UNIVERSITY OF ARKANSAS

Random Forest

Jiahui Chen
Department of Mathematical Sciences
University of Arkansas

Introduction

- It is ensemble learning method and one of the top ten methods in data science.
- Random Forest proposed by Tin Kam Ho in 1995 and Breiman in 2001--- It is a top method now.
- It is designed to reduce the overfitting in the original decision trees.
- Random Forest classifier and regressor are available.
- The first choice for machine learning beginners

Introduction

- Random forest builds a large number of de-correlated trees.
- Use **CART** (Classification And Regression Tree) to grow a tree.
  - Binary split
  - GINI to measure impurity
- Combine Breiman's Bagging (**B**ootstrap **AGG**regat**ING**)
- Do aggregating
- Predictions are given by the mean values of all trees.

Ensemble

- Ensemble Methods: a broad category of techniques that involve combining the decisions from multiple models to improve the overall performance
- Idea: a group of weak learners can come together to form a strong learner
- Both **boosting** and **bagging** are specific types of ensemble methods
- Bagging (Bootstrap Aggregating): generating multiple versions of a predictor and using these to get an aggregated predictor
- Boosting: sequentially training models where each model tries to correct the errors made by the previous ones

BAGGING

- **Bootstrap**: Randomly sample the original data with replacement
- **Aggregating:** Use consensus to make a decision (**wisdom of the crowd** phenomena)

Example: original data = {1, 2, 4 , 6, 10}
Via Bootstrap -> new data = {1, 1, 1, 6, 10}
Via bootstrap -> new data = { 1, 2, 4, 4, 4,4}

Algorithm

Training set: $\mathcal{D} = \left\{\left(\mathbf{x}^{(i)}, y^{(i)}\right) \middle| \mathbf{x}^{(i)} \in \mathbb{R}^n, y^{(i)} \in \{0,1\}\right\}_{i=1}^{M}$

Each tree is constructed using the following algorithm:

Let the number of training labels be M, and the number of features in the data be n.

1. Select the number n_{tree} of tress to be grown
2. Select the number $n_{features}$ of input features to be used to determine the decision at a node of the tree; $n_{features} << M$.

Algorithm

Continue

3. Randomly choose a set of samples for this tree (i.e. take a bootstrap sample). Use the rest of the labels and features to estimate the error of the tree.

4. For each node of the tree, randomly choose $n_{features}$ features to make the decision at that node.

5. Calculate the best split based on these $n_{features}$ features in the tree set using the GINI index.

6. No used feature is to be reused in each tree.

7. Each tree is fully grown and not pruned (as may be done in constructing a normal tree classifier).

For prediction, a new sample is pushed down a tree, which gives a prediction. This procedure is iterated over all trees in the ensemble, and the average vote of all trees is reported as random forest prediction.

Important Parameters in Random Forest

- Number of trees (n_{tree})
- Number of features $\left(n_{features}\right)$ used to grow a decision tree
  - At each node, consider randomly subset of $n_{features}$ features to choose the best split
- Try-and-error approach to determine n_{tree} and $n_{features}$ for a given problem.

Example

Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Cool	Normal	Weak	Yes

Quantification

Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	0	2	1	0	0
D2	0	2	1	1	0
D3	1	2	1	0	1
D4	2	1	1	0	1
D5	2	0	0	0	1
D6	2	0	0	1	0
D7	1	0	0	1	1
D8	0	0	0	0	1

Sunny = 0, Overcast = 1, Rain =2
Cool = 0, Mild = 1, Hot = 2
Normal = 0, High = 1
Weak = 0 ,Strong = 1
No = 0, Yes = 1

For more accurate setting, one should use one hot code

Tree Building

Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	0	2	1	0	0
D2	0	2	1	1	0
D3	1	2	1	0	1
D4	2	1	1	0	1
D5	2	0	0	0	1
D6	2	0	0	1	0
D7	1	0	0	1	1
D8	0	0	0	0	1

We want to build a 3-tree random forest. By using bootstrap, the samples in each tree are:

Tree 1: {D6, D4, D7, D6, D5, D7, D3, D6}

Tree 2: {D4, D6, D4, D5, D1, D1, D2, D6}

Tree 3: {D1, D1, D4, D6, D8, D4, D4, D8}

Tree Building

- We are now building decision tree for Tree 1 = {D6, D4, D7, D6, D5, D7, D3, D6} with max features = 2
- At the root node, we randomly pick two features to decide which one will give the best split. Assume they are {X[0]: Outlook, X[2]: Humidity}. We have 3 choices to split the root node:

