UPPSALA
UNIVERSITET

- Decision trees
- Ensemble methods
- Bagging
  - Random forests
- Boosting

Machine learning – Block 2

Måns Magnusson
Department of Statistics, Uppsala University

November 2022

UPPSALA UNIVERSITET

This week's lecture

- Trees
- Bagging
- Random Forest
- Boosting (Trees)

UPPSALA UNIVERSITET

- Decision trees
- Ensemble methods
- Bagging
  - Random forests
- Boosting

Assignment 1

Short evaluation.

UPPSALA UNIVERSITET

Decision trees: basic idea

A popular method that can be used for both classification and regression is decision trees.

UPPSALA UNIVERSITET

Decision trees: basic idea

A popular method that can be used for both classification and regression is decision trees.

Have you ever played the game "20 questions"?

UPPSALA UNIVERSITET

Decision trees: basic idea

A popular method that can be used for both classification and regression is decision trees.

Have you ever played the game "20 questions"?

Decision trees is more or less that game!

UPPSALA UNIVERSITET

Decision trees: basic idea

A popular method that can be used for both classification and regression is decision trees.

Have you ever played the game "20 questions"?

Decision trees is more or less that game!

In the case of classification, the idea is to classify the new observation by asking a series of questions. Depending on what the answer to the first question is, different second questions are asked, and so on. Questions are asked until a conclusion is reached.

UPPSALA UNIVERSITET

Decision trees: basic idea

Consider the following data set with animal data:

Name	Body temp	Gives birth	Legs	Class
Human	warm-blooded	yes	yes	mammal
Whale	warm-blooded	yes	no	mammal
Cat	warm-blooded	yes	yes	mammal
Cow	warm-blooded	yes	yes	mammal
Python	cold-blooded	no	no	reptile
Komodo dragon	cold-blooded	no	yes	reptile
Turtle	cold-blooded	no	yes	reptile
Salmon	cold-blooded	no	no	fish
Eel	cold-blooded	no	no	fish
Pigeon	warm-blooded	no	yes	bird
Penguin	warm-blooded	no	yes	bird

