Random Forests

Inteligência Artificial
PCS3438

Anna Helena Reali Costa
Escola Politécnica da USP
Engenharia de Computação (PCS)

Bootstrap

- **Bootstrapping** is any test or metric that depends on **random sampling with replacement**.
- Bootstrapping allows **estimating a sampling distribution** of almost any statistic.
- Generally, it falls into the broadest class of **resampling methods.**

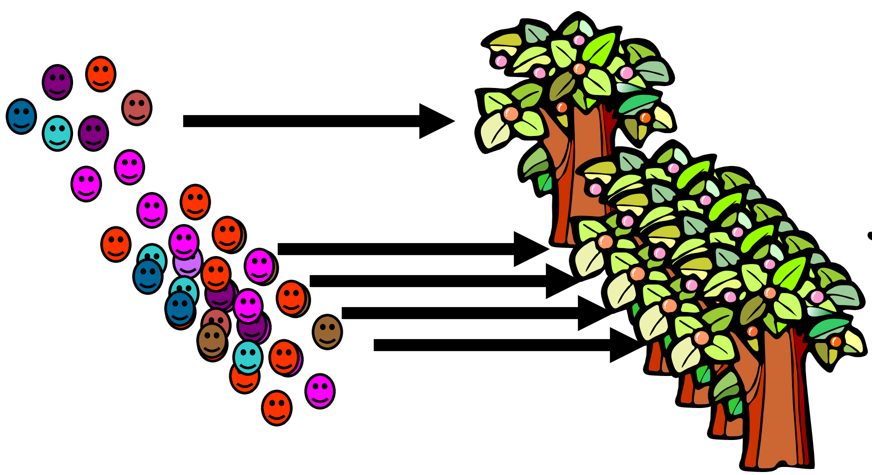
Bagging

A name derived from "**bootstrap aggregation**".

Bagging decision trees: builds multiple decision trees by repeatedly resampling training data with replacement, and voting the trees for a consensus prediction.

Bagging decision trees

- Select a bootstrap sample, L_B from L.
- Grow a decision tree from L_B. Repeat B times.



$$\hat{f}_1(x), \hat{f}_2(x), \hat{f}_3(x), \cdots \hat{f}_B(x)$$

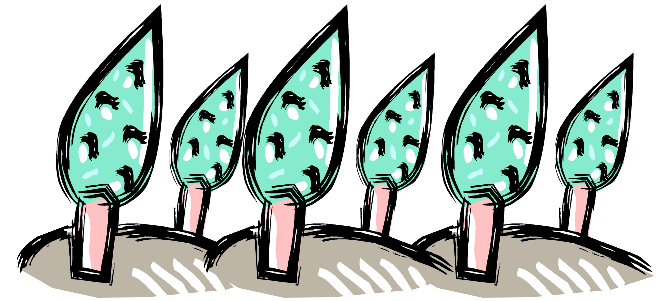
- Estimate the class of x_n:
  - Majority vote (Classification Tree)

Bagging decision trees

- Leads to better model performance because it decreases the variance of the model, without increasing the bias.
  - While the predictions of a single tree are highly sensitive to noise in its training set, the average of many trees is not, as long as the trees are not correlated.
  - Bootstrap sampling is a way of de-correlating the trees by showing them different training sets.
- The number of samples/trees, B, is a free parameter.

Random Forests

Random Forests



- It is a <u>specific</u> type of bagging
- As in bagging, we build a number of decision trees on bootstrapped training examples
- But when building these decision trees, each time a split in a tree is considered, **a random sample of m<p predictors is chosen** as split candidates from the full set of **p** predictors ("feature bagging")
- A fresh sample of **m** predictors is taken at each split (in bagging, m = p):
  - typically $m = p^{-1/2}$ for CT and m = p/3 for RT.