Outlook ≤ 0.5,

Outlook ≤ 1.5, and

Humidity ≤ 0.5

GAIN = GINI(parent) – weighted average GINI of children

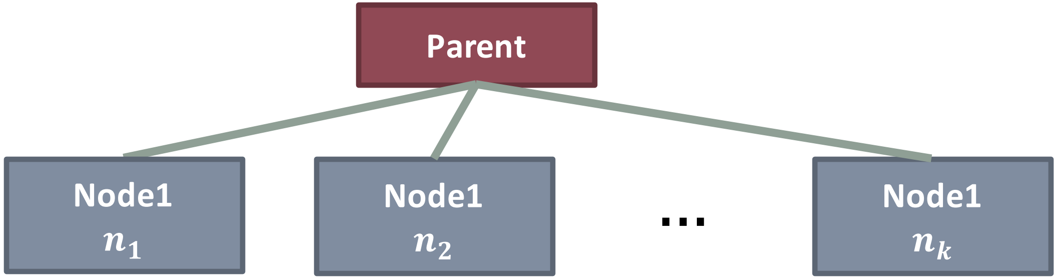
GINI is impurity

GINI Index and Gain in Splitting

Gini index for a given node $\boldsymbol{t}$

$$\mathbf{GINI}(\boldsymbol{t}) = \sum_{j} \boldsymbol{p}(\boldsymbol{j}|\boldsymbol{t})(\mathbf{1} - \boldsymbol{p}(\boldsymbol{j}|\boldsymbol{t}))$$

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



$$\text{Gain} = \text{Gini(Parent)} - \frac{n_1}{\sum n_i}\text{Gini(Node 1)} - \frac{n_2}{\sum n_i}\text{Gini(Node 2)} - \cdots - \frac{n_k}{\sum n_i}\text{Gini(Node k)}$$

Day	Outlook X[0]	Temperature X[1]	Humidity X[2]	Wind X[3]	Play ball
D1	0	2	1	0	0
D2	0	2	1	1	0
D3	1	2	1	0	1
D4	2	1	1	0	1
D5	2	0	0	0	1
D6	2	0	0	1	0
D7	1	0	0	1	1
D8	0	0	0	0	1

Tree 1 : {D6, D4, D7, D6, D5, D7, D3, D6}

Outlook $\leq$ 0.5? {F, F, F, F, F, F, F, F} (0,8)

Outlook $\leq$ 1.5? {F, F, T, F, F, T, T, F} (3,5)

Humidity $\leq$ 0.5? {T, F, T, T, T, T, F, T} (6,2)

- Outlook $\leq$ 0.5 gives split: [0,8], GINI=1-0-1=0
- Outlook $\leq$ 1.5 gives split: [3,5], GINI=$1-\left(\frac{3}{8}\right)^2-\left(\frac{5}{8}\right)^2=0.469$
- Humidity $\leq$ 0.5 gives split: [6,2], GINI=$1-\left(\frac{6}{8}\right)^2-\left(\frac{2}{8}\right)^2=0.375$

Day	Outlook X[0]	Temperature X[1]	Humidity X[2]	Wind X[3]	Play ball
D1	0	2	1	0	0
D2	0	2	1	1	0
D3	1	2	1	0	1
D4	2	1	1	0	1
D5	2	0	0	0	1
D6	2	0	0	1	0
D7	1	0	0	1	1
D8	0	0	0	0	1

Tree 1 : {D6, D4, D7, D6, D5, D7, D3, D6}

Outlook $\leq$ 1.5? {F, F, T, F, F, T, T, F} (3,5)

- Outlook $\leq$ 1.5 gives split: [3,5],

GAIN=0.46875 – (3/8)*(0) – (5/8)*0.48=0.16875

- GINI({D6, D4, D7, D6, D5, D7, D3, D6})=1-(3/8)^2 – (5/8)^2 = 0.46875
- Left node = {D7, D7, D3}, label 0: 0, label 1: 3

