Machine Learning Techniques

DATASCI 420

Lesson 06-02 Ensemble Models: Random Forest

Lecture Outline

- Assignment 1 Discussion
- Review:
  - Underfitting and Overfitting
  - Decision Tree
- Ensembles, Random Forests
- Break
- Lecture 4 Preview
  - Data Science Modelling
  -*Model performance evaluation…*

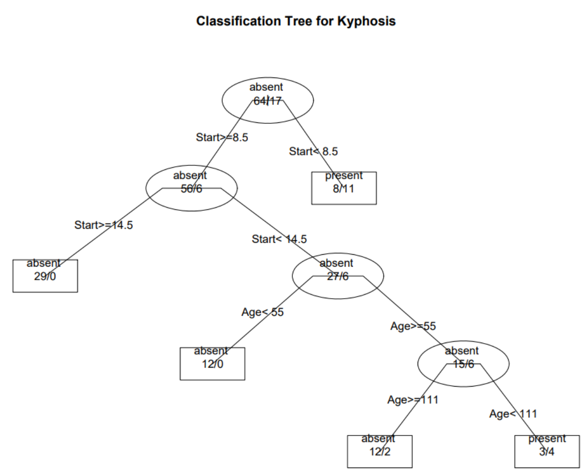
Answers to Some Frequently Asked Questions

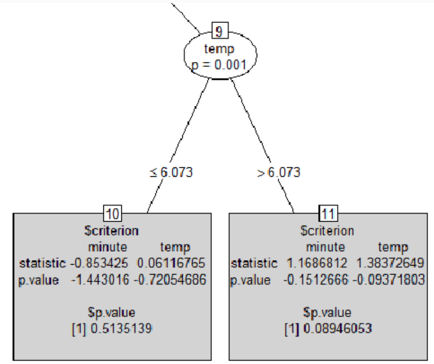
- Q: Why my decision tree only splits four variables and stops splitting?
  - A: Each decision tree algorithm implementation has some default settings to prevent overfitting. For instance in R:
- Q: How to interpret my decision tree?
  - A: You can develop a set of rules from the decision tree. Check all the conditions (path from a root node to a leaf node), and the majority voting in the leaf node.
- Q: What does the p-value mean in ctree of the partykit library in R?
  - A: In this ctree model, to determine which variable to split, it is doing some statistical testing for statistical independence hypothesis: $H_0^i : D(Y \mid X_i) = D(Y)$

    The lower p-value, the stronger statistical dependence between Y and X.

rpart(*formula***, data=, method=,control=)** where

formula	is in the format *outcome ~ predictor1+predictor2+predictor3+*ect.
data=	specifies the data frame
method=	**"class"** for a classification tree **"anova"** for a regression tree
control=	optional parameters for controlling tree growth. For example, control=rpart.control(minsplit=30, cp=0.001) requires that the minimum number of observations in a node be 30 before attempting a split and that a split must decrease the overall lack of fit by a factor of 0.001 (cost complexity factor) before being attempted.





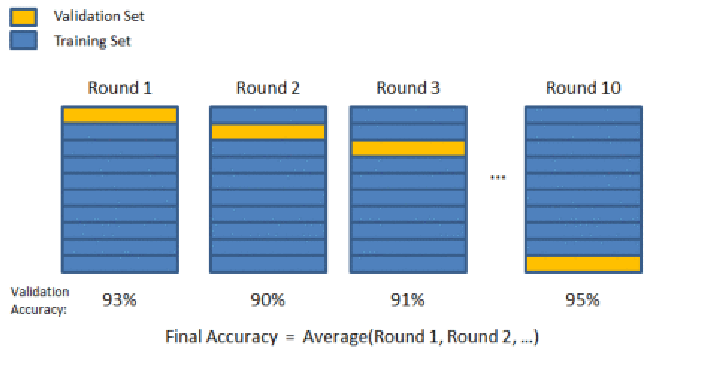
Common Mistakes/Flaws Observed in Submissions

- Submitted Model Performance on Training Set
  - What matters most is the performance on Validation Set.
  - It is always a good habit to check the performance on both training and validation sets, in order to get a sense of overfitting

- ROC Curve
  - Is ROC curve a good performance measurement for this problem?
    - ROC (or Area Under ROC Curve (AUC)) is useful only when ranking matters.
    - ROC is only applicable for binary classification.
  - ROC curve should have two inputs: a column of ground truth and a column of predicted label. Not anything else. Do no calculate ROC of two predictor variables.

Common Pitfalls in Machine Learning

- Overfitting
  - Split the data into training and validation, and only care about the performance on validation
  - Cross validation.

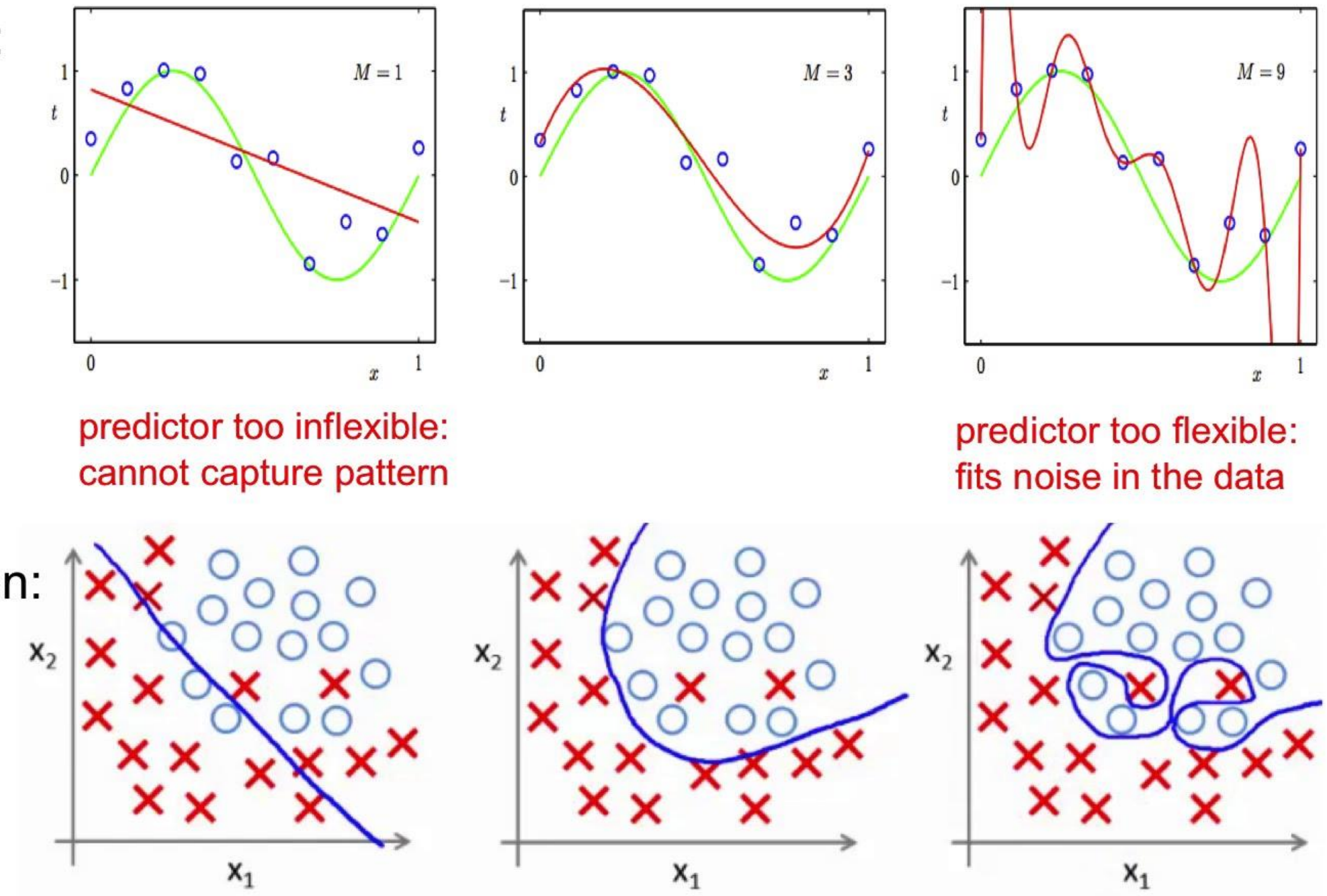


- Target leaking:
  - Predicting readmission. You have one binary variable “readmission”, which is your target column. You also have columns “readmission time”, “readmission location”, “readmission reason”.
- Model has good performance on validation, but not applicable