Boosting

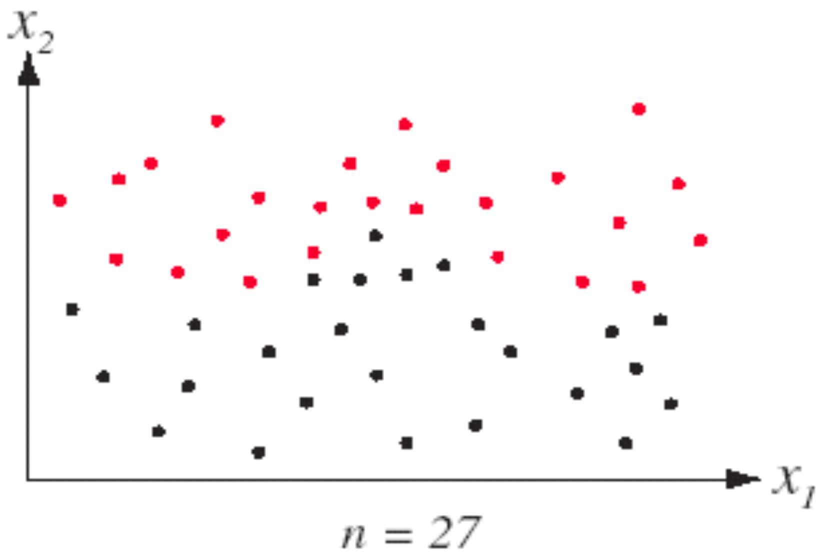
Boosting

- Boosting is an **ensemble technique** that attempts to create a strong classifier from a number of weak classifiers.
- This is done by building a model from the training data, then creating a second model that attempts to correct the errors from the first model. Models are added until the training set is predicted perfectly or a maximum number of models are added.

Boosting

- Similar to bagging, except that the trees are grown <u>sequentially</u>: each tree is grown using information from previously grown trees.
- Boosting does not involve bootstrap sampling.
- Form an ensemble whose joint decision rule has higher accuracy.

Basic procedure
(using a linear classifier)



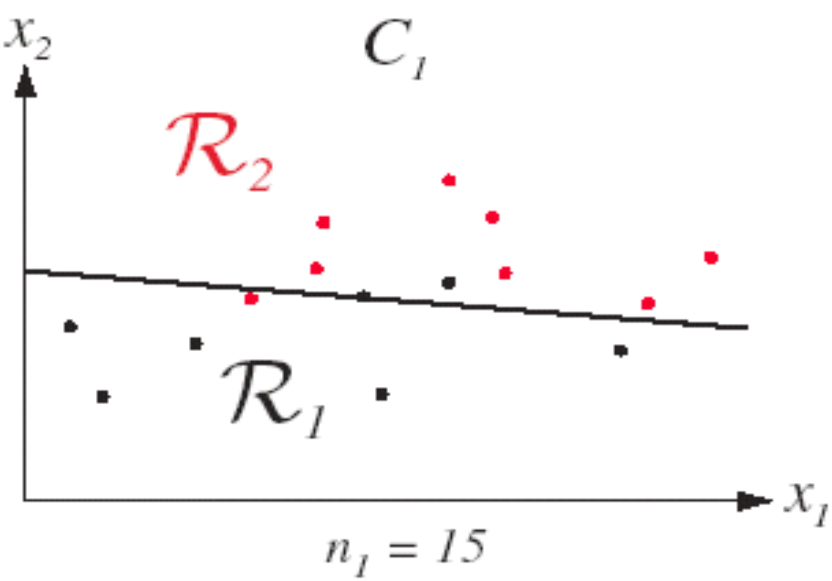
Problem

- 2-dimensional
- 2-category

Final classification

- Voting of 3 component classifiers

Train classifier C_1 with D_1

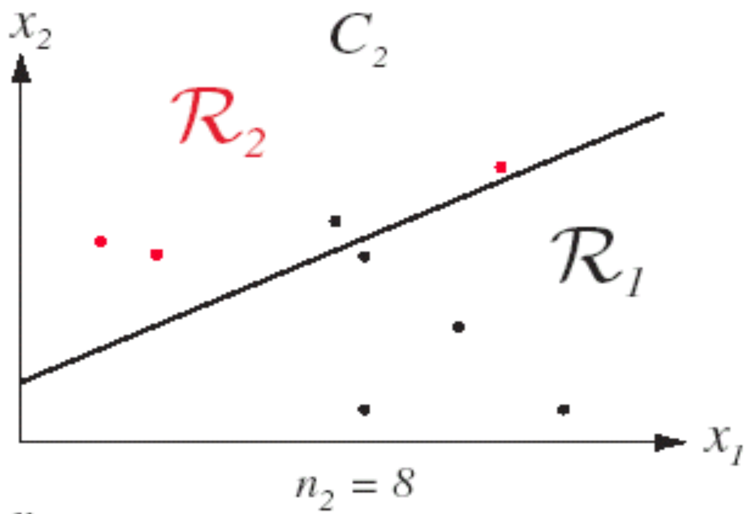


Randomly select a subset of samples from the training set.

