# Supervised methods

## Classification and Regression Tree
## Random Forest

Dr. Jakub Kuzilek

# Introduction

# Introduction

Today you will learn:

- Basic concepts of decision tree algorithms
- Ensemble methods concepts
- Random Forest

# Introduction

## Small recap: What is conditional probability? Can you define it?

# Introduction

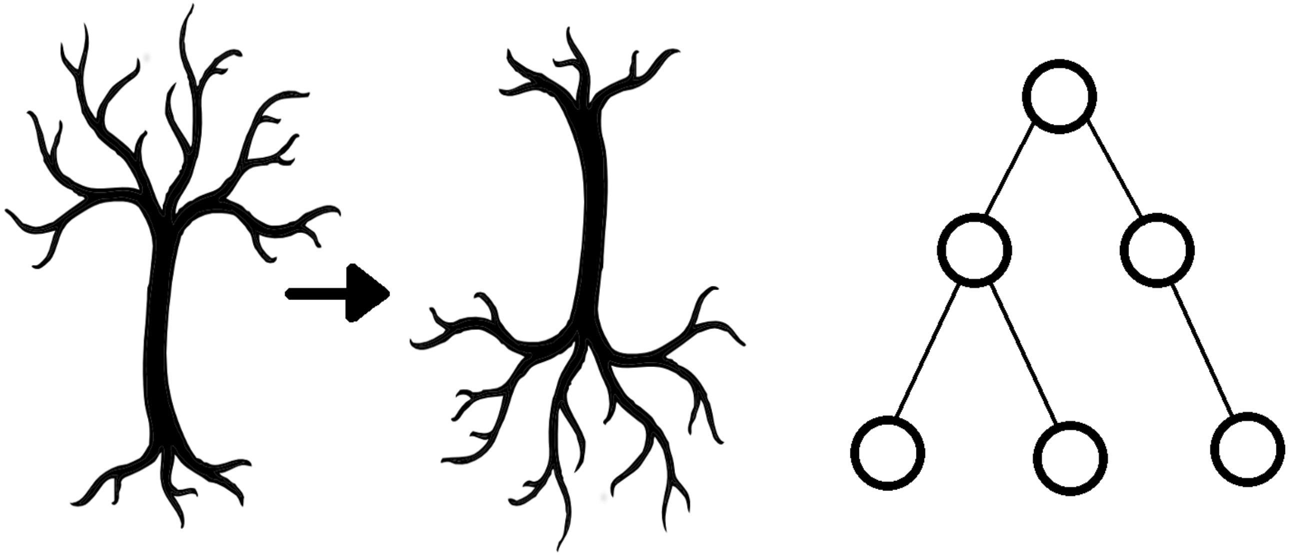
**Small recap: What is conditional probability? Who is it connected to joint probability?**

$$p(A|B)$$

$$p(A,B) = p(A|B)p(B)$$

# Decision Trees

## What is decision tree?

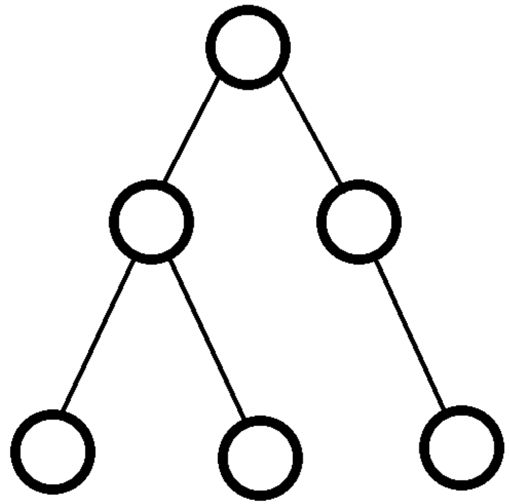


- It is tree-like structure
- Each node represents a test of “something”
- Each edge represents the test outcome
- Nodes are connected by edges.
- Starting node is called root.
- Terminal nodes are called leaves.