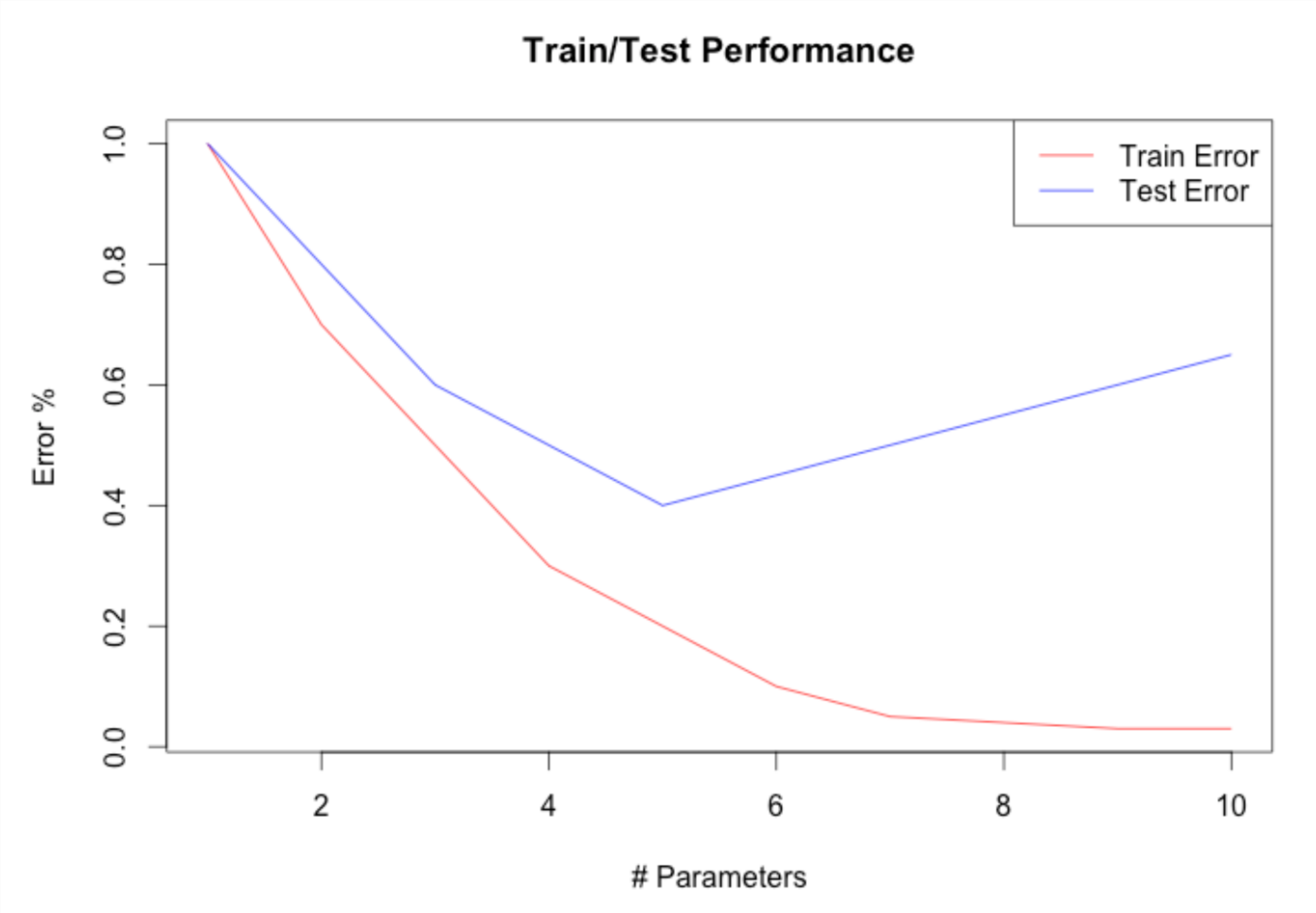
Under- and Over-fitting examples

Regression:

Classification:



Indicators of Underfitting and Overfitting



- Model performs poorly on both training and testing data
  - Underfitting, or
  - Not relevant data
- Model performs well on training, but poorly on testing
  - Overfitting

Reducing Underfitting

- Increase model complexity, for e.g.
  - Increase the number of levels in a decision tree
  - Increase the number of hidden layers in a neural network.
  - Decrease the number of neighbors (k) in k-NN
- Increase the number of features, or create more relevant features
- In iterative training algorithms, iterate long enough so that the objective function has converged.

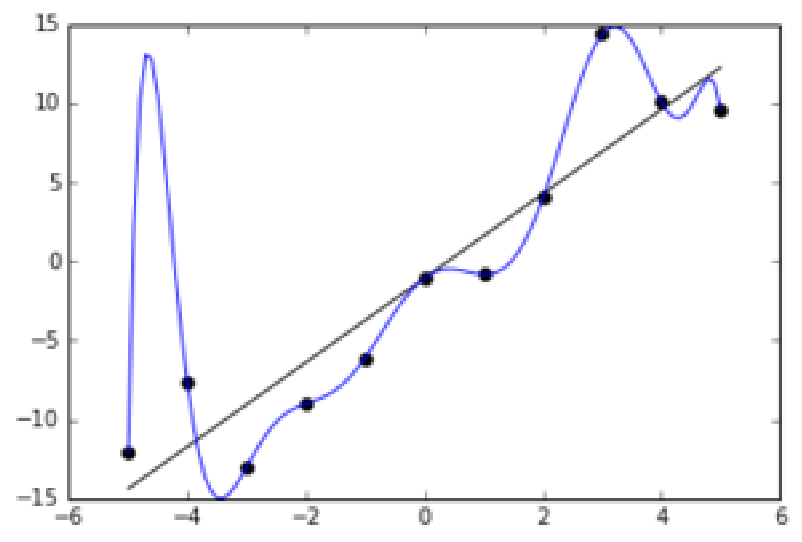
Reducing Overfitting

- Decrease model complexity, for e.g.
  - Prune a decision tree
  - Reduce the number of hidden layers in a neural network.
  - Increase the number of neighbors (k) in k-NN
- Decrease the number of features
  - More aggressive feature selection
- Regularization (control feature complexity)
  - Penalize high weights.
  - L-1 regularization (LASSO) very efficient at pushing weights of non-informative features to 0.
- Gather more training data if possible
- In iterative training algorithms, stop training earlier to prevent "memorization" of training data

