Fakultas Informatika
School of Computing
Telkom University

Machine Learning

Ensemble Methods

ADF

Outline

- Decision Trees
- Random Forest
- Random Decision Tree

Review Decision Trees

Decision Trees

- Decision trees have a long history in machine learning
  - The 1st popular algorithm dates back to 1979
- Very popular in many real world problems
- Intuitive to understand
- Easy to build, easy to use, easy to interpret

- But in practice, they are not that awesome

Decision Trees

- "Trees have one aspect that prevents them from being the ideal tool for predictive learning, namely **inaccuracy**"
  - - the elements of statistical learning

- Work great with the data used to create them
- **Not flexible when it comes to classifying new sample**

Smaller Decision Trees

- More flexible, reduce overfitting
- Bias-Variance Tradeoff
  - Bias: Representation Power of Decision Trees
  - Variance: require a sample size exponential in depth

Random Forest

Random Forest

- A class of ensemble methods specifically designed for decision tree classifier
- It combines the predictions made by multiple decision trees,
  - where each tree is generated based on the values of an independent set of random vectors,
- Combining the simplicity of decision trees with flexibility resulting in a vast improvement in accuracy

Random Forest Algorithm

- Choose T—number of trees to grow
- Choose m<M (M is the number of total features) —number of features used to calculate the best split at each node (typically 20%)
- For each tree
  - Choose a training set by choosing N times (N is the number of training examples) with replacement from the training set
  - For each node, randomly choose m features and calculate the best split
  - Fully grown and not pruned
- Use majority voting among all the trees

Random Forest Example

Original Dataset

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
No	No	No	125	No
Yes	Yes	Yes	180	Yes
Yes	Yes	No	210	No
Yes	No	Yes	167	Yes

Original Dataset

- Bootstrap with the same size of the original data

Bootstrap Dataset

	Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
1	No	No	No	125	No
2	Yes	Yes	Yes	180	Yes
3	Yes	Yes	No	210	No
4	Yes	No	Yes	167	Yes

Original Dataset

	Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
2	Yes	Yes	Yes	180	Yes
1	No	No	No	125	No
4	Yes	No	Yes	167	Yes
4	Yes	No	Yes	167	Yes

Bootstrapped Dataset

Random Forest Algorithm

- Create a decision tree using the bootstrapped dataset
- But ONLY use a random subset of variable (feature/columns) at each step
- For this example, let's use m=2
(2 randomly selected features)

Building Decision Tree

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Building Decision Tree

randomly select 2

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Bootstrapped Dataset

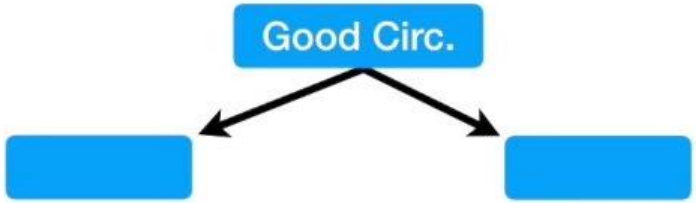
Building Decision Tree

Choose best splitting point
(for this example: blood circulation)

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Building Decision Tree

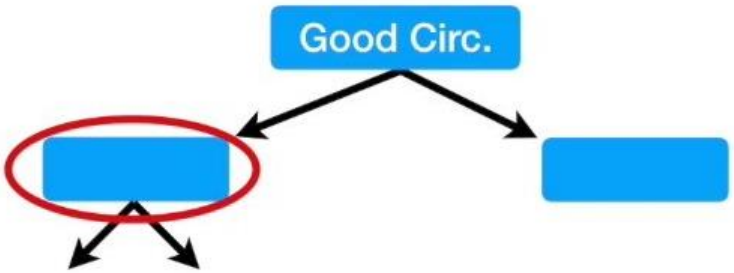


Remove the already chosen feature for the next step

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Building Decision Tree

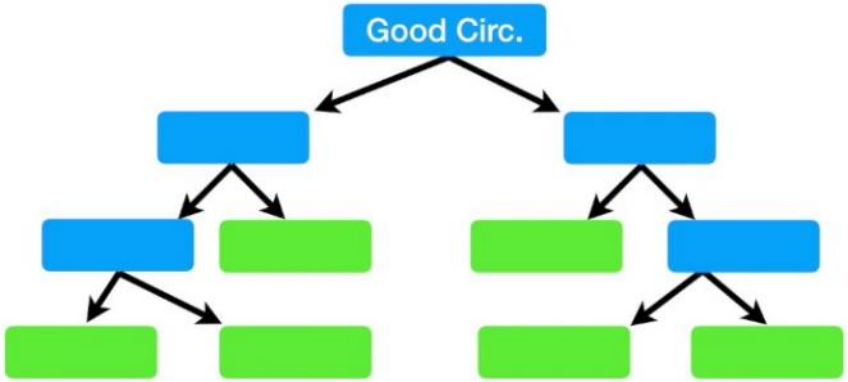


Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Repeat for the next Step,
Randomly select m feature from the remaining set