# Decision tree - classification

To classify new sample:

1. start at root
2. perform test
3. follow the edge corresponding to outcome
4. if node is not leaf go to 2
5. if node is leaf predict outcome associated with leaf



## Advantages & Disadvantages

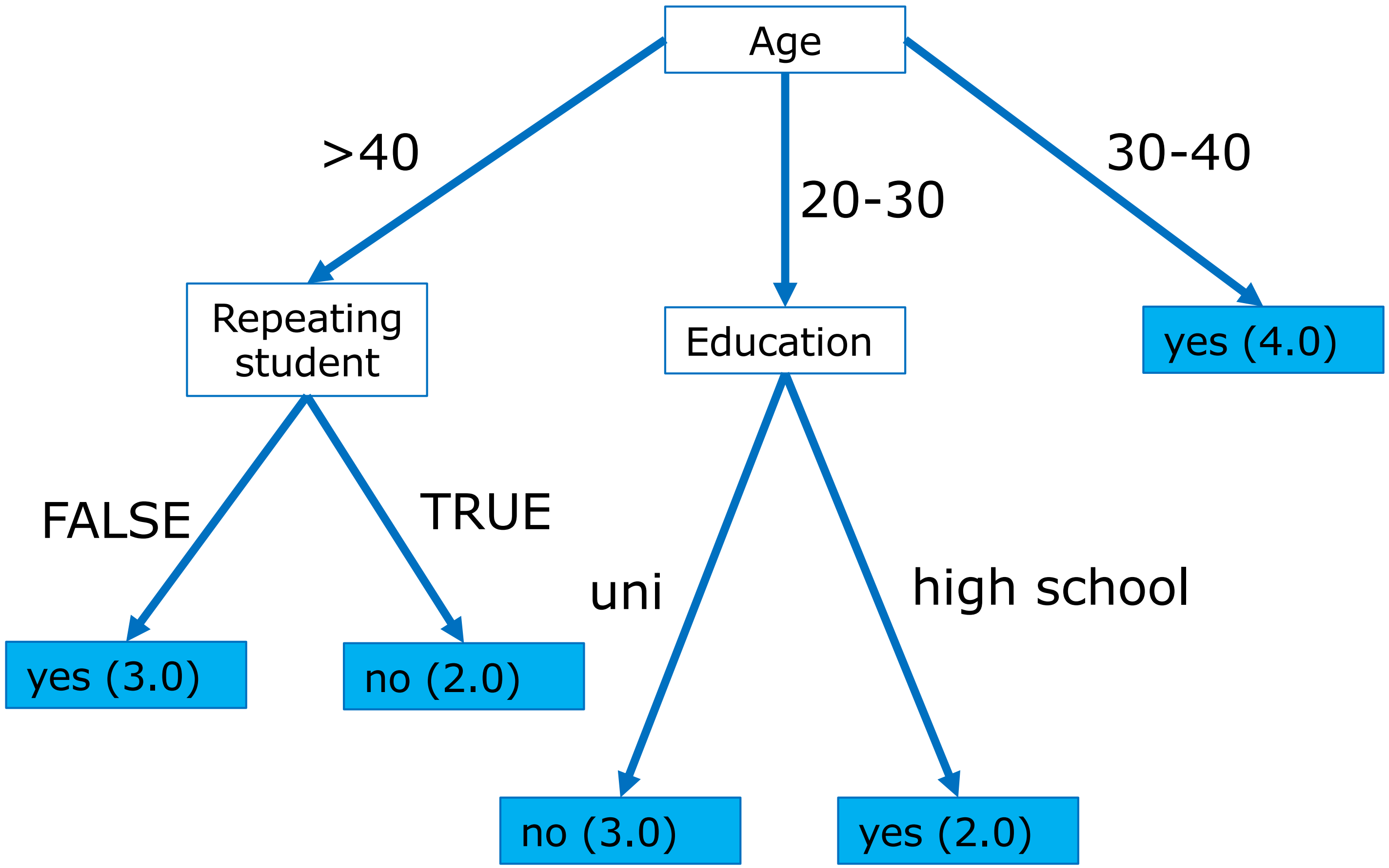
- simple to understand and interpret
- allow addition of new scenarios (nodes)
- decision trees are **unstable** - they strongly depends on data used for tree building
- calculations can be very complex with increasing number of attribute values