Regularization: A Popular Way of Controlling Overfitting

- Loss Function of Training
  - You can almost always increase the complexity of f_θ to reduce SSE
  - Increase the risk of overfitting
- Add regularization to control overfitting
  - L1 (LASSO) or L2 (Ridge regression) regularization

$$SSE = \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2$$



$$LOSS = \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2 + \lambda_1 \sum_{k=1}^{m} |\theta_k| + \lambda_2 \sum_{k=1}^{m} \theta_k^2$$

$$\lambda_1, \lambda_2 \geq 0$$

$$\lambda_1 = 0, \lambda_2 > 0 : \text{Ridge regression}$$

$$\lambda_2 = 0, \lambda_1 > 0 : \text{LASSO}$$

$$\lambda_1, \lambda_2 > 0 : \text{Elastic net}$$

What to remember about classifiers

- Try simple classifiers first
- Better to have smart features and simple classifiers than simple features and smart classifiers
- Use increasingly powerful classifiers with more training data

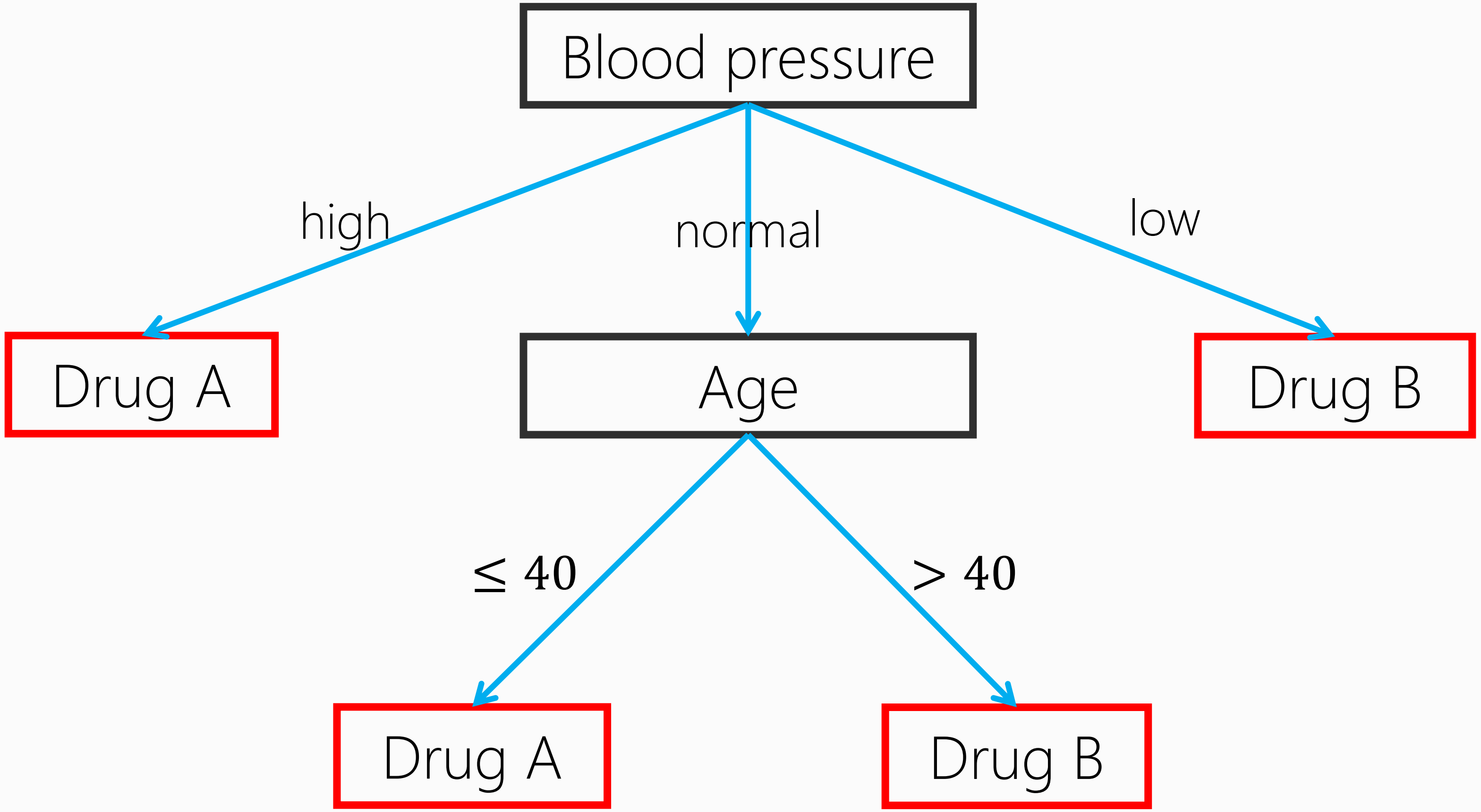
Review: Decision Tree

Unique Features

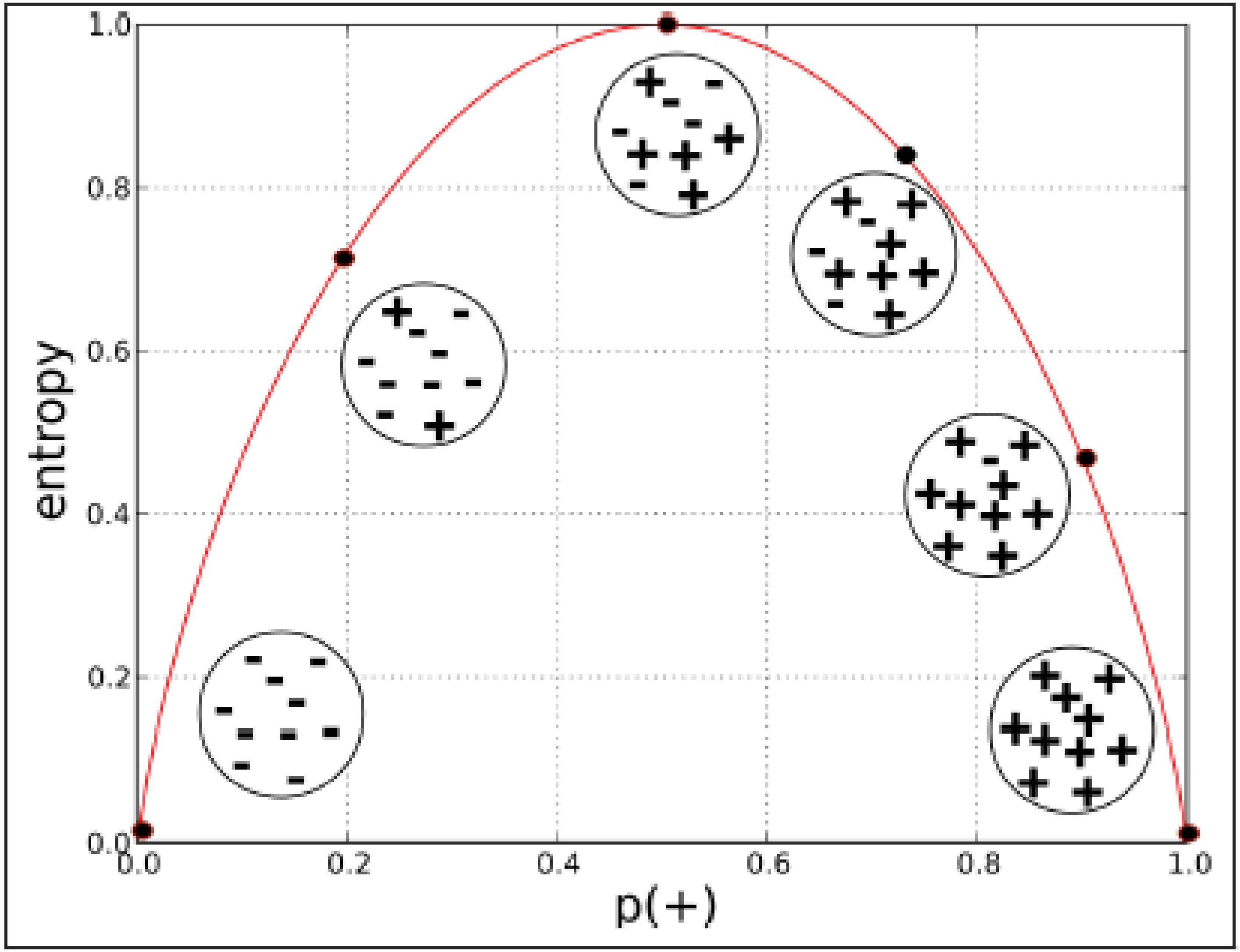
1. Automatically selects features
2. Able to handle large number of features
3. Numeric, nominal, missing
4. Easy to ensemble (Random Forrest, Boosted DT)
5. Transparent and easily explainable☺…