UPPSALA UNIVERSITET

Decision trees: basic idea

1. Does it give live birth?

UPPSALA UNIVERSITET

Decision trees: basic idea

1. Does it give live birth? (mammal)
2. Is it warm-blooded?

UPPSALA UNIVERSITET

Decision trees: basic idea

1. Does it give live birth? (mammal)
2. Is it warm-blooded? (bird)
3. Does it have legs?

UPPSALA UNIVERSITET

Decision trees: basic idea

1. Does it give live birth? (mammal)
2. Is it warm-blooded? (bird)
3. Does it have legs? (reptile)
4. Else (fish)

The regions of a tree

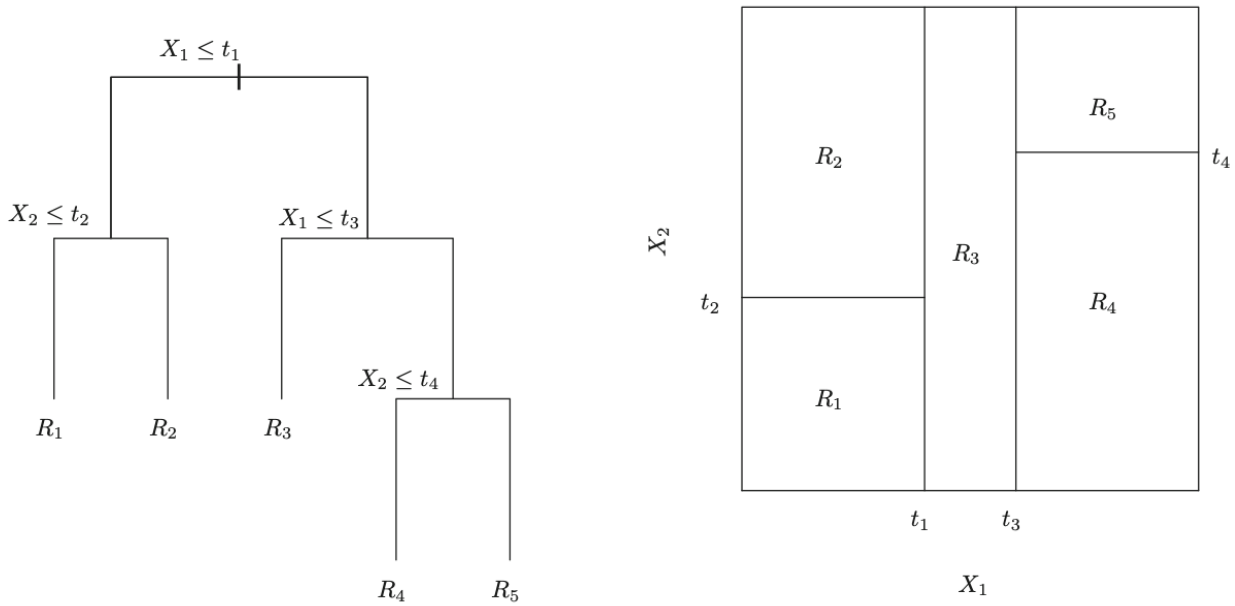

Figure: Regions of a tree (Garreth et al, 2013, Fig. 8.3)

UPPSALA UNIVERSITET

Regression Trees

$$T(x) = \sum_{m=1}^{M} \gamma_m I(x \in R_m),$$

where M is the total number of regions and $I(x \in R_m)$ is an indicator variable if x_i belongs to region R_m. γ_m is the prediction for region m.

UPPSALA UNIVERSITET

Regression Trees

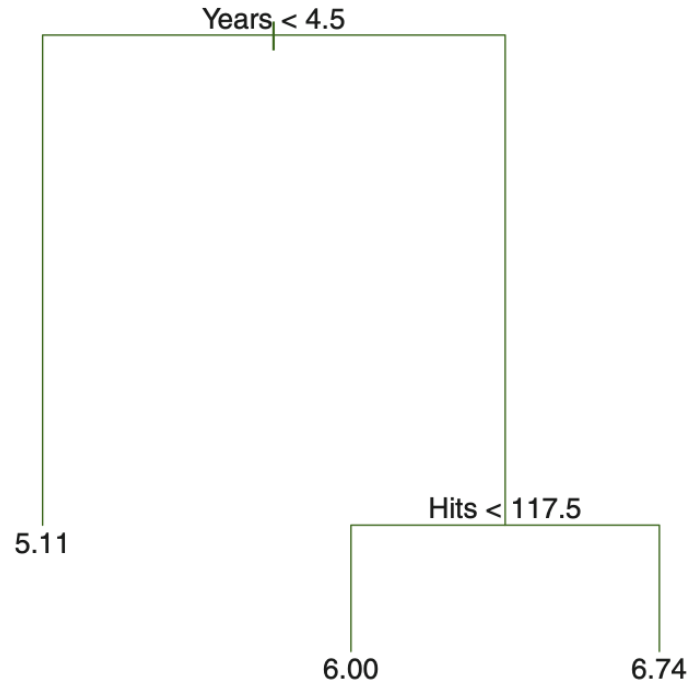

Figure: Regression Tree (Garreth et al, 2013, Fig. 8.1.)

- The `Hitters dataset`: Salaries of Baseball players.
- The end of the tree contain the observations.