# Example

| Background | Age | Prev. education | Repeating student | Passed |
|---|---|---|---|---|
| low | 20-30 | university | FALSE | no |
| low | 20-30 | university | TRUE | no |
| low | 30-40 | university | FALSE | yes |
| medium | >40 | high school | FALSE | yes |
| medium | 30-40 | high school | TRUE | yes |
| high | 20-30 | university | FALSE | no |
| medium | 20-30 | high school | FALSE | yes |
| high | >40 | high school | FALSE | yes |
| high | 20-30 | high school | TRUE | yes |
| high | 30-40 | university | TRUE | yes |
| low | 30-40 | high school | FALSE | yes |
| high | >40 | university | TRUE | no |
| medium | >40 | high school | TRUE | no |
| high | >40 | university | FALSE | yes |

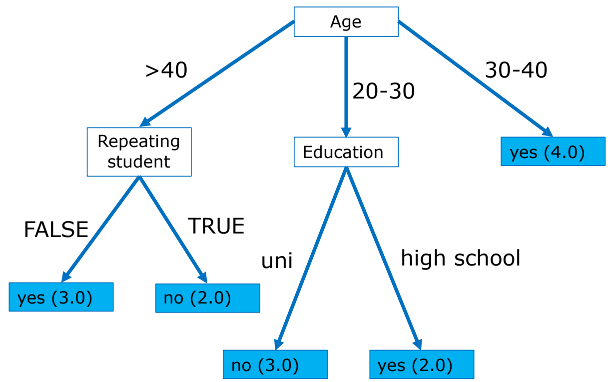
## Example

New student: medium, 20-30, high school and new student



# Building Decision Tree

- Top-Down Induction of Decision Trees (TDIDT)
- Learning in top-down fashion:
  - divide the problem to subproblems
  - solve subproblems
- Algorithm:
  1. select a test for node - create branch for each possible outcome
  2. split instances into subsets - one for each branch coming from node
  3. repeat from 1. recursively using instances that reach the branch
  4. stop when branch contains instances of only one class



# Building Decision Tree

## How to select best test ~ best feature for data split?

= splits of training data set containing mostly samples of single class

- Select features with high degree of “order”:
  - maximum order: all samples in one class
  - minimum order: all classes are equally likely

# Measures of the "order"

- Gini impurity

$$G(S) = \sum_{i=1}^{m} p_i \,(1 - p_i) = 1 - \sum_{i=1}^{m} p_i^2 = \sum_{i \neq k} p_i \, p_k$$