Review: Decision Tree

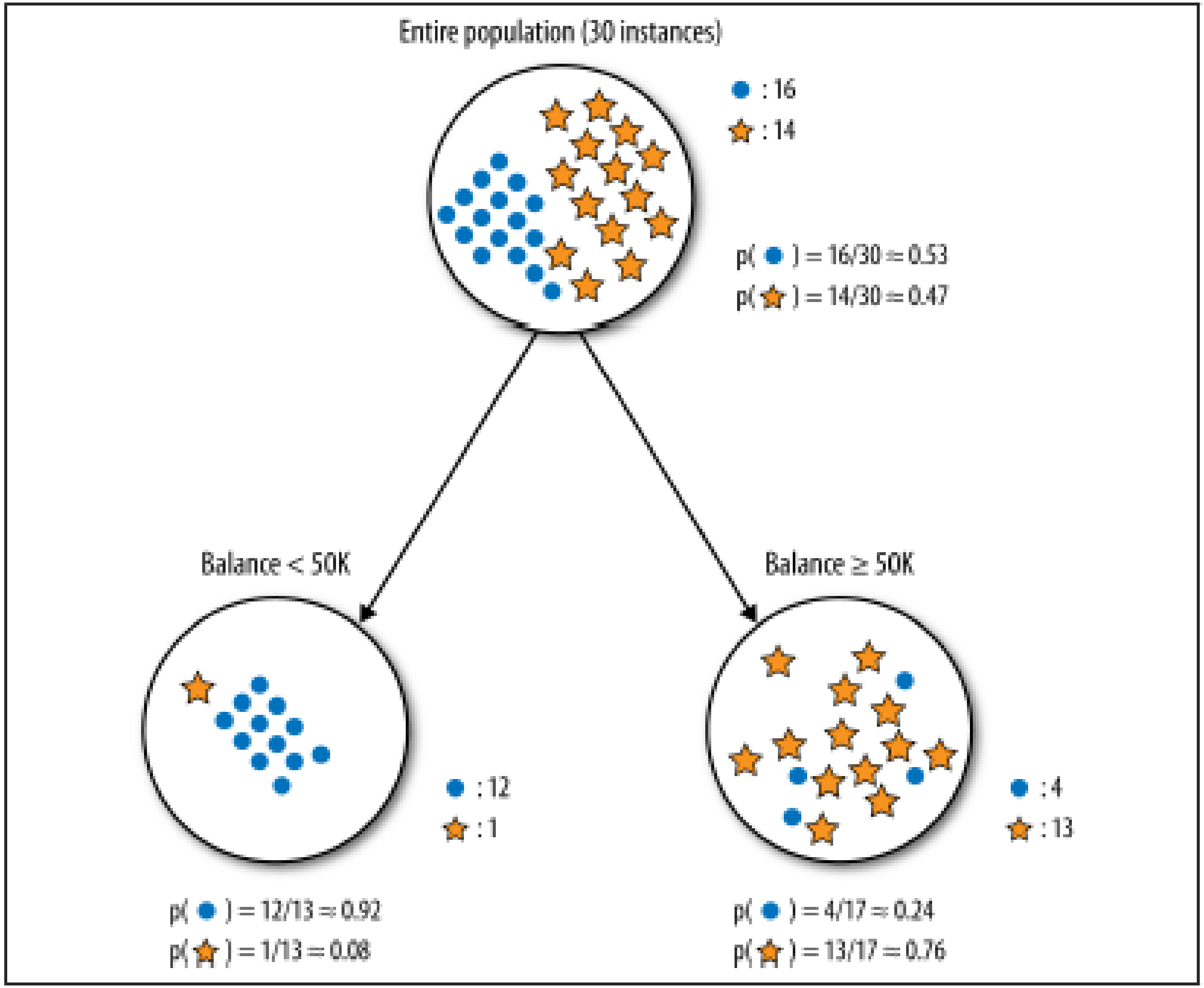
Assignment of drug to a patient



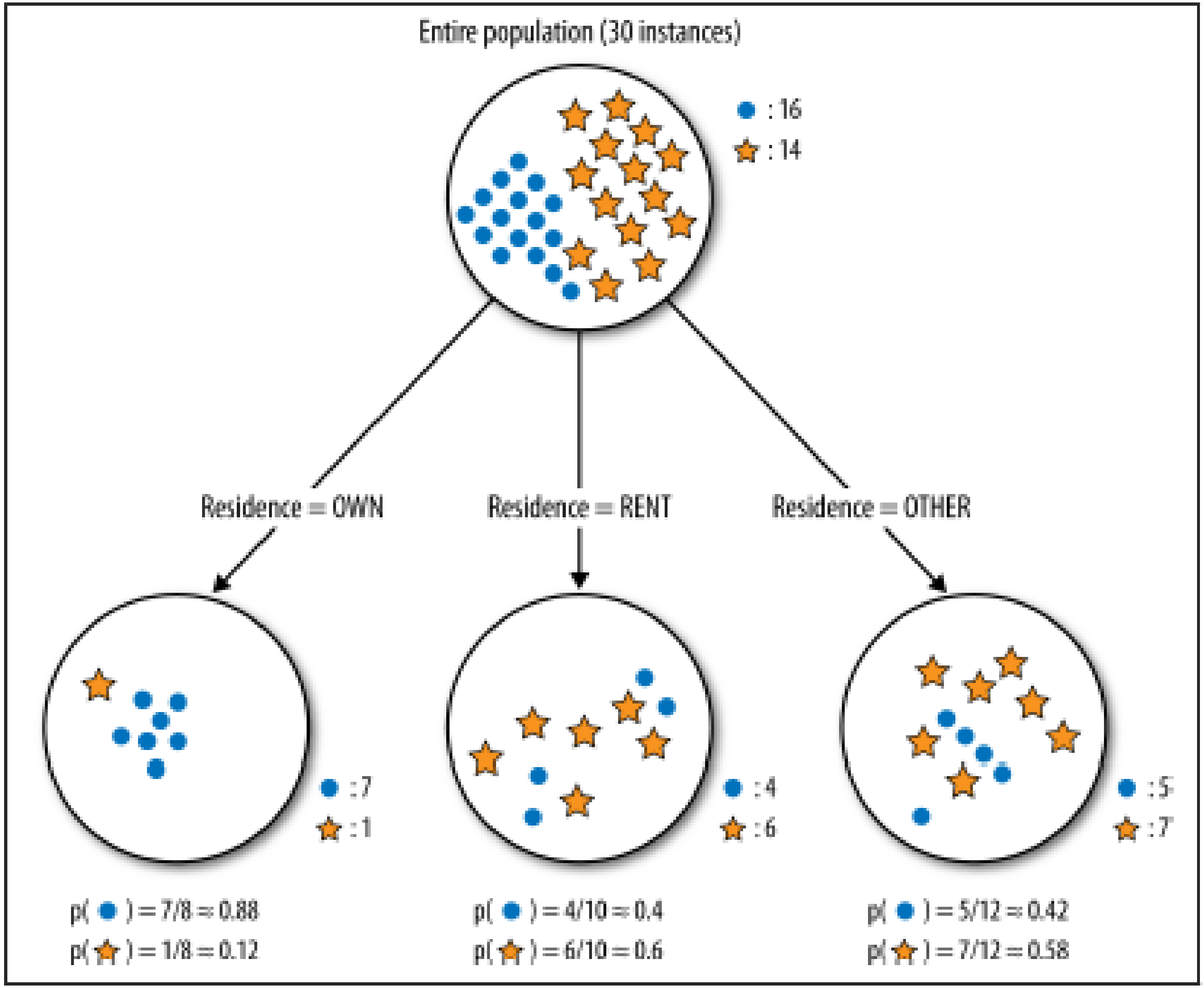
It's all about minimizing the entropy (variance)...