UPPSALA UNIVERSITET

- Decision trees
- Ensemble methods
- Bagging
  - Random forests
- Boosting

Regression Trees

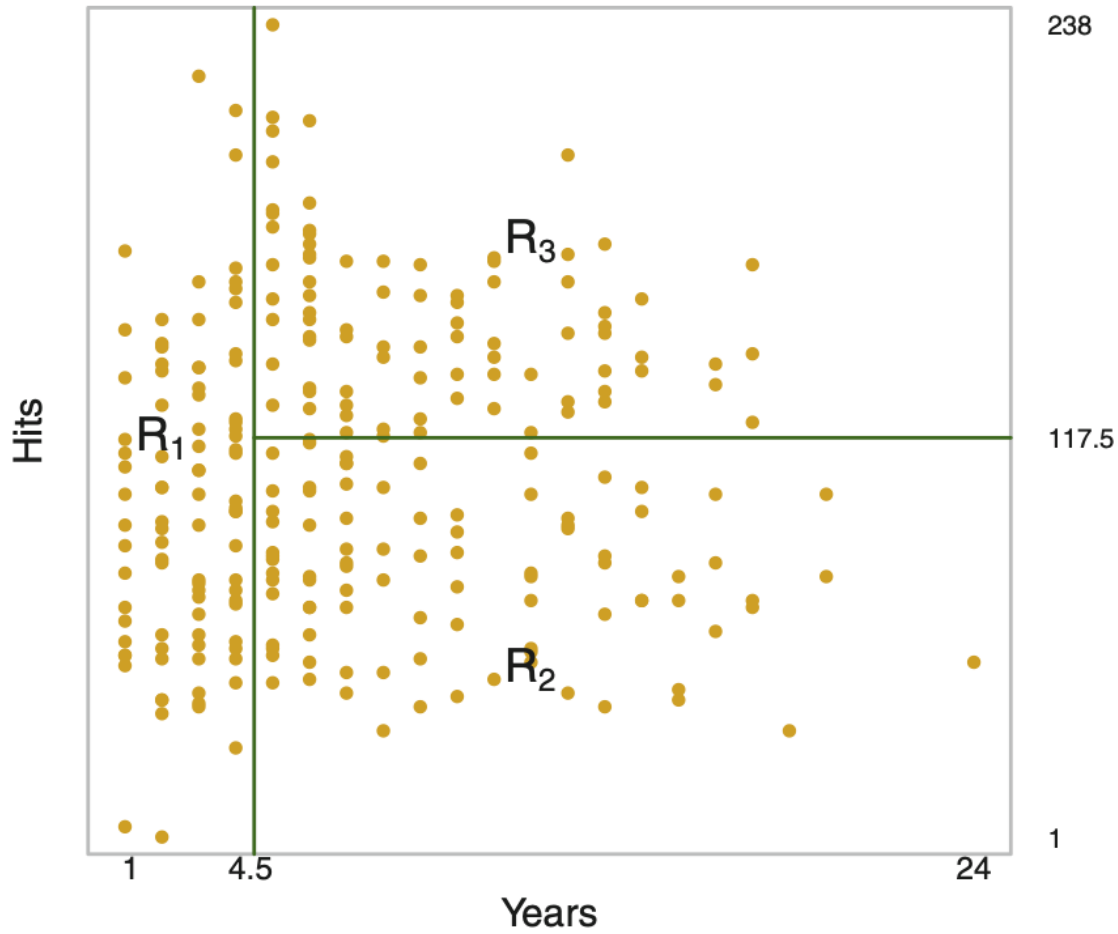

Figure: `Hitters` data and regression tree regions (Garreth et al, 2013, Fig. 8.2.)

UPPSALA UNIVERSITET

Growing a Decision Tree

1. A tree has two groups of parameters $\Theta = (\gamma, R)$ that we need to learn.

UPPSALA UNIVERSITET

Growing a Decision Tree

1. A tree has two groups of parameters $\Theta = (\gamma, R)$ that we need to learn.
2. We want a tree that minimize $L(\theta) = (y_i - T_\Theta(x_i))^2$

UPPSALA UNIVERSITET

Growing a Decision Tree

1. A tree has two groups of parameters $\Theta = (\gamma, R)$ that we need to learn.
2. We want a tree that minimize $L(\theta) = (y_i - T_\Theta(x_i))^2$
3. Usually we estimate γ_m as the mean of y_i in the region as:

$$\hat{\gamma}_m = \frac{1}{N_m} \sum_{x_i \in R_m}^{N_m} y_i \,,$$

where N_m is the number of observations in region R_m.

UPPSALA UNIVERSITET

Growing a Decision Tree

1. A tree has two groups of parameters $\Theta = (\gamma, R)$ that we need to learn.
2. We want a tree that minimize $L(\theta) = (y_i - T_\Theta(x_i))^2$
3. Usually we estimate γ_m as the mean of y_i in the region as:

$$\hat{\gamma}_m = \frac{1}{N_m} \sum_{x_i \in R_m}^{N_m} y_i \,,$$

   where N_m is the number of observations in region R_m.
4. Learning R_m is generally computationally infeasable so we use a greedy heuristic.

UPPSALA UNIVERSITET

Growing a Decision Tree: Greedy Algorithm

Let $\mathcal{S}$ be the set of all observations $\{1, ..., N\}$ and `S[[m]]` be the set of observation indecies in R_m and `l` is the minimal number of leafs per node.