$$G(S, A) = \sum_{i} \frac{|S_i|}{|S|} G(S_i)$$

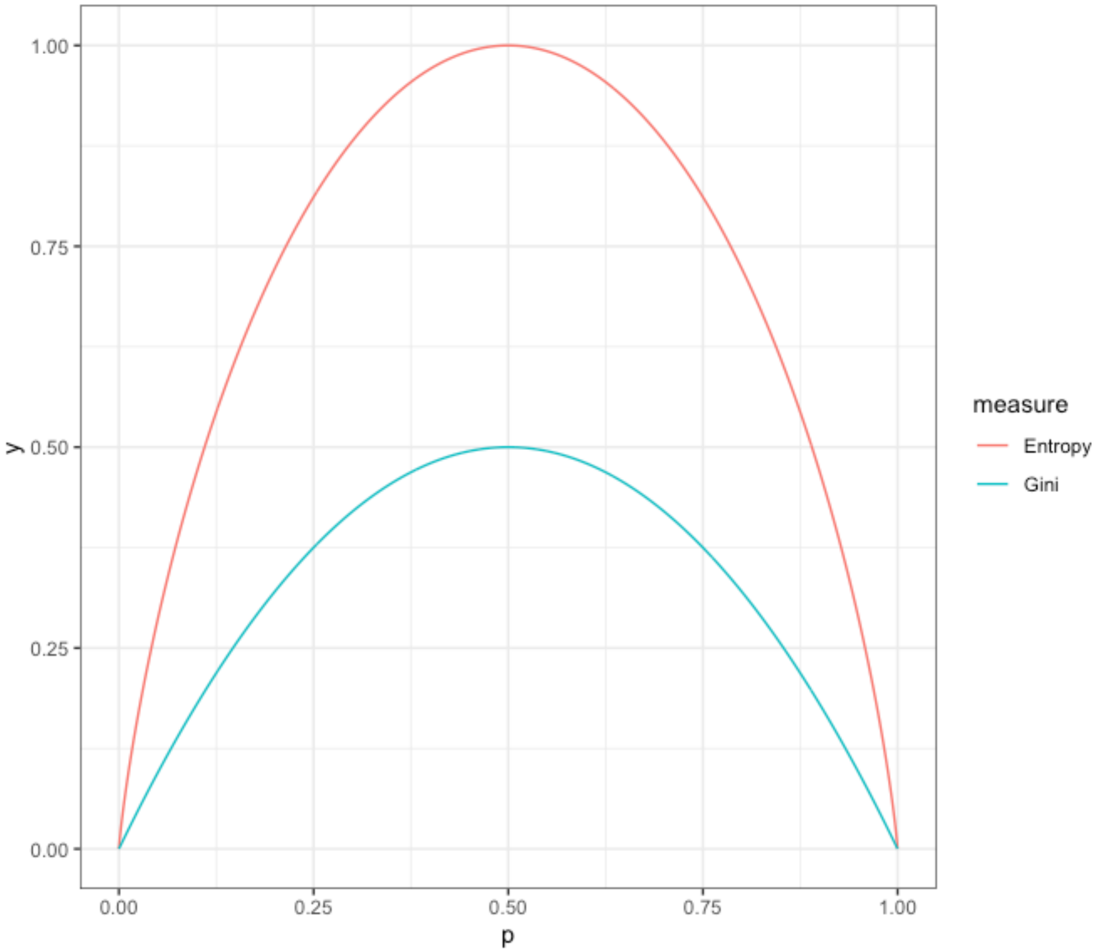
- Information gain

$$E(S) = -\sum_{i=1}^{m} p_i \, log p_i$$

$$Gain(S, A) = E(S) - \sum_{i} \frac{|S_i|}{|S|} E(S_i)$$

# Entropy vs. Gini Impurity

For binary classification.



## Example cont.

passed = yes

passed = no

Attribute Age:

$$E(S) = -\frac{5}{14}log\left(\frac{5}{14}\right) - \frac{9}{14}log\left(\frac{9}{14}\right) = 0.940$$

$$E(age = 20-30) = -\frac{2}{5}log\left(\frac{2}{5}\right) - \frac{3}{5}log\left(\frac{3}{5}\right) = 0.971$$

$$E(age = 30-40) = -\frac{4}{4}log\left(\frac{4}{4}\right) - \frac{0}{4}log\left(\frac{0}{4}\right) = 0$$

$$E(age => 40) = -\frac{3}{5}log\left(\frac{3}{5}\right) - \frac{2}{5}log\left(\frac{2}{5}\right) = 0.971$$

$$I(S, age) = \sum_i \frac{|S_i|}{|S|} E(S_i)$$

$$= \frac{5}{14}0.971 + \frac{4}{14}0 + \frac{5}{14}0.971 = 0.693$$

$$Gain(S, age) = E(S) - I(S, age) = 0.247$$

## Example cont.

Similarly we can compute for others:

$$Gain(S, education) = 0.151$$

$$Gain(S, otherLoans) = 0.048$$

$$Gain(S, salary) = 0.029$$

$$Gain(S, age) = 0.247$$

Thus we will choose age - it has largest information gain.

