

# **Ensembles Learning ou Aprendizagem em Conjunto**

---

Cristiane Neri Nobre



# Ensembles

- Métodos que geram muitos classificadores e combinam os seus resultados
  - “Consultar vários especialistas para tomada de decisão”
- É amplamente aceito que o desempenho de um conjunto de muitos classificadores fracos é geralmente melhor do que um único classificador, dada a mesma quantidade de informação de treinamento

# Ensembles

## `sklearn.ensemble`: Ensemble Methods

The `sklearn.ensemble` module includes ensemble-based methods for classification, regression and anomaly detection.

**User guide:** See the [Ensemble methods](#) section for further details.

<code>ensemble.AdaBoostClassifier(...)</code>	An AdaBoost classifier.
<code>ensemble.AdaBoostRegressor([base_estimator, ...])</code>	An AdaBoost regressor.
<code>ensemble.BaggingClassifier([base_estimator, ...])</code>	A Bagging classifier.
<code>ensemble.BaggingRegressor([base_estimator, ...])</code>	A Bagging regressor.
<code>ensemble.ExtraTreesClassifier(...)</code>	An extra-trees classifier.
<code>ensemble.ExtraTreesRegressor([n_estimators, ...])</code>	An extra-trees regressor.
<code>ensemble.GradientBoostingClassifier(*[, ...])</code>	Gradient Boosting for classification.
<code>ensemble.GradientBoostingRegressor(*[, ...])</code>	Gradient Boosting for regression.
<code>ensemble.IsolationForest(*[, n_estimators, ...])</code>	Isolation Forest Algorithm.
<code>ensemble.RandomForestClassifier(...)</code>	A random forest classifier.
<code>ensemble.RandomForestRegressor(...)</code>	A random forest regressor.
<code>ensemble.RandomTreesEmbedding(...)</code>	An ensemble of totally random trees.
<code>ensemble.StackingClassifier(estimators[, ...])</code>	Stack of estimators with a final classifier.
<code>ensemble.StackingRegressor(estimators[, ...])</code>	Stack of estimators with a final regressor.
<code>ensemble.VotingClassifier(estimators, *[, ...])</code>	Soft Voting/Majority Rule classifier for unfitted estimators.
<code>ensemble.VotingRegressor(estimators, *[, ...])</code>	Prediction voting regressor for unfitted estimators.
<code>ensemble.HistGradientBoostingRegressor(...)</code>	Histogram-based Gradient Boosting Regression Tree.
<code>ensemble.HistGradientBoostingClassifier(...)</code>	Histogram-based Gradient Boosting Classification Tree.

# Ensembles

Condições necessárias para um bom desempenho

- **Diversidade**

- Classificadores base devem ser independentes
  - Ideal: cometer erros diferentes

- **Acurácia**

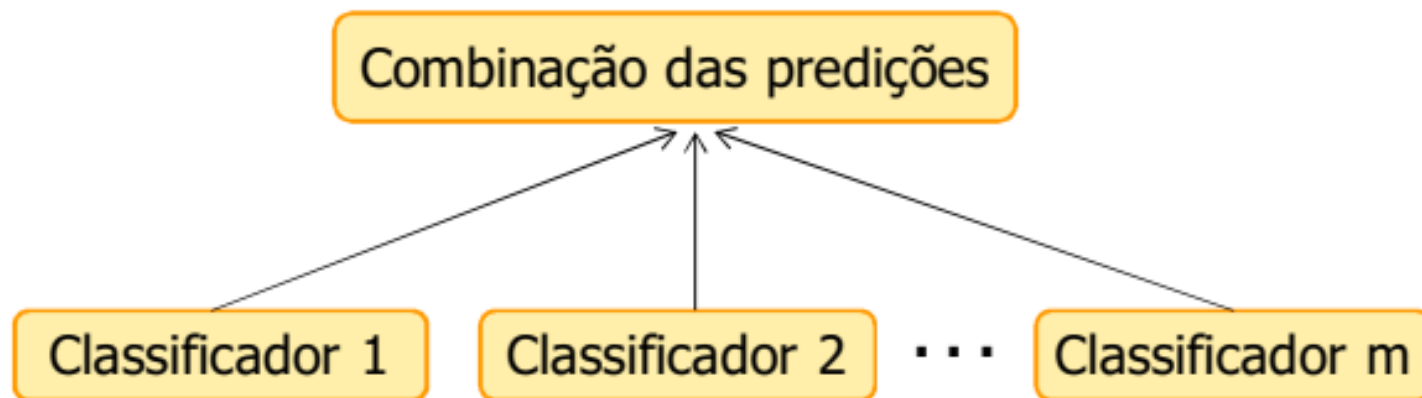
- Desempenho dos classificadores base deve ser melhor que classificação aleatória