It's all about minimizing the entropy (variance)...



It's all about minimizing the entropy (variance)...



Review: Induction of decision trees

- Top down approach
  - Build the decision tree from top to bottom, from the root to the leaves
- Greedy selection of a test feature
  - Compute an evaluation measure for all features
  - Select the feature with the best measure
- Divide and Conquer/ Recursive Descent
  - Divide examples according to values of test feature
  - Apply the procedure recursively to the subsets
  - Terminate the recursion if
    - All cases belong to the same class, no more examples are available, cutoff condition has been satisfied (minimum node size)
    - Maximum depth of the tree has been reached

Ensemble of Models and Random Forests

Why Ensemble?

- Think about a patient with some complicated disease
  - A group (panel) of doctors are involved in diagnosis
  - Each doctor may diagnose based on a specific set of data, and/or on his own specific domain expertise (model)
  - The final diagnosis is made by majority voting, weighted average (some doctors might be more experienced, their diagnosis take higher weights than others)
- Benefits of ensemble models:
  - Usually perform better than each individual model
  - Reduce the variance in the predictions, generalize better than individual models
  - Make the process of building the machine learning solutions more scalable

Different Ways of Ensembling

- Bagging:
  - Each model is trained on a subset of observations and/or features independently
- Boosting:
  - Model i+1 is trained on a sampled subset of observations, where observations that are not classified corrected by model i have higher probability of being sampled
- Different ways of making the final decision from the decisions of multiple models to be ensembled:
  - Simple average
  - Weighted average
    - Based on performance of each model (Random Forest, Boosted Decision Tree)
    - Weights are determined by another machine learning model

Random Forest (Decision Forests)

Ensemble of multiple independently trained decision trees

- Each tree is trained using a sample of observations and a sample of independent variables
  - Think about three doctors diagnosing heart disease. One doctor is trained by just looking at ECG, one doctor is a Chinese medicine doctor who is trained only by only touching the pulse, and one doctor is trained by looking at the ultrasound image
  - Each doctor is trained on data of different patients (there might be overlapping among the sets of patients)

Advantages of Random Forest:

- Significantly better performance than individual trees
- Automatic Feature Selection
- Less risk of overfitting
- Can be parallelized easily (training of multiple doctors can happen at the same time independently)

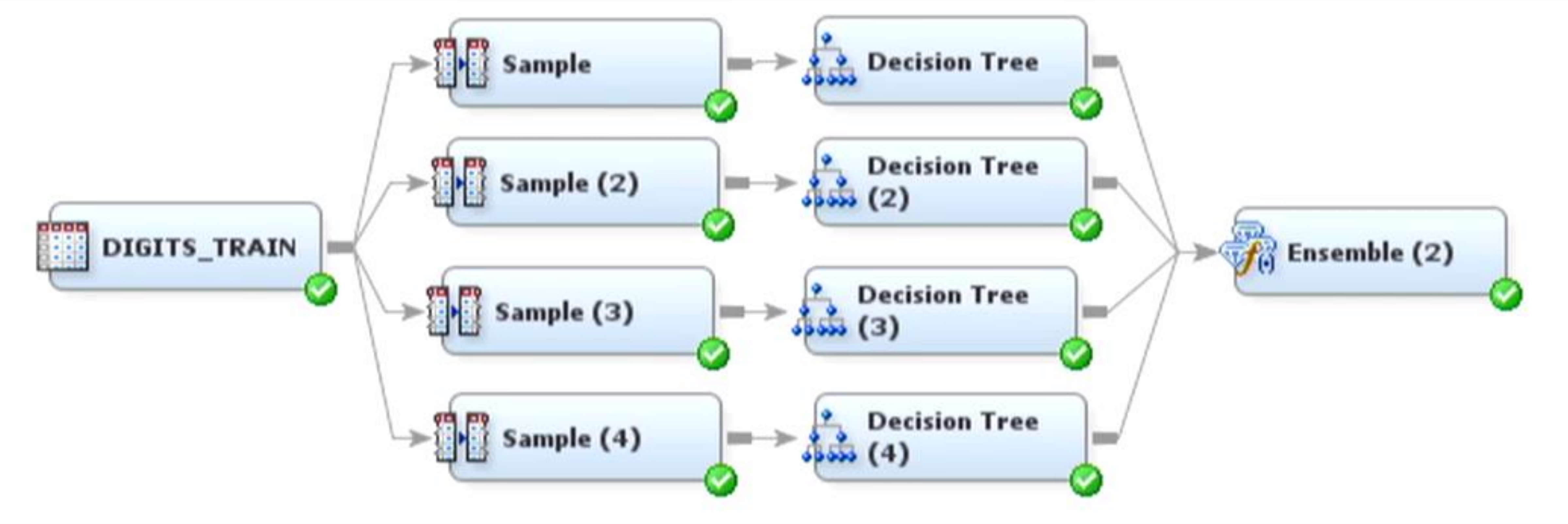
Disadvantages:

- Less interpretability than decision trees
- In some algorithms, data is copied in order to train each tree. Has higher requirement in memory space than individual trees.

