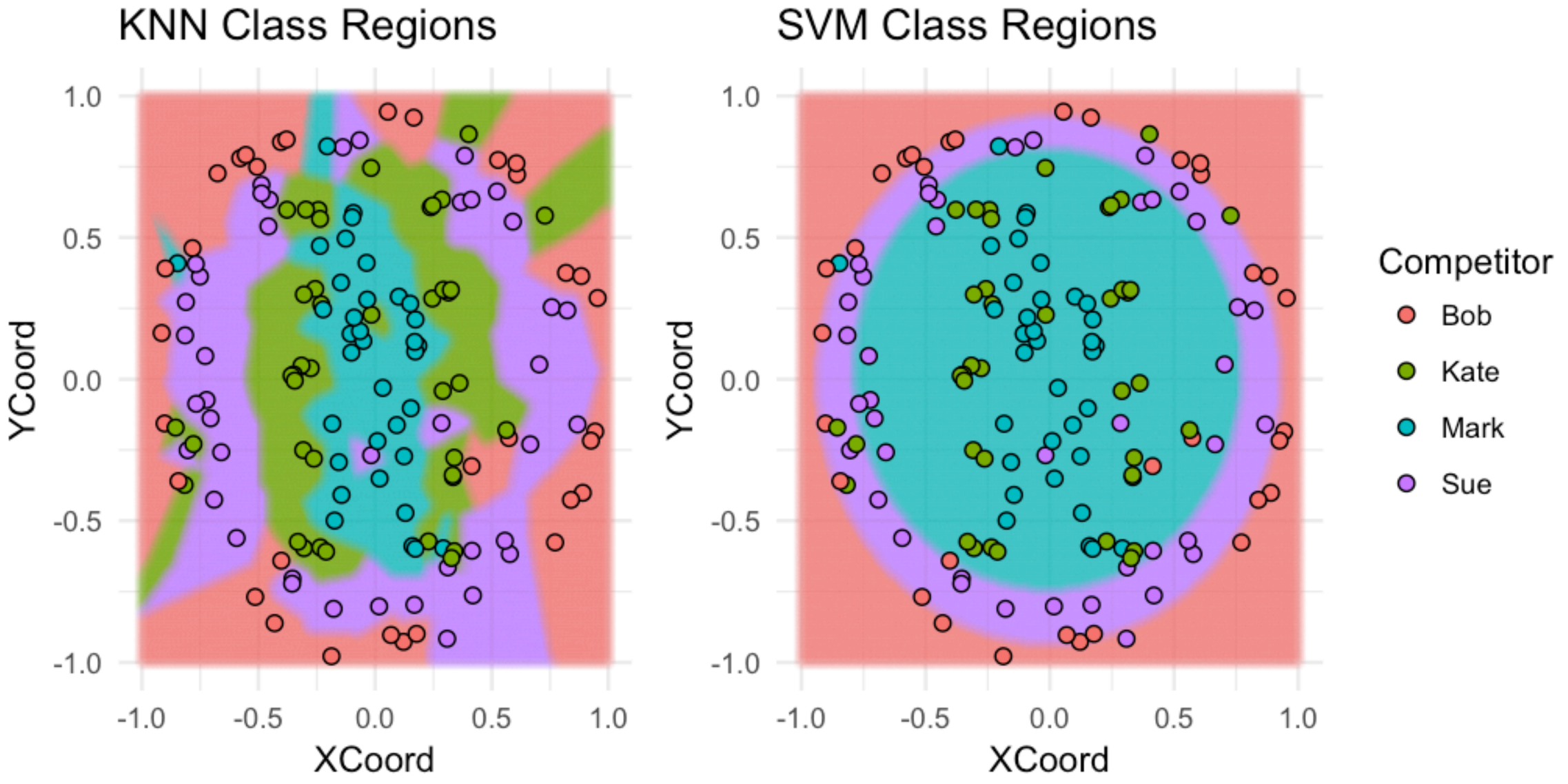
# Stacking

- Vamos supor que treinamos 5 modelos para prever inadimplência e queremos combiná-los com voting

- Obtivemos as seguintes predições para alguns exemplos:

| C1 | C2 | C3 | C4 | C5 | Maioria | Ytrue |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 |


Training Darts
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0
Competitor
Bob
Kate
Mark
Sue
Test Darts


KNN Class Regions
SVM Class Regions
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0
Competitor
Bob
Kate
Mark
Sue