# Tree pruning

- grown tree covers all the training samples
- very often overfits the data -> difficulties with unseen combinations of feature values (ie. low,>40, high school, TRUE).
- to reduce error tree can be pruned:
  - pre-pruning - stop growing when information becomes unreliable, tends to stop early
  - post-pruning - grow tree and simplify it later, preferred

# Tree pruning

- pre-pruning is based on statistical significance test:
  - chi-square test between feature and class distributions in node
  - only features with statistically significant difference are allowed for selection
- post-pruning algorithm:
  1. learn a complete tree
  2. as long as performance increases try simplify the tree
  3. evaluate resulting trees
  4. return best performing tree

# Numerical attributes

- trees in nature works with categorical variables
- how to deal with continuous numerical variables?
- in every step of building the tree, best attribute selection is started by selection of best numerical attributes split:
  - For each possible split:
    1. estimate the information gain
    2. select the split with largest information gain
- numerical variables can appear several times in the final tree
- categorical variables appears just once – information is exhausted, and reuse gives no advantage

# Algorithms

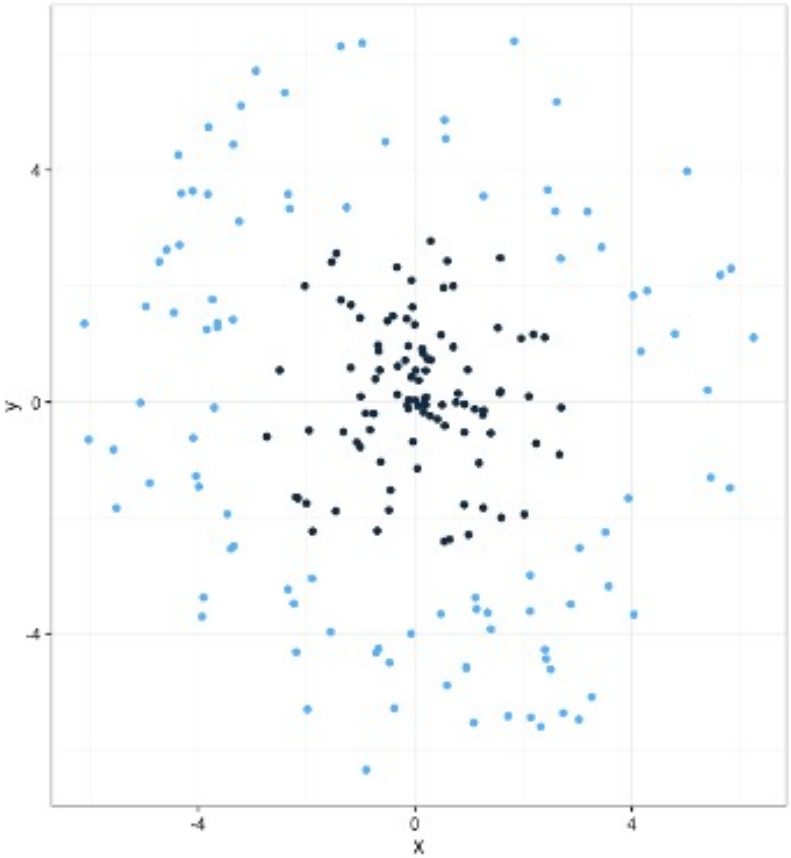
- ID3 (entropy or IG, no pruning, no numerical vars)
- C4.5 (better ID3, normalized IG, post-pruning, ignores numerical vars and missing values)
- CART (Gini, binary tree, post-pruning)