Random Forests

- Combination of decision trees and bagging concepts
- A large number of decision trees is trained, each on a different bagging sample
- At each split, only a random number of the original variables is available (i.e. small selection of columns)
- Data points are classified by majority voting of the individual trees

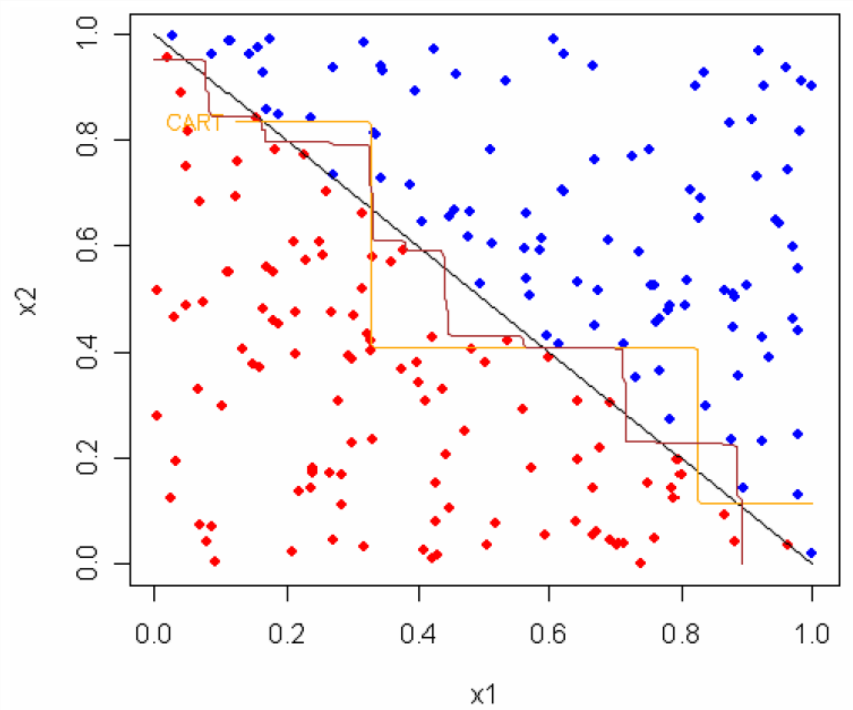
Random Forests



Bagging: reduces variance – Example 1

- Two categories of samples: blue, red
- Two predictors: x1 and x2

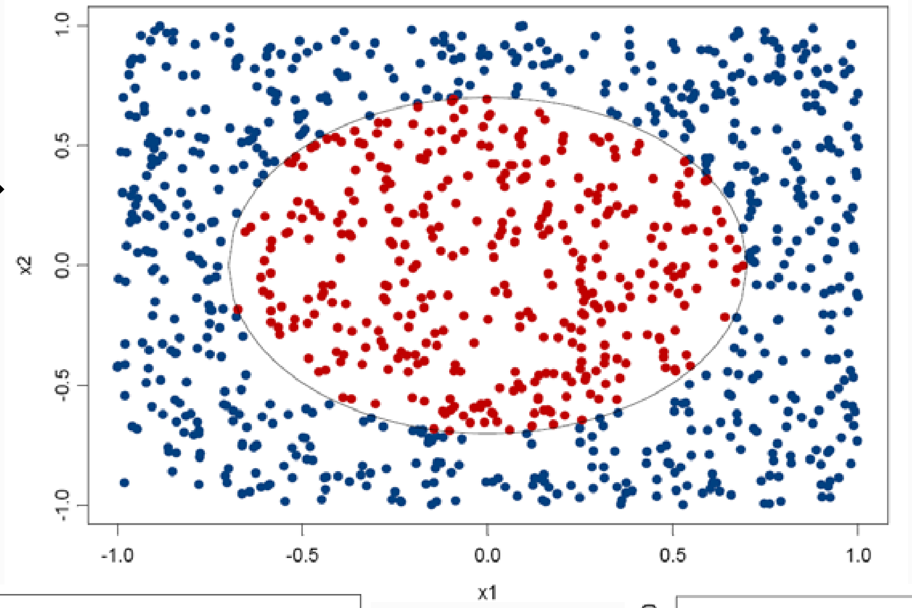
  Diagonal separation…hardest case for tree-based classifier

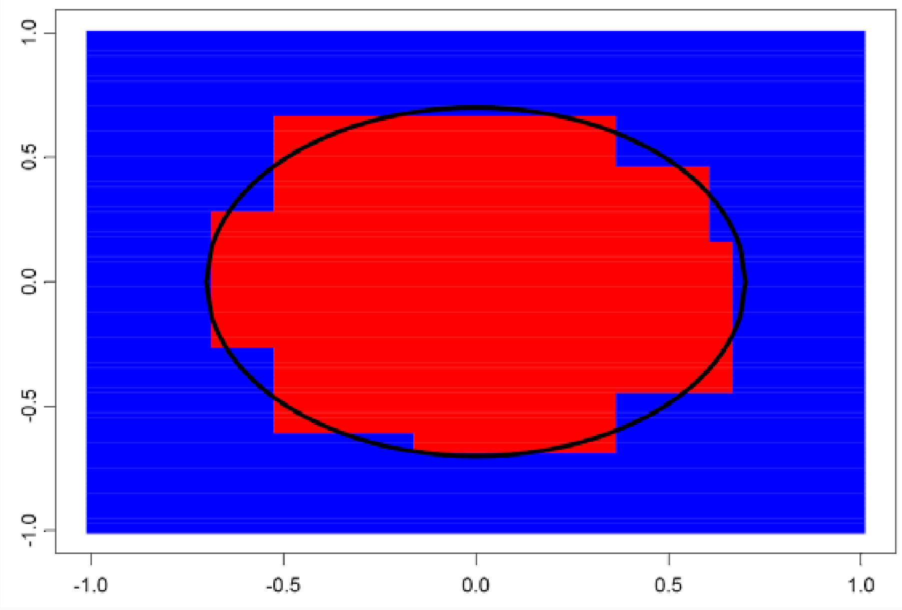


  - Single tree decision boundary in orange.
  - Bagged predictor decision boundary in red.

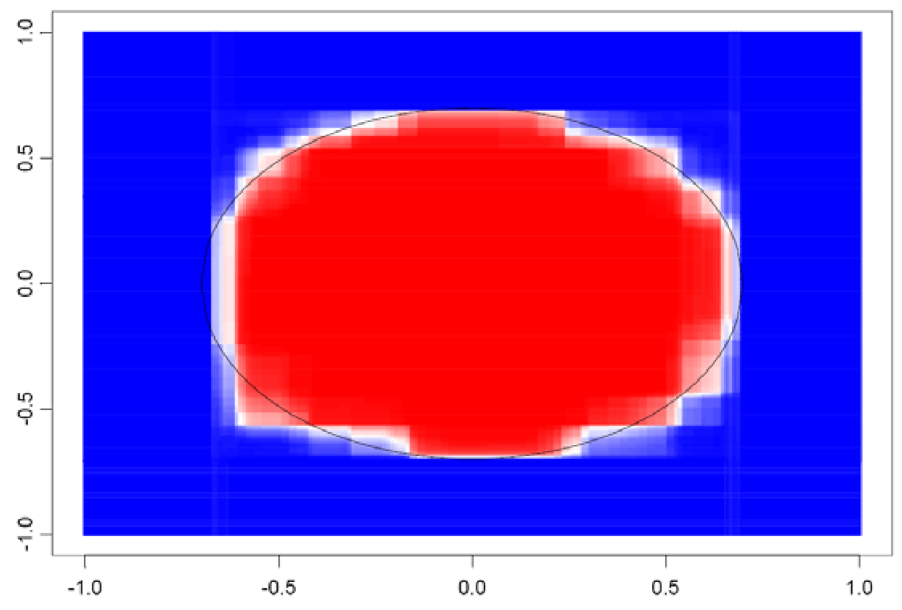
Bagging: reduces variance – Example 2

Ellipsoid separation →
Two categories,
Two predictors





Single tree decision boundary



100 bagged trees..

Random forests

```
D = training set                        F = set of tests
k = nb of trees in forest               n = nb of tests

for i = 1 to k do:
   build data set Di by sampling with replacement from D
   learn tree Ti (Tilde) from Di:
      at each node:
         choose best split from random subset of F of size n
      allow aggregates and refinement of aggregates in tests

make predictions according to majority vote of the set of k trees.
```

Random Forest: How Many Trees to Train?