# Ensembles

- Sejam 3 classificadores induzidos para os mesmos dados, com acurácia 0.6
- Se eles cometem os mesmos erros
  - Acurácia do ensemble será 0.6
- Se eles são completamente independentes
  - Ensembles erra classificação apenas se pelo menos 2 classificadores erram na predição

# Ensembles

- Combinação de previsões
  - Voto (média)
  - Voto (média) ponderado



# Bagging

- Cada classificador é induzido por uma amostra diferente do conjunto de treinamento
  - Mesmo tamanho do conjunto original
  - Usa *bootstrap*
- Classe definida por votação
- Tende a reduzir variância associada com classificadores base



# Técnica de amostragem Bootstrap

- No método **bootstrap**,  $r$  subconjuntos de treinamento são gerados a partir do conjunto de exemplos original. Os exemplos são **amostrados aleatoriamente** desse conjunto, **com reposição**. Logo um exemplo pode estar presente em um determinado subconjunto de treinamento mais de uma vez.
- Os **subconjuntos de testes** são formados pelas sequências que **não estão no conjunto de treino**  
Cerca de um terço das instâncias são deixados de fora da amostra de bootstrap e não são usados na construção da k-árvore.

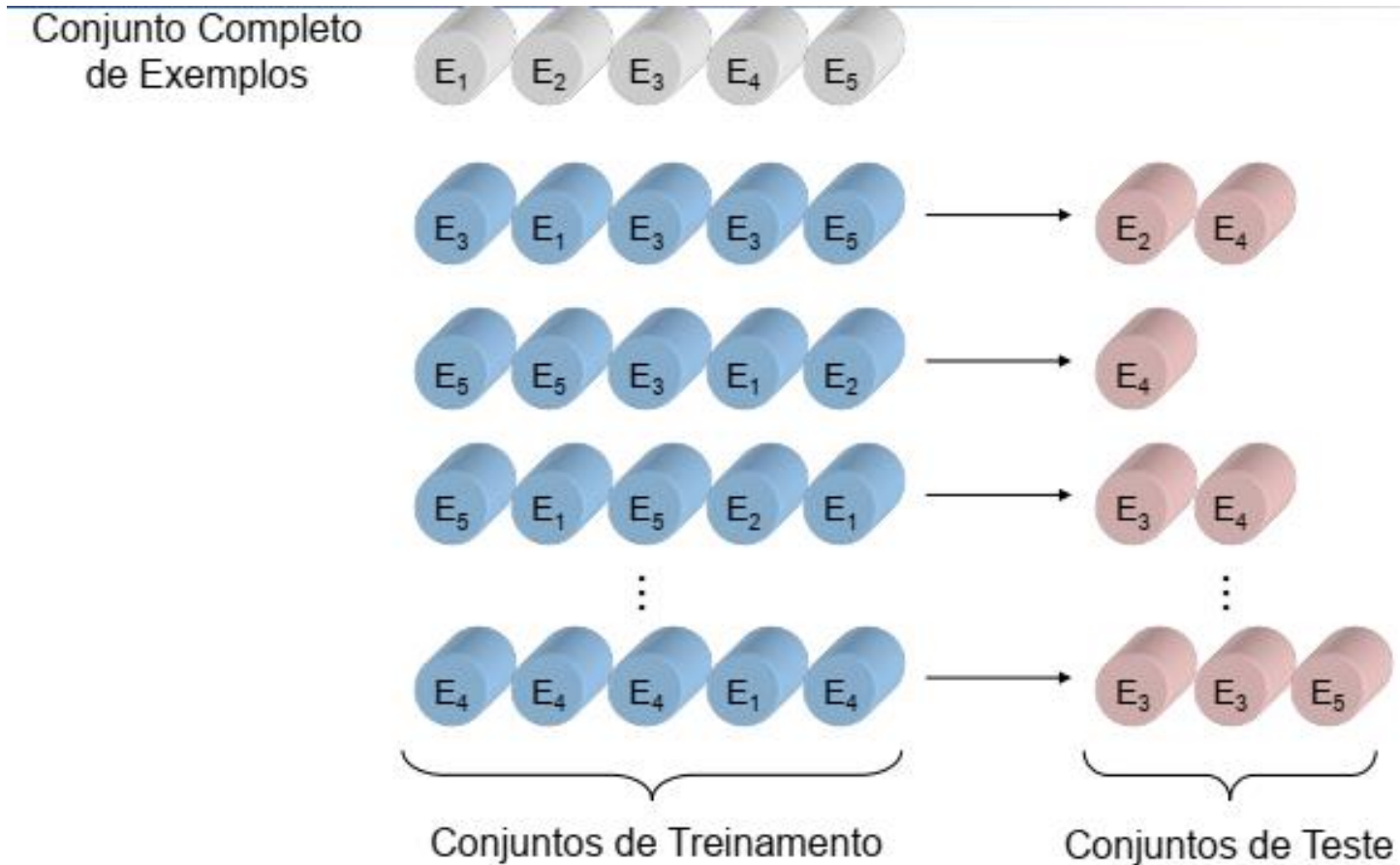
# Técnica de amostragem Bootstrap

Normalmente adota-se  $r \geq 100$ . A ideia básica é repetir o experimento um número alto de vezes e estimar o desempenho nesses experimentos replicados.

Por este motivo, o bootstrap é um procedimento custoso.

Há vários estimadores bootstrap, e o mais comum é o  $e_0$ . Neste cada conjunto de treinamento tem  $n$  exemplos, amostrados com reposição do conjunto original, sendo  $n$  o número total de exemplos nesse conjunto

# Técnica de amostragem Bootstrap



# Técnica de amostragem Bootstrap

## Como se avalia o resultado final?

- O resultado final é dado pela média do desempenho observado em cada subconjunto de teste.

# Bagging

- Seja o conjunto de dados de treinamento  $\{x_1, x_2, x_3, x_4, x_5, x_6\}$

$x_1, x_6, x_3, x_5, x_3, x_1$

Amostra 1

$x_3, x_4, x_1, x_5, x_5, x_1$

Amostra 2

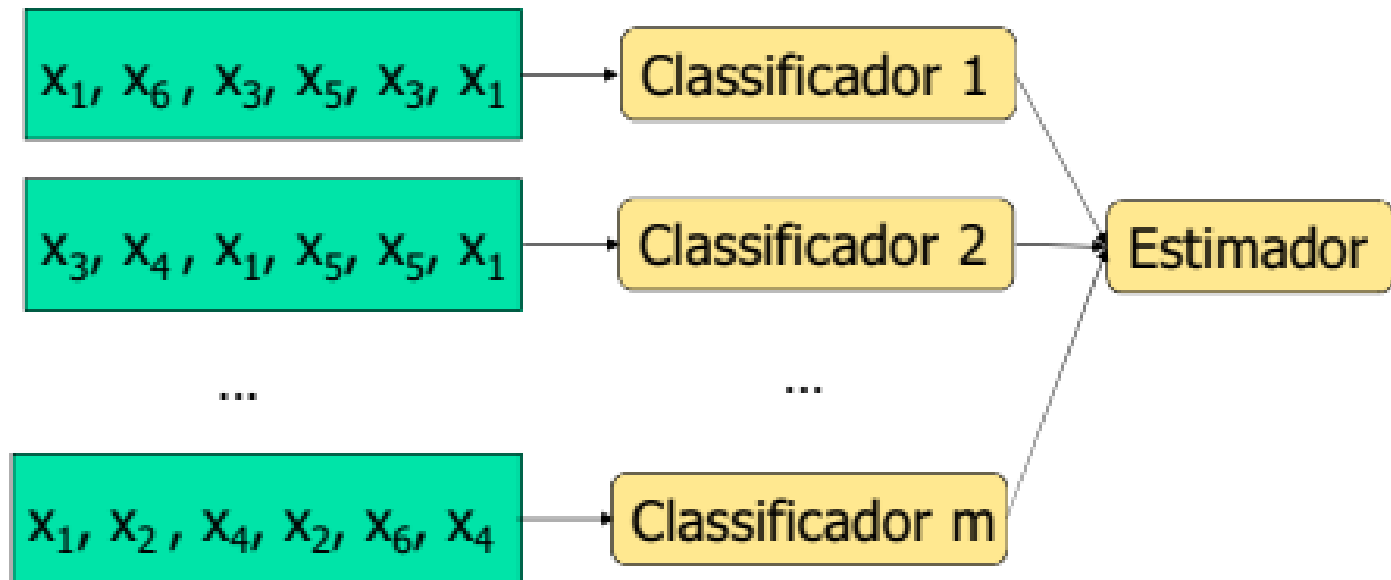
...