Building Decision Tree



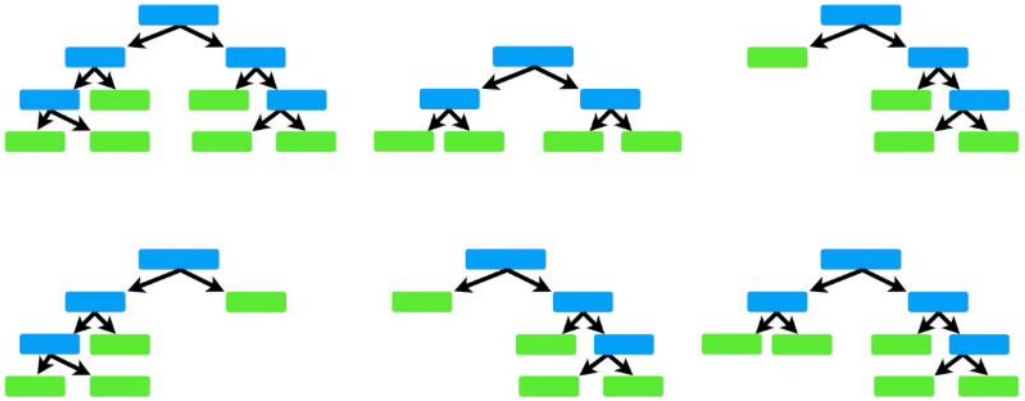
Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	180	Yes
No	No	No	125	No
Yes	No	Yes	167	Yes
Yes	No	Yes	167	Yes

Bootstrapped Dataset

Build the tree as usual, but only considering a random subset of feature at each step

Random Forest

- Repeat the process to make new trees
  - Make new bootstrapped dataset
  - Build a tree with only considering a random subset of features at each step