Train the first classifier C_1 with this subset.

Classifier C_1 is a weak classifier.

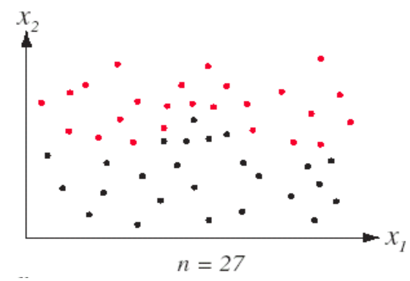
Train classifier C_2 with D_2



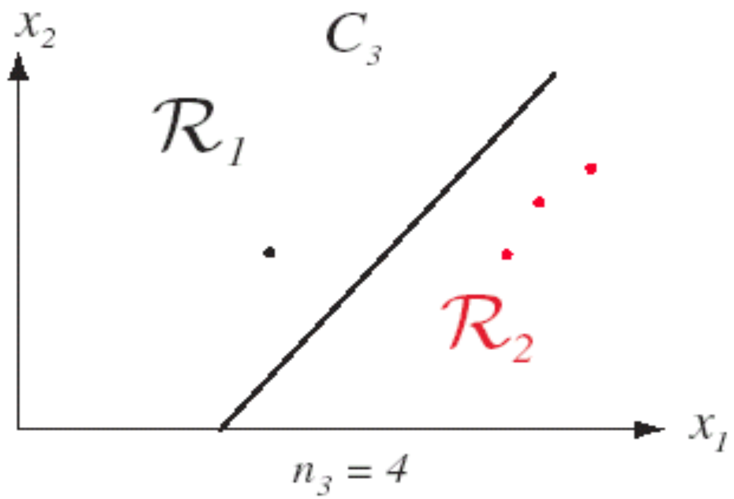
Find a second training set that is the "most informative" given C_1:

- ½ should be correctly classified by C_1.
- ½ should be incorrectly classified by C_1.

Train a second classifier, C_2, with this set.