...

$x_1, x_2, x_4, x_2, x_6, x_4$

Amostra m

# Bagging



# Boosting

- Conjunto de técnicas
  - Adaboost é uma das mais conhecidas
- A cada iteração
  - Induz classificador
  - Pondera cada exemplo do conjunto de dados completo pelo desempenho do classificador base
    - Quanto mais difícil de ser aprendido, maior o peso associado ao exemplo
- Boosting funciona de forma semelhante a minimização por gradiente descendente

# Boosting

Seja o conjunto de dados de treinamento  $\{x_1, x_2, x_3, x_4, x_5\}$

Exemplos:	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	Soma dos pesos = 1.0  C: correta I: incorreta
Pesos atuais:	0.2	0.2	0.2	0.2	0.2	
Classificação:	C	I	C	C	I	
Novos pesos:	0.2	0.4	0.2	0.2	0.4	

Exemplos:	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
Pesos atuais:	0.2	0.4	0.2	0.2	0.4
Classificação:	C	I	C	I	C
Novos pesos:	0.2	0.6	0.2	0.4	0.4

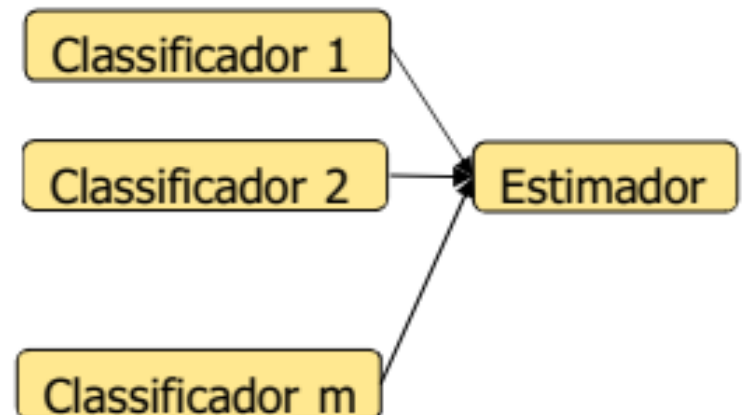


# Boosting

- Indicado para classificadores base fracos
  - Acurácia ligeiramente melhor que palpite aleatório
- Convergência rápida
- Pouco indicado para dados com ruídos e pequenos conjuntos de dados
  - Por focar em exemplos difíceis de serem classificados

# Stacking

- Um algoritmo estimador aprende a combinar previsões de modelos base
  - Modelos gerados por algoritmos base
  - Saídas combinadas por algoritmo estimador
    - Algoritmo de AM
- Algoritmos base podem ser:
  - Homogêneos
  - Heterogêneos



# Ensembles de Árvores de decisão

- Combina a predição de várias árvores de decisão
- Duas principais abordagens:
  - Extreme Gradient Boosting
  - Random Forest