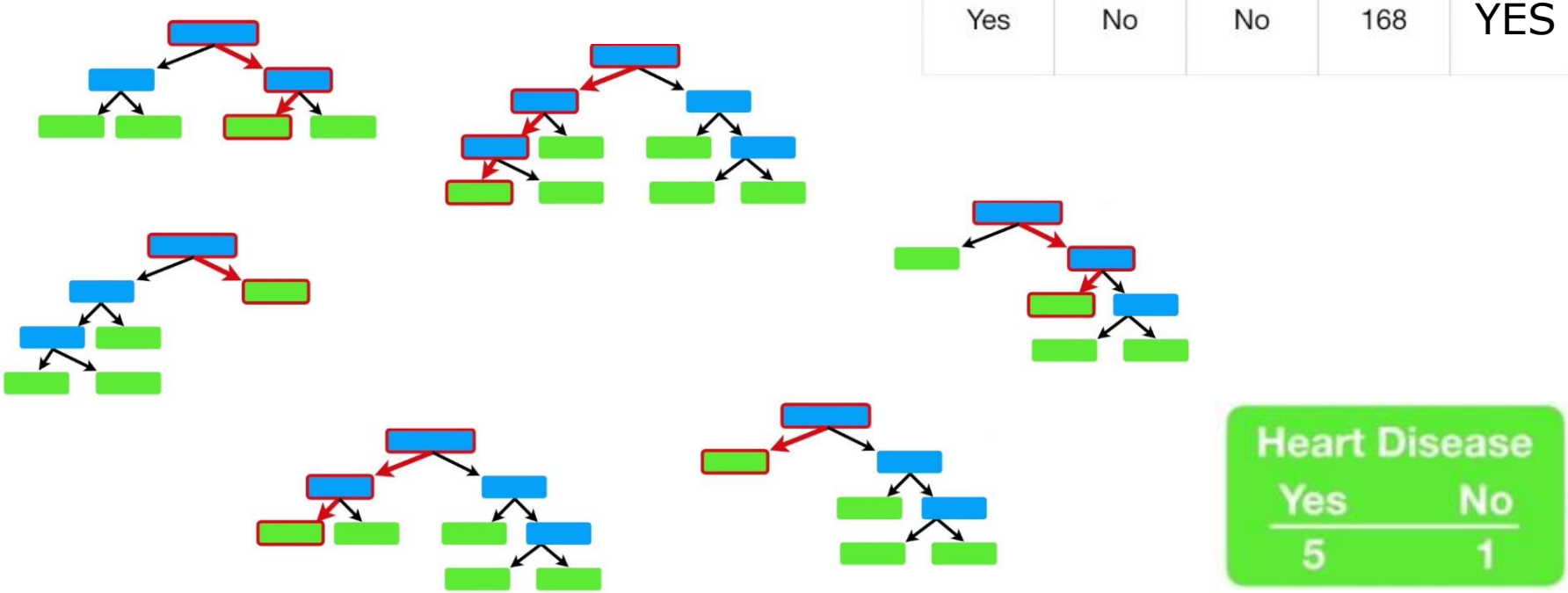


Random Forest Test Example

Test Example

- Test new Data to each trees

Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	No	No	168	YES



Random Forest Performance Test

Out of bag Data

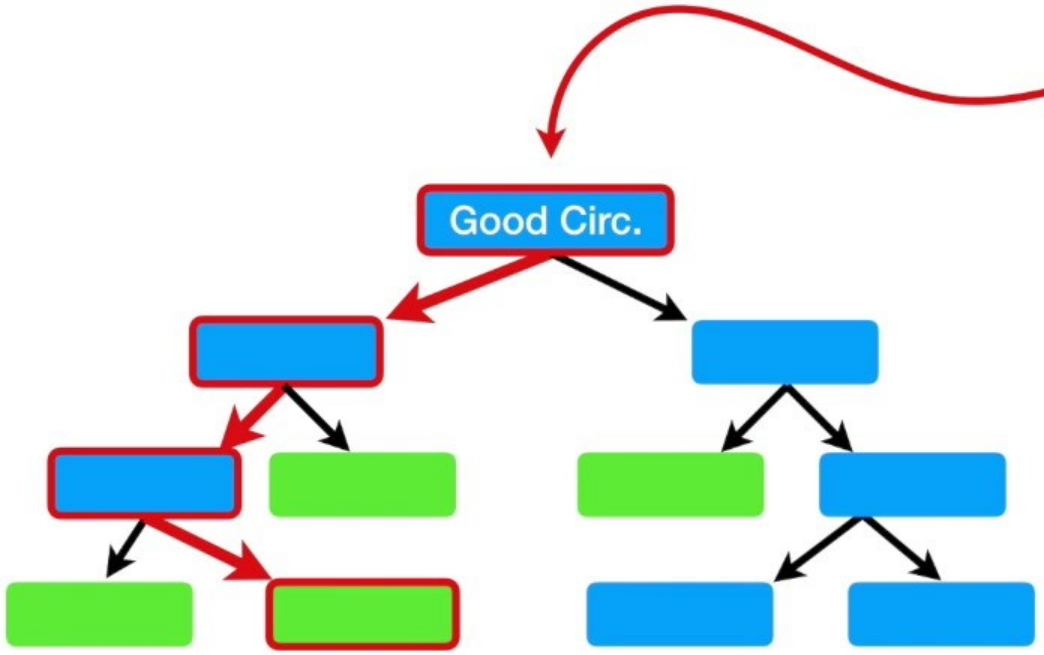
	Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
1	No	No	No	125	No
2	Yes	Yes	Yes	180	Yes
3	Yes	Yes	No	210	No
4	Yes	No	Yes	167	Yes

Original Dataset

	Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
2	Yes	Yes	Yes	180	Yes
1	No	No	No	125	No
4	Yes	No	Yes	167	Yes
4	Yes	No	Yes	167	Yes

Bootstrapped Dataset

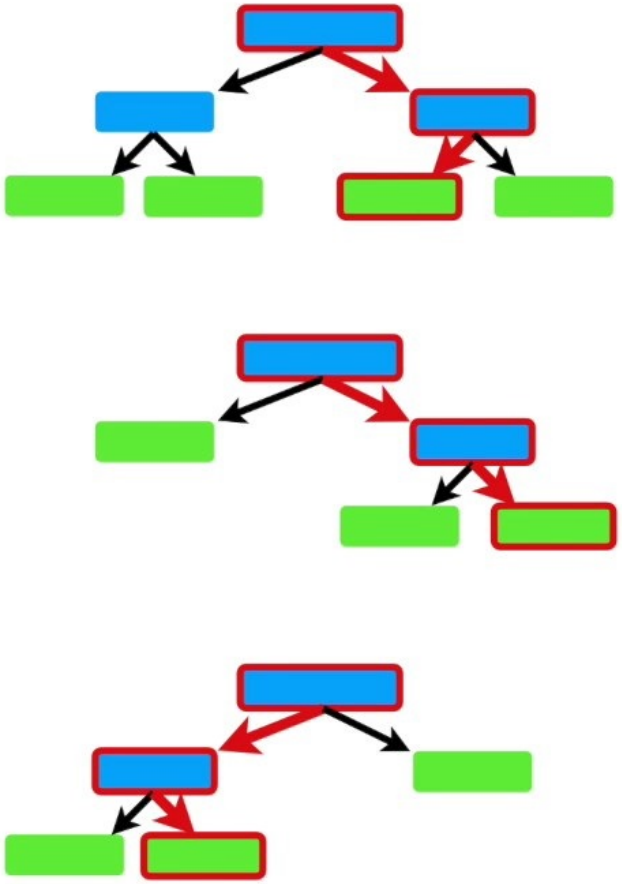
Out of bag Data



Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

Test the out-of-bag data to the forest

Out of bag Data



Chest Pain	Good Blood Circ.	Blocked Arteries	Weight	Heart Disease
Yes	Yes	No	210	No