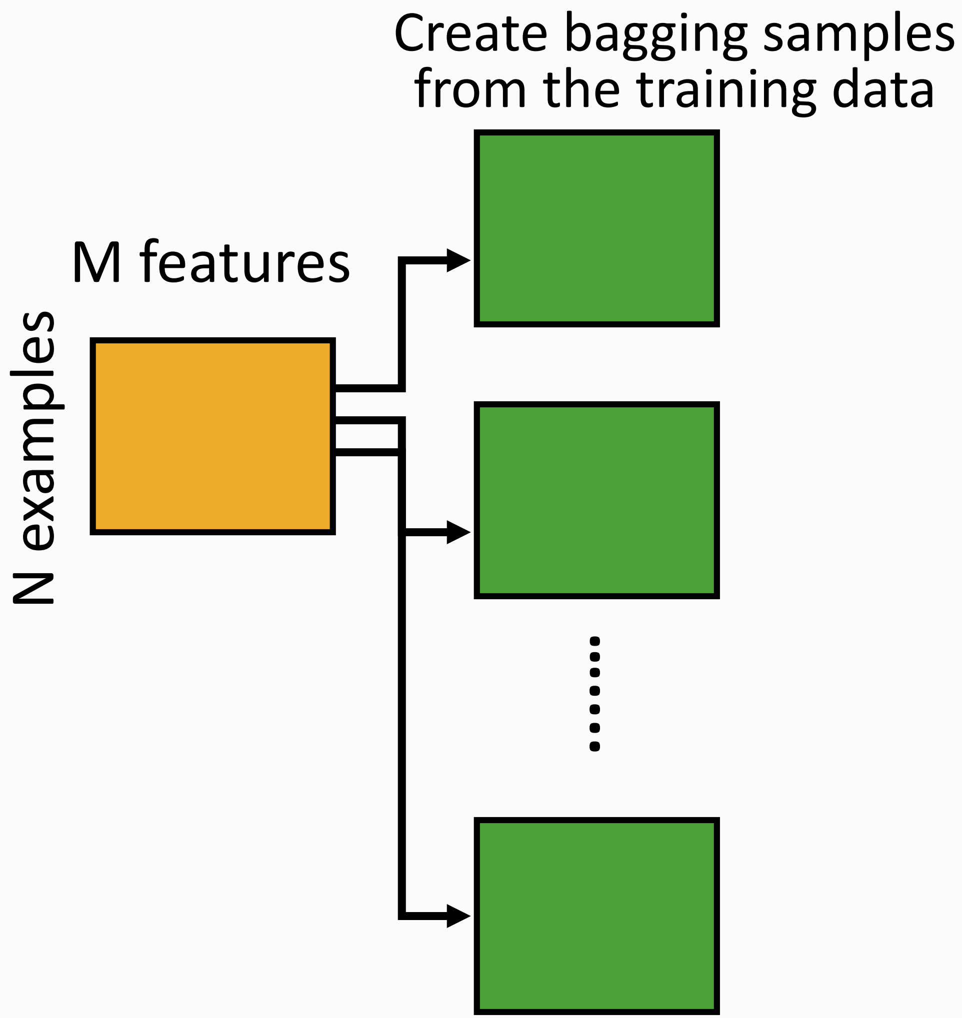
- Rule of thumb:
  - Classification problem: $\sqrt{p}$
  - Regression problem: p/3

- Optimal number is still case by case
  - Start with rule of thumb
  - Tune it to optimize performance