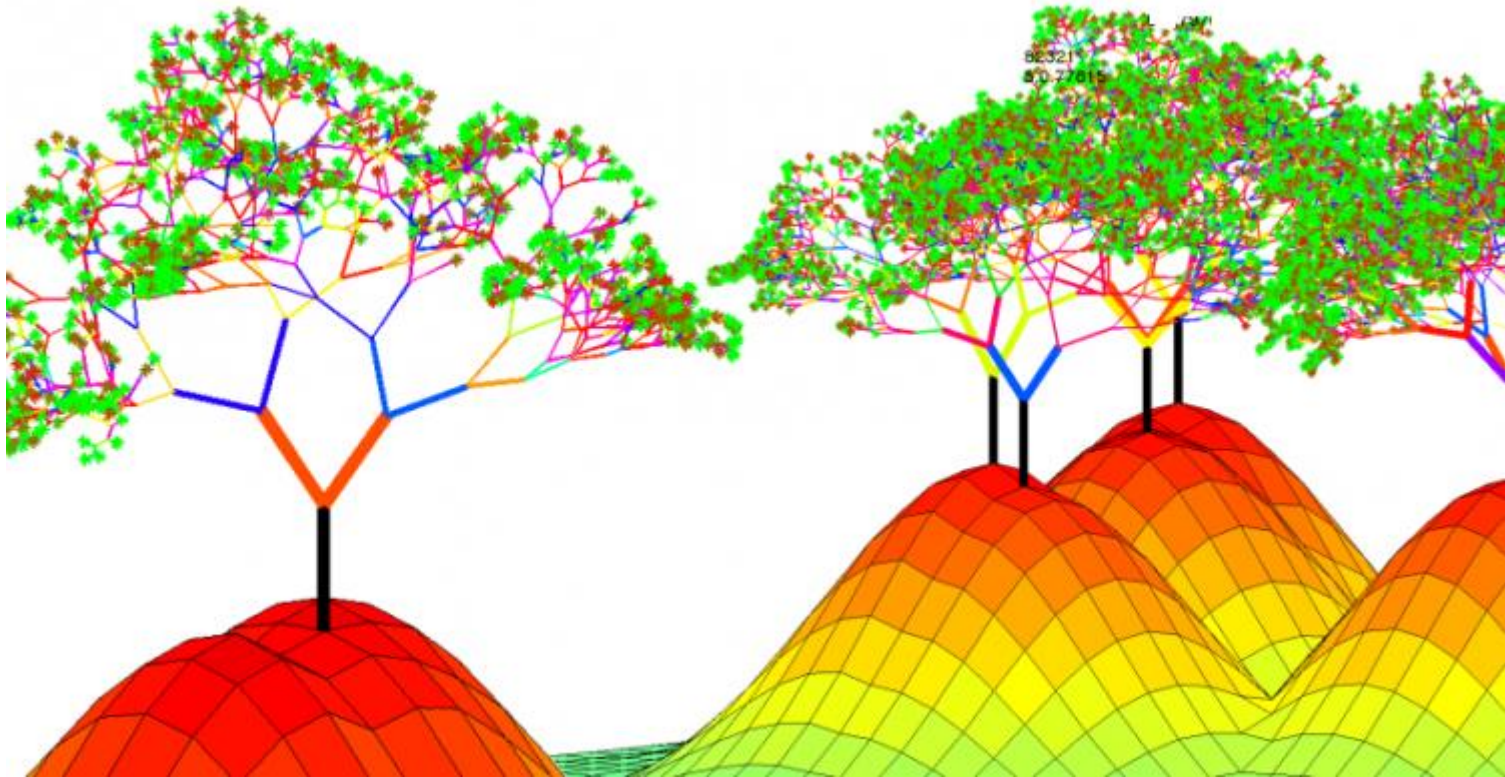
# Extreme Gradiente Boosting

- XGBoost
- Combina árvores geradas pelo algoritmo CART
- Pondera a resposta de cada árvore para reduzir complexidade do modelo final