Track the performance for all out-of-bag data

Random Forest Accuracy

- Measure how accurate the random forest by the proportion of out-of-bag sample that were correctly classified by the Random Forest
- Observe the $\boldsymbol{m}$ random sample and choose the RF with highest out-of-bag accuracy
  - Usually start with $m = \sqrt{M}$ or $m = \log_2 M$
  - M = total feature (dimension)

Random Forest Recap

Random Forest Recap

- Bagging+random features
- random forests tries to improve on bagging by "de-correlating" the trees.
  - Each tree has the same expectation.
- Improve accuracy
  - Incorporate more diversity and reduce variances
- Improve efficiency
  - Searching among subsets of features is much faster than searching among the complete set

Random Decision Tree

Random Decision Tree

- Single-model learning algorithms
  - Fix structure of the model, minimize some form of errors, or maximize data likelihood
    - (eg., Logistic regression, Naive Bayes, etc.)
  - Use some "free-form" functions to match the data given some "preference criteria" such as information gain, gini index and MDL.
    - (eg., Decision Tree, Rule-based Classifiers, etc.)

Random Decision Tree

- Such methods will make mistakes if
  - Data is insufficient
  - Structure of the model or the preference criteria is inappropriate for the problem
- Learning as Encoding
  - Make no assumption about the true model, neither parametric form nor free form
  - Do not prefer one base model over the other, just average them

Random Decision Tree Algorithm

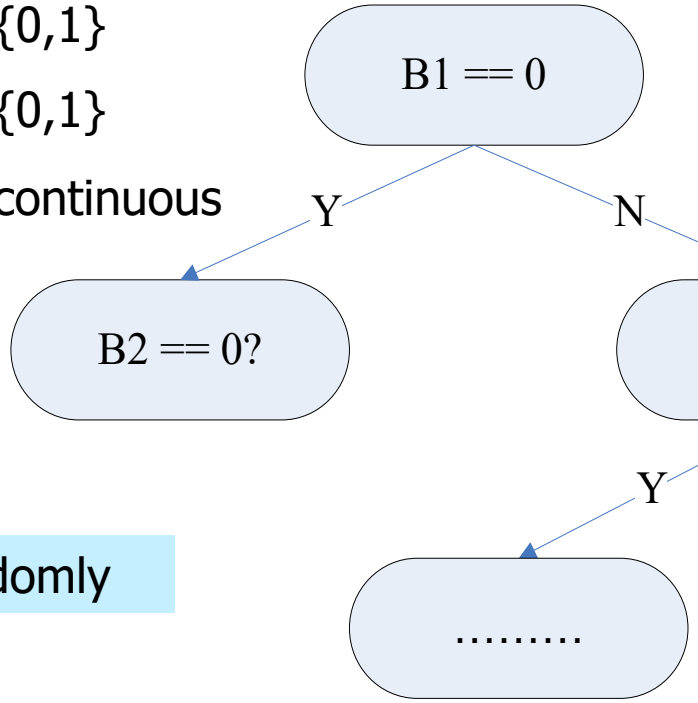
- At each node, an un-used feature is chosen randomly
  - A discrete feature is un-used if it has never been chosen previously on a given decision path starting from the root to the current node.
  - A continuous feature can be chosen multiple times on the same decision path, but each time a different threshold value is chosen
- We stop when one of the following happens:
  - A node becomes too small (<= 3 examples).
  - Or the total height of the tree exceeds some limits, such as the total number of features.
- Prediction
  - Simple averaging over multiple trees

Random Decision Tree Algorithm

B1: {0,1}

B2: {0,1}

B3: continuous



B1 chosen randomly

B2: {0,1}

B3: continuous

B3 chosen randomly

Random threshold 0.3

B2: {0,1}

B3: continuous

B2 chosen randomly

B3: continous

Random threshold 0.6

Random Decision Tree

- Potential Advantages
  - Training can be very efficient. Particularly true for very large datasets.
  - No cross-validation based estimation of parameters for some parametric methods.
  - Natural multi-class probability.
  - Imposes very little about the structures of the model.

Question?

Fakultas Informatika
School of Computing
Telkom University

THANK YOU