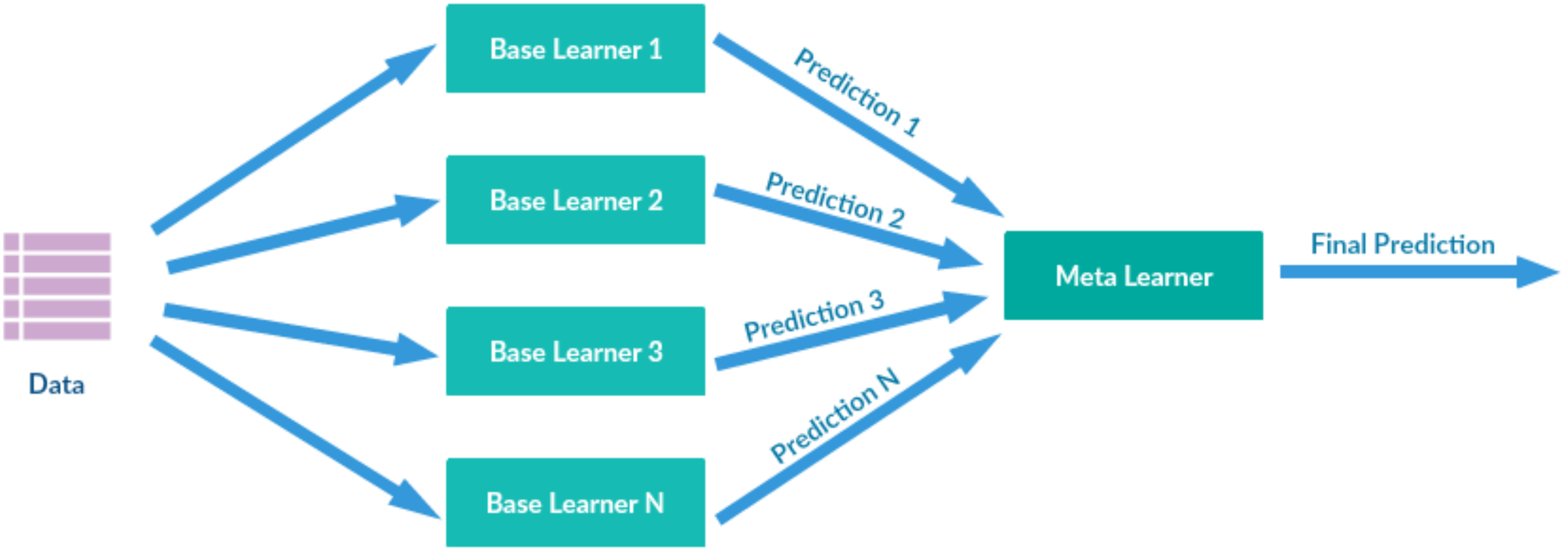
# Stacking

- Acontece que votação não parece a melhor forma de combinar esses resultados
- Existem alguns comportamentos entre os classificadores que parecem mais informativos para determinar uma predição melhor Y
- Por que não usar as predições de C1, C2, C3, C4 e C5 como features X de um novo modelo tentando prever Y?
- Isso é chamado de **Stacking,** e é extremamente poderoso!

# Stacking

- Para cada exemplo no treino, podemos criar uma nova instância de treinamento *[$P_1$, $P_2$, …, $P_n$]* com as predições de cada um dos nossos modelos base *[$C_1$, $C_2$, …, $C_n$]*
- Esse novo modelo é um *meta-learner,* e ele vai aprender a melhor forma de combinar as predições *[$P_1$, $P_2$, …, $P_n$]*
- Ao invés de usarmos as predições binárias dos $C_i$ como *features*, é comum usarmos as probabilidades

# Stacking


Base Learner 1
Base Learner 2
Base Learner 3
Base Learner N
Prediction 1
Prediction 2
Prediction 3
Prediction N
Meta Learner
Final Prediction
Data

<code> … </code>

# Recapitulando

- **Regressão Logística e Árvores de Decisão**: algoritmos de classificação
- **Regressão Linear e Árvores de Regressão**: algoritmos de regressão
- **Ensembles**: Classe de métodos que combinam múltiplos modelos
- **Bagging**: Tipo de ensemble que treina múltiplos modelos do mesmo algoritmo com diferentes amostras dos dados
- **RandomForests**: Tipo de Bagging que utiliza Árvores como weak learners
- **Voting**: Tipo de ensemble que utiliza a votação (ou média) do resultado de múltiplos modelos diferentes
- **Stacking**: Tipo de ensemble que treina um modelo (meta-learner) com a saída de outros modelos (weak-learners)

# Quando usar cada um?

| Método | Quando usar? |
| --- | --- |
| Regressão Logística | Primeiro a se tentar quando tratando de um novo problema: baseline |
| Árvores de Decisão | Quando se quer entender melhor as features e sua correlação com o target |
| Random Forests (ou GradientBoosting) | Passo seguinte à Regressão Logística. Melhora de desempenho com esforço mínimo |
| Bagging, Voting e Stacking | Quando melhorias de desempenho tem um grande impacto prático. Quando você quer diferentes modelos pra diferentes populações |

# Ensembles

- Então por que não criar milhares de modelos para todos os problemas?
- **Restrição computacional** (treino longo e latência na predição), complexidade de implementação, dificuldade de manutenção e ganhos marginais a partir de um certo ponto
- Porém existem sim *ensembles* extremamente complexos e poderosos, especialmente em competições

# Conclusões

- Técnicas de ensembles podem ser bem poderosas, porém aumentam consideravelmente a **complexidade** da sua solução
- São muito populares em competições pois abrem infinitas possibilidades de modelagem
- Na prática, porém, são difíceis de avaliar corretamente, colocar em produção e manter

# DÚVIDAS?!