# Random Forest - Por quê Forests? Muitas árvores!

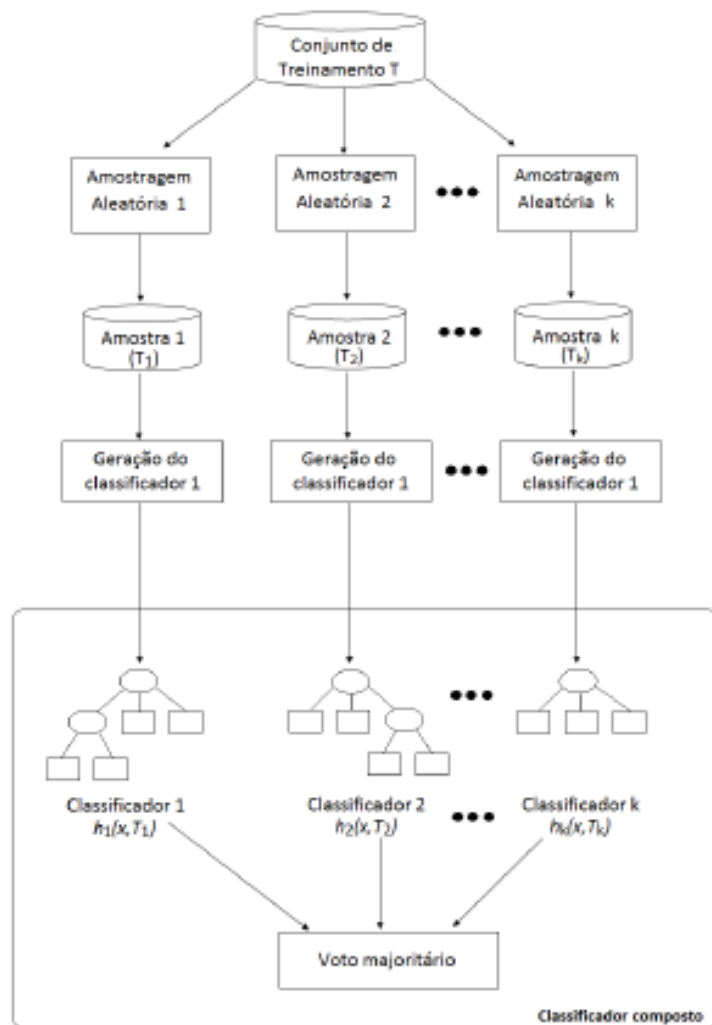


Por quê Random **Forests**?  
Muitas árvores...