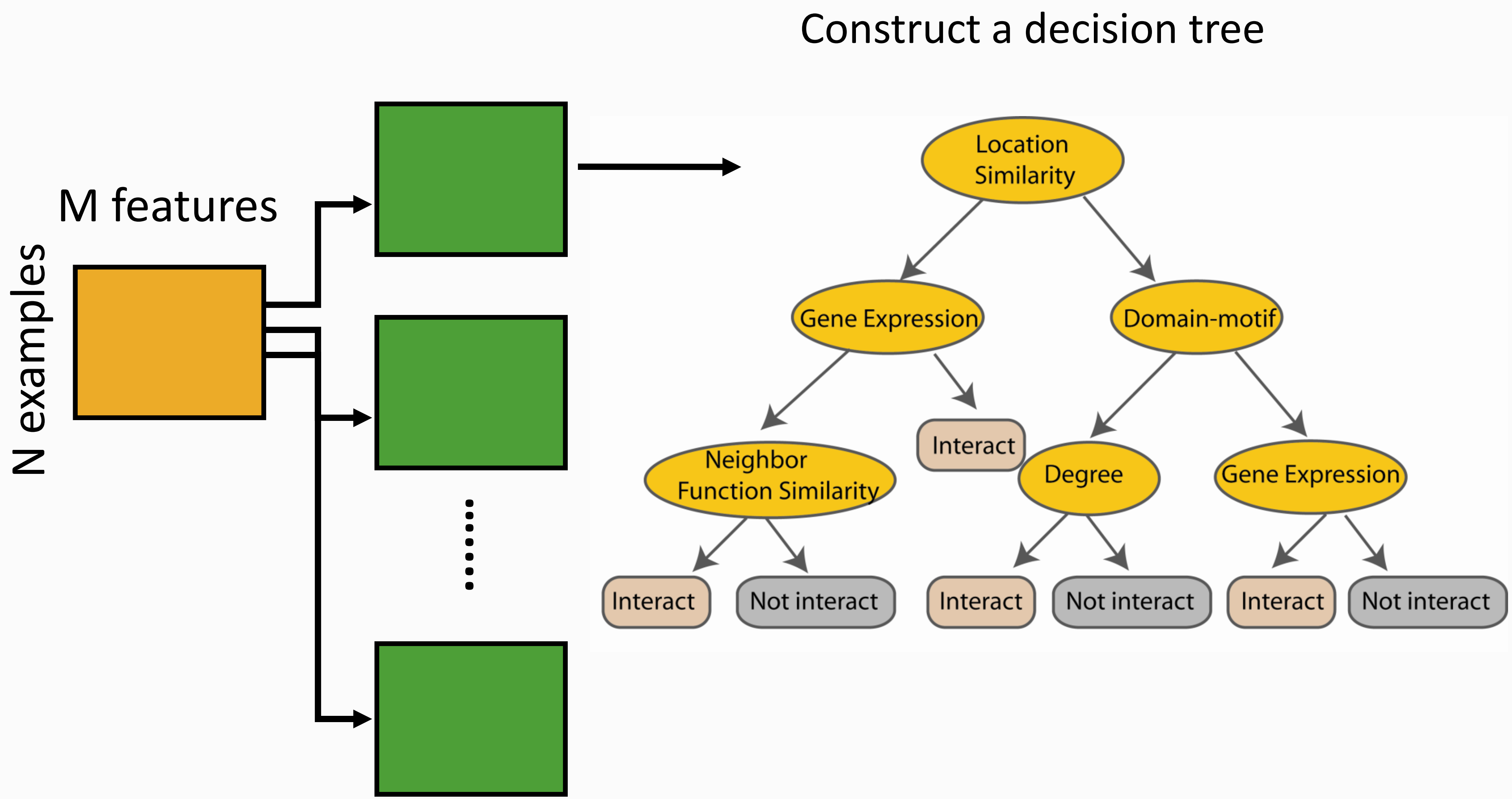
Random Forest Classifier

Training Data

Random Forest Classifier

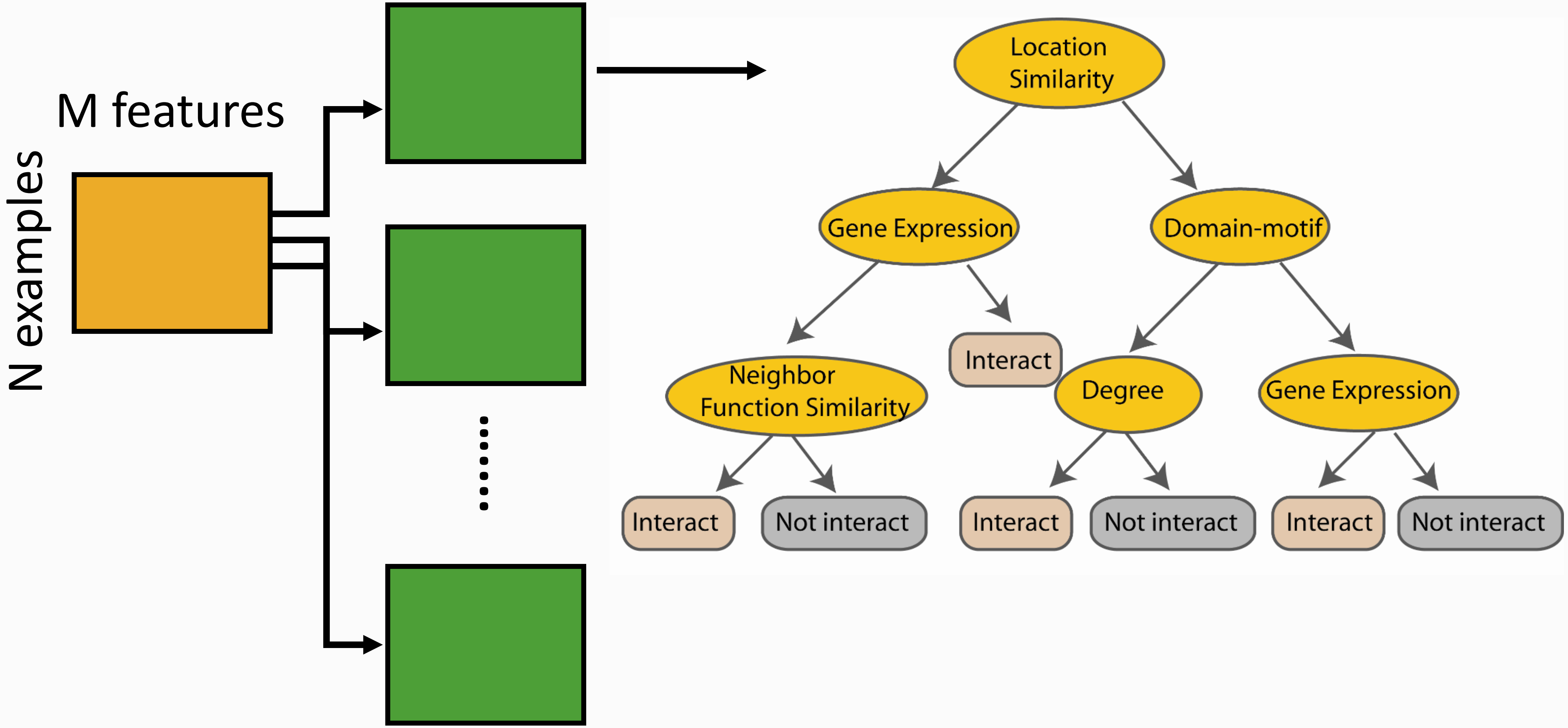


Random Forest Classifier

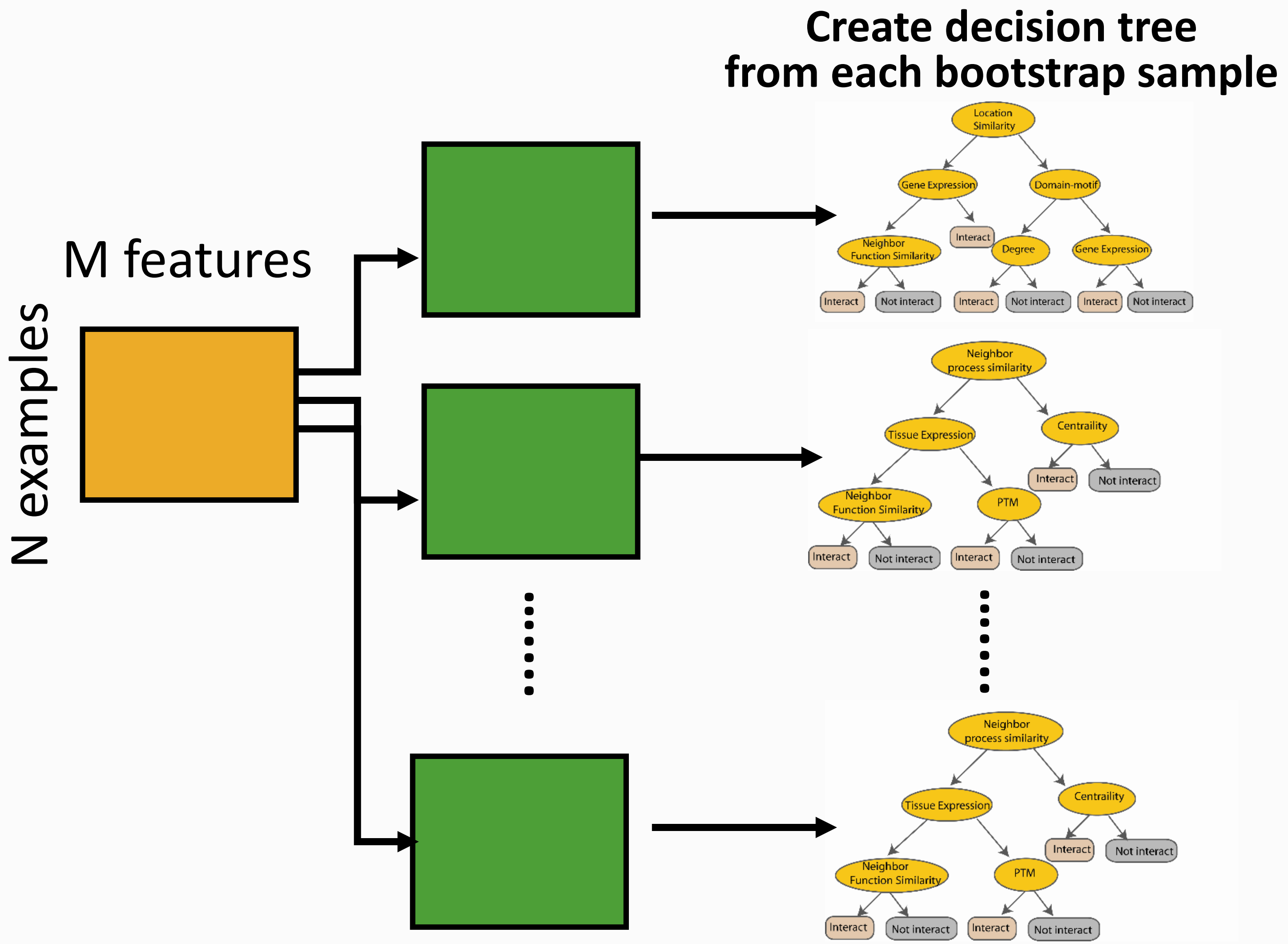


Random Forest Classifier

At each node in choosing the split feature
choose only among *m*<*M* features



Random Forest Classifier



Random Forest Classifier