Input: $\mathcal{S}$, `X`, `y`, `l`

1. `S[[1]]` $= \mathcal{S}$,`M = 1`, `m = 1`
2. `while m <= M` then do:
   - 2.1 `if(size(S[[m]]) >= 2*l)`
     - 2.1.1 `S[[M+1]], S[[M+2]], j[m], s[m] = split_tree(X[S[[m]],], y[S[[m]],],l)`
     - 2.1.2 `M = M + 2`
   - 2.2 `else`
     - 2.2.1 compute $\hat{\gamma}$ for `S[[m]]`
   - 2.3 `m = m + 1`

Output: `j`, `s`, γ

Example: `j = {Years, Hits}`, `s = {4.5, 117.5}`, $\hat{\gamma} = \{122, 317, 245\}$

UPPSALA UNIVERSITET

How to do a split - the math.

Here we try to compute Eq. (9.12-9.14) in ESL:

$$R_1(j,s) = \{X|X_j \leq s\} \text{ and } R_2(j,s) = \{X|X_j > s\}$$

$$\min_{j,s} \left(\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right)$$

Inner minimization is solved by:

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m}^{N_m} y_i \, ,$$

UPPSALA UNIVERSITET

How to do a split?

Input: $\mathbf{X}, \mathbf{y}, l$

1. $SS = \text{Inf}$ # Store sum of squares in matrix of dim $P \times N_s$
2. $S = \text{Inf}$ # Store split point in matrix of dim $P \times N_s$
3. for $j \in \{1, ..., P\}$ # all features
   - 3.1 for $k \in \{1, ..., N_s\}$ # all observations in set s
     - 3.1.1 $s = x_{k,j}$ # Split point (use the value of x)
     - 3.1.2 if ($|R_1(s,j)| < l$ or $|R_2(s,j)| < l$) next # Dont create too few leaves
     - 3.1.3 $\hat{c}_1 = \frac{1}{|R_1(s,j)|} \sum_{x_i \in R_1(s,j)} y_i$
     - 3.1.4 $\hat{c}_2 = \frac{1}{|R_2(s,j)|} \sum_{x_i \in R_2(s,j)} y_i$
     - 3.1.5 $SS_{k,j} = \sum_{x_i \in R_1(s,j)} (y_i - c_1)^2 + \sum_{x_i \in R_2(s,j)} (y_i - c_2)^2$ # Compute Sum of Squares
     - 3.1.6 $S_{k,j} = \text{s}$
4. $k_{final}, j_{final} = \min_{k,j} SS$
5. $s_{final} = S_{k_{final}, j_{final}}$
6. return $R_1(s_{final}, j_{final}), R_2(s_{final}, j_{final}), s_{final}, j_{final}$

UPPSALA UNIVERSITET

Decision trees: Classification Trees

How do we do if we have a classification tree?

UPPSALA UNIVERSITET

Decision trees: Classification Trees

How do we do if we have a classification tree?

We just change the loss function.

UPPSALA UNIVERSITET

Decision trees: Classification Trees

How do we do if we have a classification tree?

We just change the loss function.

Let $p(j|t)$ be the fraction of observations in class j at the node t and let c be the number of classes.

UPPSALA UNIVERSITET

Decision trees: Classification Trees

How do we do if we have a classification tree?

We just change the loss function.

Let $p(j|t)$ be the fraction of observations in class j at the node t and let c be the number of classes.

The Gini for node t is defined as

$$Gini(t) = 1 - \sum_{j=1}^{c} (p(j|t))^2$$

UPPSALA UNIVERSITET

Decision trees: Classification Trees

How do we do if we have a classification tree?

We just change the loss function.

Let $p(j|t)$ be the fraction of observations in class j at the node t and let c be the number of classes.

The Gini for node t is defined as

$$Gini(t) = 1 - \sum_{j=1}^{c} (p(j|t))^2$$

Gini is a measure of "impurity". If all observations belong to the same class, then

$$Gini(t) = 1 - 1^2 - 0 - \ldots - 0 = 0.$$

UPPSALA UNIVERSITET

Decision trees: Classification Trees

How do we do if we have a classification tree?

We just change the loss function.

Let $p(j|t)$ be the fraction of observations in class j at the node t and let c be the number of classes.

The Gini for node t is defined as

$$Gini(t) = 1 - \sum_{j=1}^{c} (p(j|t))^2$$

Gini is a measure of "impurity". If all observations belong to the same class, then

$$Gini(t) = 1 - 1^2 - 0 - \ldots - 0 = 0.$$

The Gini is maximized when all classes have the same number of observations at t.