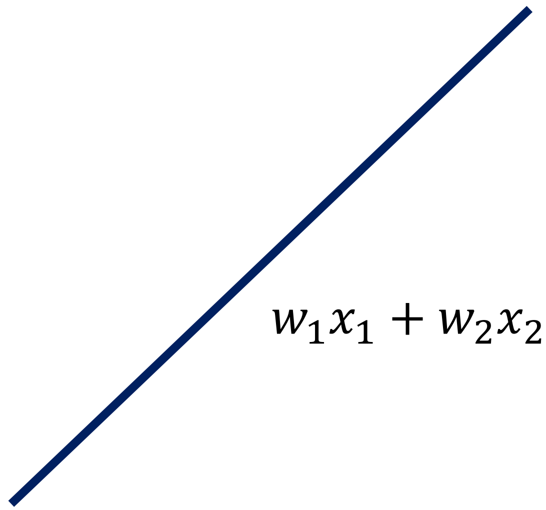
# Ensemble methods

Random Forest

## Introduction

- Data, which are hard to classify
- Only simple weak models are available

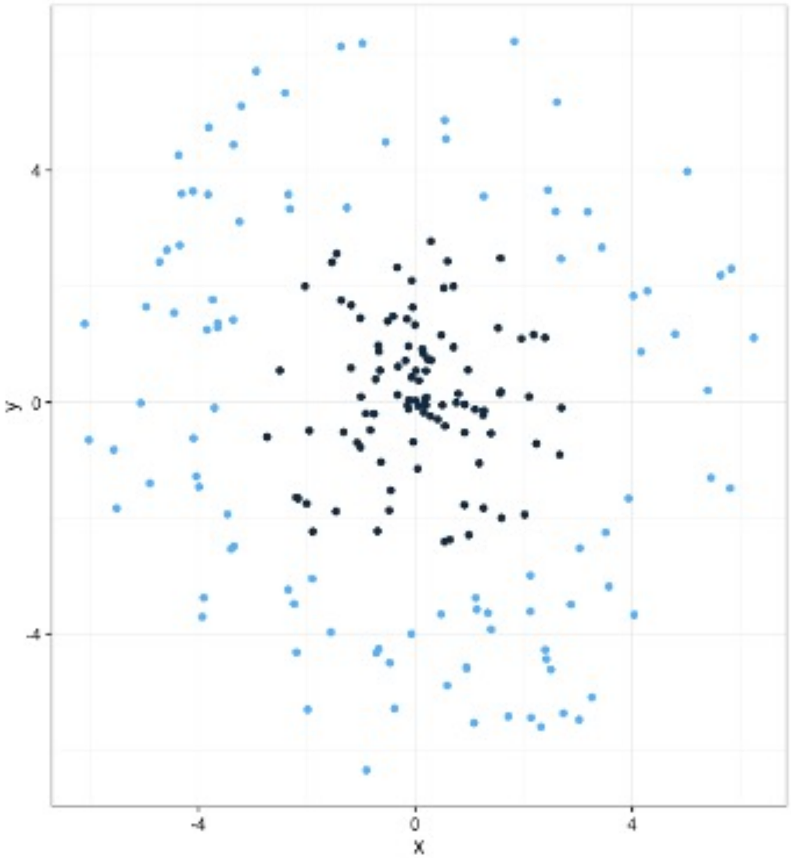


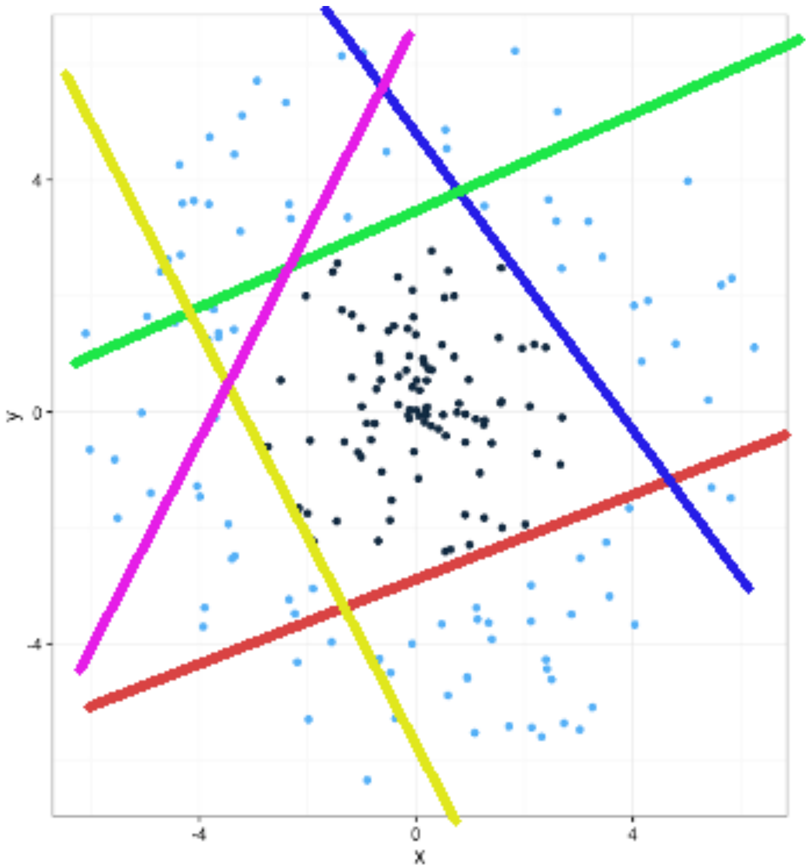



$$w_1x_1 + w_2x_2 + \ w_0 = 0$$

## Introduction

- Data, which are hard to classify
- Only simple weak models are available
- **Combination?**



# Introduction

Approaches:

- ***B**ootstrap **agg**regat**ing** (Bagging)*
  - ➢ each model in ensemble (bag) vote for the final classification with equal weight
- *Boosting*
  - ➢ incrementally building an ensemble by training new model instance to emphasize the training samples previously mis-classified

## Bagging

# Input:

Dataset $T = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_m}, y_m)\}$

Base learning algorithm $\mathbb{L}$

Number of base learners $M$

boosting

# Algorithm:

$$\text{for} \quad m = 1, \ldots, M:$$

$$h_m = \mathbb{L}(T_b)$$

$$end$$

Output: $H(\mathbf{x}) = \underset{y \in Y}{argmax} \sum_{m=1}^{M} \mathbb{I}\left(h_m(\mathbf{x}) = y\right)$

aggregating

# Bagging

- Each weak classifier is trained on boostraped sample of original dataset
- Boostraping is statistical method which samples the original data with replacement
- Selecting i-th sample 0,1,2,… times is Poisson distributed with $\lambda=1$