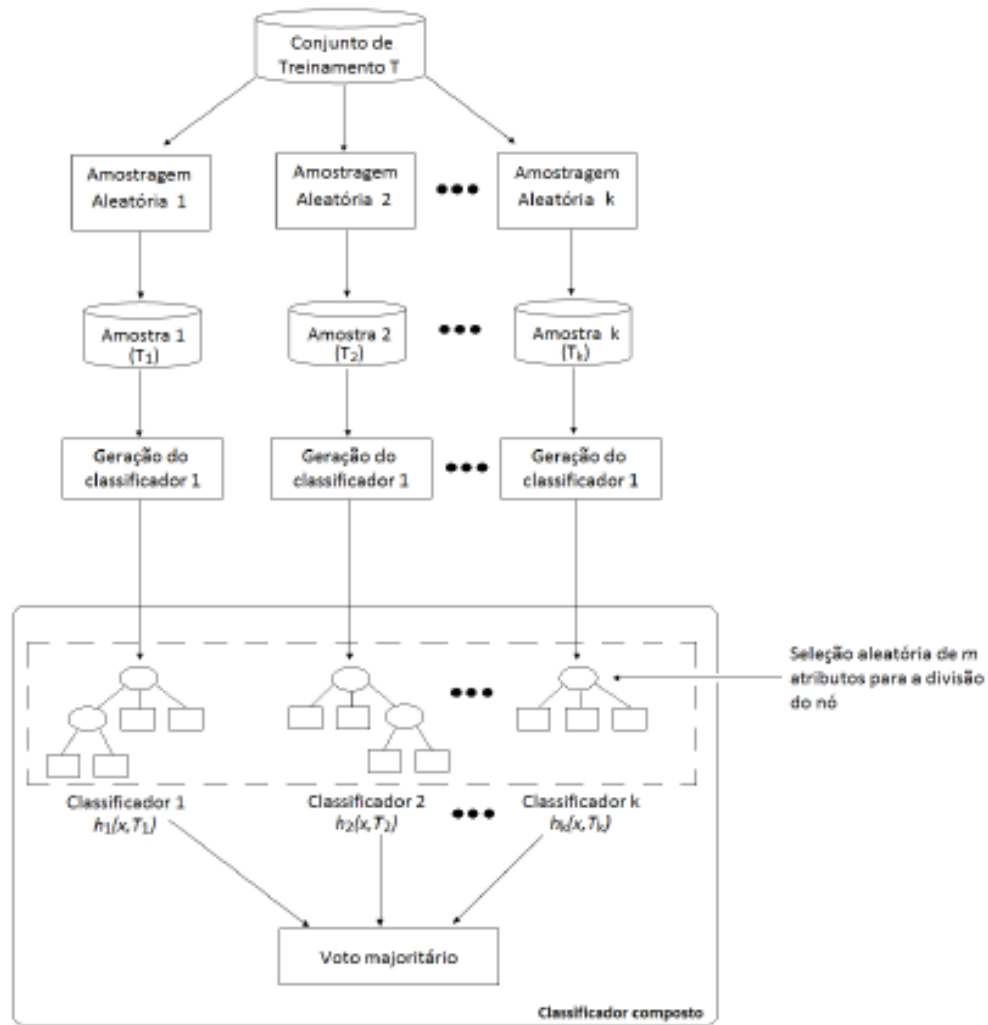


# Random Forests

- É um algoritmo ensemble proposto por Breiman (veja o artigo **Random Forests**)<sup>1</sup> que constrói muitas árvores de decisão as quais são utilizadas para classificar um novo exemplo por meio do voto majoritário
- Ou seja, o algoritmo de Random Forest (RF) é um termo geral para métodos de ensemble utilizando classificadores do tipo árvore
- Utiliza a amostragem **Bootstrap**



Funcionamento do Bagging



Funcionamento do Random Forest



# The Random Forest Algorithm

1. For  $b = 1$  to  $B$ :
  - (a) Draw a **bootstrap sample**  $\mathbf{Z}^*$  of size  $N$  from the training data.
  - (b) Grow a random-forest tree  $T_b$  to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size  $n_{min}$  is reached.
    - i. Select  **$m$  variables at random** from the  $p$  variables.
    - ii. Pick the best variable/split-point among the  $m$ .
    - iii. Split the node into two daughter nodes.
2. Output the ensemble of trees  $\{T_b\}_1^B$ .

To make a prediction at a new point  $x$ :

*Regression:*  $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x)$ .

*Classification:* Let  $\hat{C}_b(x)$  be the class prediction of the  $b$ th random-forest tree. Then  $\hat{C}_{\text{rf}}^B(x) = \text{majority vote } \{\hat{C}_b(x)\}_1^B$ .

# Algoritmo Random Forest

- Treina cada árvore com amostras geradas a partir do método de amostragem *bootstrap*
- Para cada conjunto de instâncias, o Random Forest considera somente  $m$  variáveis selecionadas aleatoriamente do conjunto de dados
- Random Forest não faz poda

O resultado final obtido a partir das árvores é dado por:

- Para problemas de **classificação**: voto majoritário
- Para problemas de **regressão**: Média dos valores preditos

# Desvantagem da Random Forest

- Difícil extrair o conhecimento das árvores, apesar de o método exibir os atributos mais relevantes

Verificar no Weka como este método funciona!

# Como definir o número de atributos a serem utilizados?

- Alguns métodos utilizam **raiz quadrada** da quantidade de atributos
- Outros utilizam o log da quantidade de atributos

# Comandos para Random Forest, em Python

**Random Forest:** `from` sklearn.ensemble `import` RandomForestClassifier

```
modelo = RandomForestClassifier(n_estimators=10, max_features=3, criterion='gini', random_state = 0)
modelo.fit(X_treino, y_treino)
```

**Lembrar de ajustar os hyperparâmetros:**

<https://scikit-learn.org/stable/modules/classes.html#module-sklearn.ensemble>

## Hyper-parameter optimizers

<code>model_selection.GridSearchCV(estimator, ...)</code>	Exhaustive search over specified parameter values for an estimator.
<code>model_selection.HalvingGridSearchCV(...[, ...])</code>	Search over specified parameter values with successive halving.
<code>model_selection.ParameterGrid(param_grid)</code>	Grid of parameters with a discrete number of values for each.
<code>model_selection.ParameterSampler(...[, ...])</code>	Generator on parameters sampled from given distributions.
<code>model_selection.RandomizedSearchCV(...[, ...])</code>	Randomized search on hyper parameters.
<code>model_selection.HalvingRandomSearchCV(...[, ...])</code>	Randomized search on hyper parameters.

# Referências

Breiman, L. **Random Forest**. Machine Learning (2001) 45: 5.

<https://doi.org/10.1023/A:1010933404324>

<https://link.springer.com/article/10.1023%2FA%3A1010933404324>

<https://edisciplinas.usp.br/course/view.php?id=78145>

[https://www.stat.berkeley.edu/~breiman/RandomForests/cc\\_home.htm#workings](https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#workings)

**Verificar sobre os parâmetros default**

<https://medium.com/turo-engineering/how-not-to-use-random-forest-265a19a68576>