UPPSALA UNIVERSITET

Decision trees: Classification Trees

How do we do if we have a classification tree?

We just change the loss function.

Let $p(j|t)$ be the fraction of observations in class j at the node t and let c be the number of classes.

The Gini for node t is defined as

$$Gini(t) = 1 - \sum_{j=1}^{c} (p(j|t))^2$$

Gini is a measure of "impurity". If all observations belong to the same class, then

$$Gini(t) = 1 - 1^2 - 0 - \ldots - 0 = 0.$$

The Gini is maximized when all classes have the same number of observations at t.

One criterion for splitting could be to minimize the Gini in the next level of the tree. That way we will get "purer" nodes.

UPPSALA UNIVERSITET

Important concepts

- Tree depth: the length of the longest path from the root to a leaf (i.e. greatest number of questions that the tree can ask).

UPPSALA UNIVERSITET

Important concepts

- Tree depth: the length of the longest path from the root to a leaf (i.e. greatest number of questions that the tree can ask).
- Leaf size: the number of observations in a leaf.

UPPSALA UNIVERSITET

Important concepts

- Tree depth: the length of the longest path from the root to a leaf (i.e. greatest number of questions that the tree can ask).
- Leaf size: the number of observations in a leaf.

Decision trees can become quite large, which may lead to:

- Overfitting (high variance)
- Difficulties interpreting the tree

UPPSALA UNIVERSITET

Important concepts

- Tree depth: the length of the longest path from the root to a leaf (i.e. greatest number of questions that the tree can ask).
- Leaf size: the number of observations in a leaf.

Decision trees can become quite large, which may lead to:

- Overfitting (high variance)
- Difficulties interpreting the tree

The solution to this is

- Pruning: forcing the tree to be smaller by adding a stopping condition, e.g. a maximum depth or minimal leaf size.

UPPSALA UNIVERSITET

Important concepts

- Tree depth: the length of the longest path from the root to a leaf (i.e. greatest number of questions that the tree can ask).
- Leaf size: the number of observations in a leaf.

Decision trees can become quite large, which may lead to:

- Overfitting (high variance)
- Difficulties interpreting the tree

The solution to this is

- Pruning: forcing the tree to be smaller by adding a stopping condition, e.g. a maximum depth or minimal leaf size.
- But decision trees are quite bad predition models...

UPPSALA UNIVERSITET

General idea of ensembles

The idea of an ensemble is simple: If it difficult to find one really good model perhaps we can find several weaker models and combine their predictions.

A simple example: Say you have one outcome Y and 4 covariates X_1, X_2, X_3, X_4. The goal is to predict Y. A possible ensemble would be to fit

$$y = \alpha_1 + \beta_1 X_1 + \epsilon_1$$
$$y = \alpha_2 + \beta_2 X_2 + \epsilon_2$$
$$y = \alpha_3 + \beta_3 X_3 + \epsilon_3$$
$$y = \alpha_4 + \beta_4 X_4 + \epsilon_4$$

and then use the mean of their predictions

$$\hat{y}_{ensemble} = \frac{1}{4}\sum_{i=1}^{4}\hat{y} = \frac{1}{4}\sum_{i=1}^{4}(\hat{\alpha}_i + \hat{\beta}_i X_i) \quad (1)$$

UPPSALA UNIVERSITET

Two key parts of an ensemble

- The prediction models (sometimes called 'learners' in ML literature)
  - A single model in an ensemble can be a simple or a complex model
  - Often the ensemble contains many simple models.
  - Wikipedia: "*In statistics and machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone*"
- The weighting of each prediction in the final ensemble prediction
  - Models/learners with better predictive power can be given larger weights in the final prediction
  - There are many complex algorithms for weighting together the predictions from many models

UPPSALA UNIVERSITET

Ensambles of decision trees

The most common type of ensembles is ensembles of decision trees.

We will focus on this case, but note that any type of model can be included in an ensemble in principle.