- The probability that sample will occur in resampled data is around 63%, thus each classifier has not seen at least 37% of original data during training
- Reduces variance of the predicted outcome significantly
- It is efficient when using **unstable** classifiers

# Random Forest

- original bagging algorithm used CART trees
- CART tree is unstable, but not enough for the purpose of bagging
- New version of tree: **Random Tree** (more unstable)
- Many Random Trees = Random Forest
- Random Forest algorithm is the same as original bagging algorithm

# Random Tree

- input:
  - Dataset $\mathrm{T} = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_m}, y_m)\}$
  - feature subset size $K$
- algorithm:
  1. initialize node $N$ using data $T$
  2. if all samples in $N$ are of the same class return $N$
  3. if there is no feature available for split return $N$
  4. **randomly select $\mathbf{K}$ features from those available ($\mathbf{F}$)**
  5. choose best feature from $F$ with best split on $D$
  6. split $D$ to $D_i$ using best split
  7. for each subset $D_i$ repeat from 1
  8. return $N$
- output: random decision tree

# Boosting

- converts weak classifiers to strong one
- iteratively builds strong classifier $H(x)$ using combination of weak classifiers $h_i(x)$:

$$H(x) = Combine\_Outputs(\{h_1(x), \ldots, h_k(x)\})$$

- misclassified samples are the most important

# Questions?