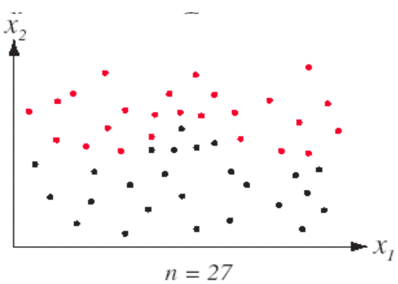


Train classifier C_3 with D_3

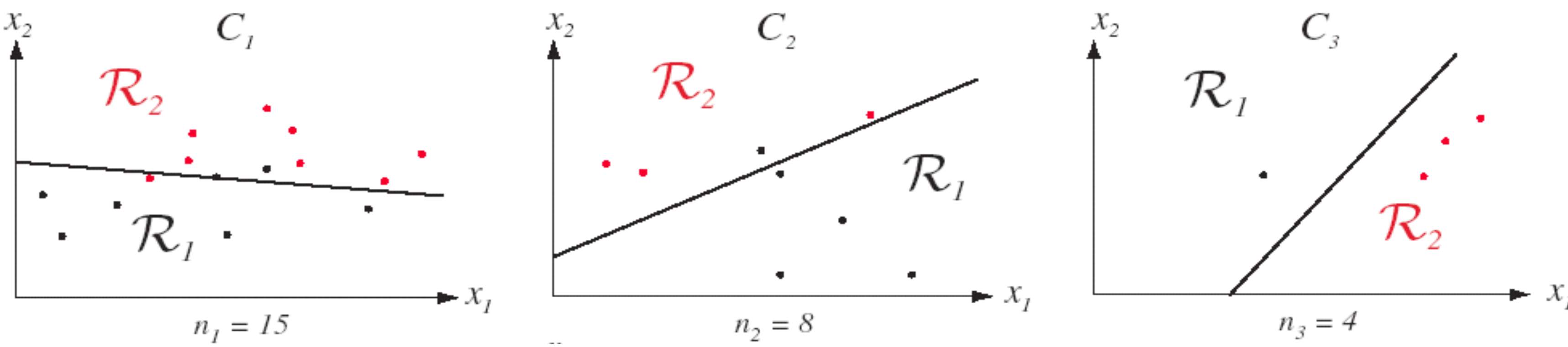


Seek a third data set which is not well classified by voting by C_1 and C_2.

Train the third classifier, C_3, with this set.

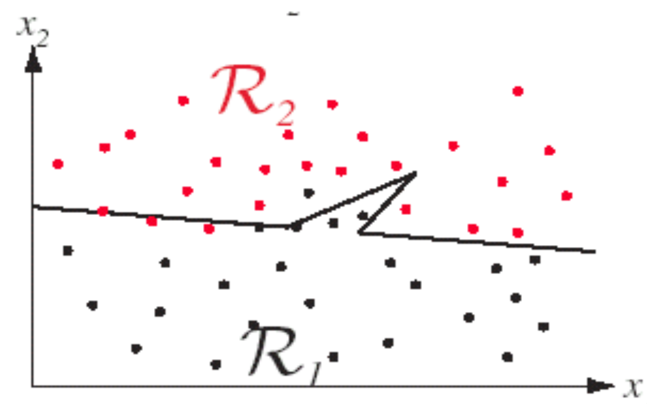


Ensemble of classifiers



Classifying a test pattern based on votes:

- If C_1 and C_2 agree on a label → use that label.
- If disagree → use the label given by C_3.



Boosting

- You are given $\{(x_1,y_1), (x_2,y_2), \ldots (x_n,y_n)\}$, and the task is to fit a model f(x) to minimize error.
- Suppose your friend wants to help you and gives you a model f.
- You check his model and find the model is good but not perfect. There are some mistakes: $f(x_1) \neq y_1$ and $f(x_2) \neq y_2$;
- How can you improve this model? But: you are not allowed to remove anything from f or change any parameter in f.

*: from http://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf

Boosting

- You can add an additional model (classification tree) h to f, so the new prediction will be f(x) + h(x).
- You wish to improve the model such that:

$$f(x_1) + h(x_1) = y_{1\,;} \qquad f(x_2) + h(x_2) = y_{2;\,\ldots.}$$
$$f(x_n) + h(x_n) = y_n$$

Or, equivalently, you wish:

$$h(x_1) = y_1 - f(x_1)\ ; \qquad h(x_2) = y_2 - f(x_2);\ \ldots$$
$$h(x_n) = y_n - f(x_n)$$

AdaBoost

- AdaBoost, short for ***<u>Adaptive Boosting</u>***, is a **machine learning meta-algorithm** formulated by Yoav Freund and Robert Schapire, who won the **2003 Gödel Prize** for their work.
- It can be used in conjunction with many other types of learning algorithms to improve performance.
- The output of the other learning algorithms ('weak learners') is combined into a **weighted sum that represents the final output** of the boosted classifier.

AdaBoost

- AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those **instances misclassified by previous** classifiers.
- AdaBoost is sensitive to **noisy data** and **outliers**.
- In some problems it can be **less** susceptible to the **overfitting** problem than other learning algorithms.
- The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing, the final model can be proven to converge to a strong learner.

References

- Mitchell, T.M. Machine learning. McGraw-Hill, 1997. Cap. 3
- James, G.; Witten, D.; Hastie, T; Tibshirani, R. An introduction to statistical learning. Springer, 2013. Cap. 8.
- www.wisdom.weizmann.ac.il/~vision/courses/2003_2/**multipleTrees.ppt**
- www.cse.lehigh.edu/~munoz/CSE497/classes/**Storey_DecisionTrees.ppt**
- http://www.saedsayad.com/decision_tree.htm
- https://class.stanford.edu/c4x/HumanitiesScience/StatLearning/asset/trees.pdf
- https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/