UPPSALA UNIVERSITET

Bagging and Boosting

Remember, the error of a prediction/classification can be decomposed as

$$error = bias + variance + bayeserror. \tag{2}$$

- Complex models/strong learners (with many parameters) tend to have small bias and large variance (tend to be overfitted)
- Shallow models/weak learners (with few parameters) tend to have small variance and large bias

UPPSALA UNIVERSITET

Bagging and Boosting

Remember, the error of a prediction/classification can be decomposed as

$$error = bias + variance + bayeserror. \tag{2}$$

- Complex models/strong learners (with many parameters) tend to have small bias and large variance (tend to be overfitted)
- Shallow models/weak learners (with few parameters) tend to have small variance and large bias

Bagging: Ensemble methods that aim to decrease the variance of complex/strong learners with low bias and large variance

Boosting: Ensemble methods that aim to decrease the bias of shallow/weak learners with low variance and large bias

UPPSALA UNIVERSITET

Bagging (Bootstrap AGGregating) – Bootstrap improvement of prediction models

Consider a sample of N units.

Bagging algorithm:

1. Draw, *with replacement*, a random sample of N units from the original sample
2. Fit a prediction model (e.g., a decision tree)
3. Repeat steps 1-2 B times
4. Weight together the predictions from the B models into a final ensemble prediction

UPPSALA UNIVERSITET

Bagging (Bootstrap AGGregating) – Bootstrap improvement of prediction models

Consider a sample of N units.

Bagging algorithm:

1. Draw, *with replacement*, a random sample of N units from the original sample
2. Fit a prediction model (e.g., a decision tree)
3. Repeat steps 1-2 B times
4. Weight together the predictions from the B models into a final ensemble prediction

Train several deep trees and combine their results by weighting together their predictions

UPPSALA UNIVERSITET

Random Forest

A random forest is a bagging ensemble method, but with one extra step. Consider a sample of N units and K observed covariates/features

Random forest algorithm:

1. Draw, *with replacement*, a random sample of N units from the original sample
2. **Draw, without replacement, a random subset of k covariates/features**
3. Fit a prediction model (e.g., a decision tree)
4. Repeat step 1-3 B times
5. Weight together the predictions from the B models into a final ensemble prediction

It is common to use $k = \sqrt{K}$ (rounded down) for classification and $k = K/3$ for regression. But these are only rules of thumb: k is a *tuning parameter*.

UPPSALA UNIVERSITET

Random Forest, variance reduction

Consider each tree to be an i.i.d. random variable with variance σ^2.
The average of these trees then have variance $\frac{1}{B}\sigma^2$. Trees constructed from the same set of covariates will be correlated and therefore not independent. The variance of the average of these correlated trees then becomes

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2.$$

The second term will vanish with increasing B leaving just the first term left which is a function of the correlation between the trees and the variance. The remaining part of the variance is minimized by only consider a subset of the covariates when constructing trees - reducing the correlation between them.

UPPSALA UNIVERSITET

Differences between bagging and random forest:

- In bagging, the trees are often highly correlated
  - If some covariates are strong predictors of the outcome (in the training data), many trees in the 'bag' will us the same covariates in their decisions
- In a random forest, the trees are less similar/correlated since all covariates are not available when each tree is constructed.

This means that a random forest (with many trees) uses the predictive ability of all covariates rather than just a few, which usually improves out of sample performance.

UPPSALA UNIVERSITET

Boosting

In boosting, models are trained sequentially where each new model tries to target weak spots of the previous models in the ensemble to improve the performance of the ensemble

Boosting

1. Fit a prediction model/classifier (e.g., a decision tree) using the original sample
   - Give the misclassified observations higher weights
2. Draw, *with replacement*,*with probability proportional to the weights*, a random sample of N units from the original sample
3. Fit a prediction model/classifier (e.g., a *shallow* decision tree) using the new sample
4. Update the weights of each observation according to the average misclassification of the trained classifiers
5. Repeat step 2-4 B times
6. Weight together the predictions from the B models into a final ensemble prediction, giving larger weights to classifiers with smaller errors

UPPSALA UNIVERSITET

Boosting

For boosting to work well, the updates of the weights must be chosen in some clever way. One successful method is *gradient descent*.

We will not focus more on the particular algorithms. For now, we are satisfied with understanding the concept of boosting:

Train a bunch of classifiers sequentially. Force each new classifier to train more on data that the previous classifiers had problems with classifying by giving those samples a higher probability to be sampled.

UPPSALA UNIVERSITET

XGBoost

1. State of the Art method

UPPSALA UNIVERSITET

XGBoost

1. State of the Art method
2. Use gradient boosting trees with regularization

UPPSALA UNIVERSITET

XGBoost

1. State of the Art method
2. Use gradient boosting trees with regularization
3. Is scalable to very large data