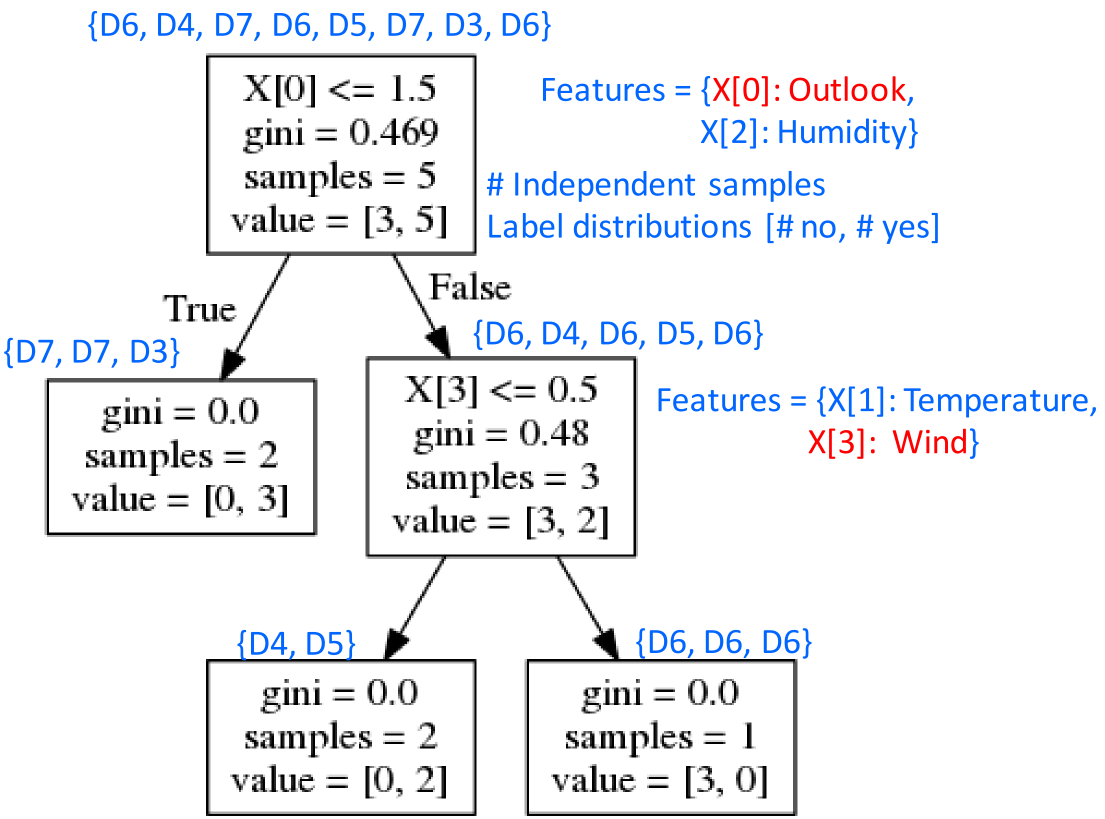
GINI(left node) = 1-(0/3)^2 –(3/3)^2=0

- Right node = {D6, D6, D6, D4, D5}, labe l 0:3, label 1:2

GINI(right node) = 1 – (3/5)^2 – (2/5)^2 = 0.48

Tree Building

- The fully grown of Tree 1



Tree Building

Day	Outlook X[0]	Temperature X[1]	Humidity X[2]	Wind X[3]	Play ball
D1	0	2	1	0	0
D2	0	2	1	1	0
D3	1	2	1	0	1
D4	2	1	1	0	1
D5	2	0	0	0	1
D6	2	0	0	1	0
D7	1	0	0	1	1
D8	0	0	0	0	1

Node 2 {D6, D4, D6, D5, D6}

Temp ≤ 0.5? {T, F, T, T, T} (4,1)

Temp ≤ 1.5? {T, T, T, T, T} (5,0)

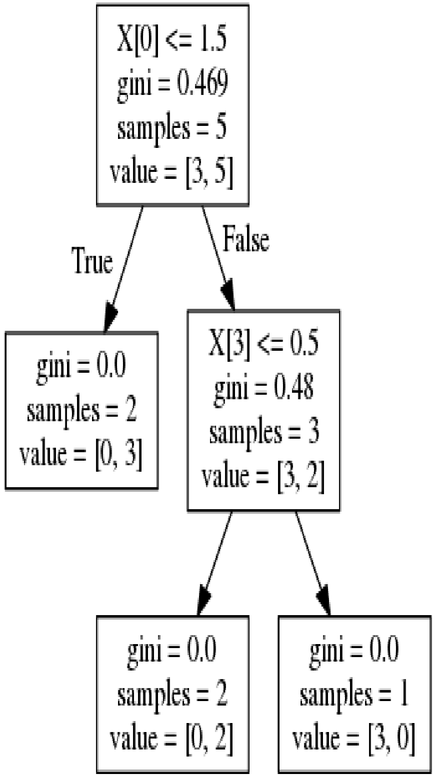
Wind ≤ 0.5? {F, T, F, T, F} (2,3)

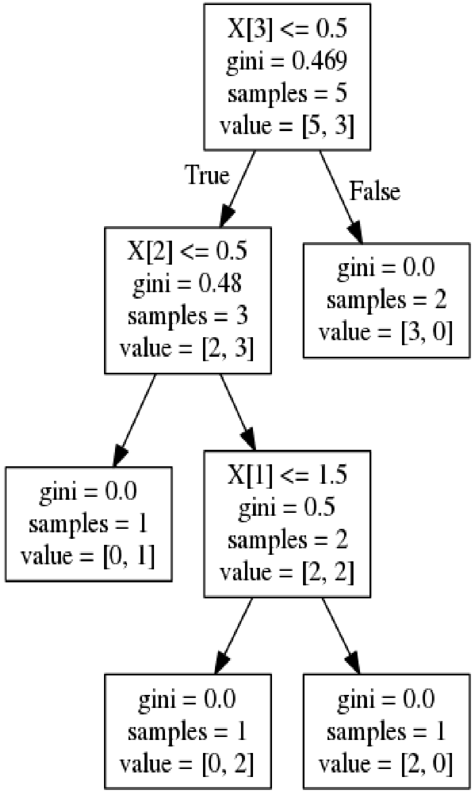
- A complete Random Forest for Example with number of trees = 3, max features = 2

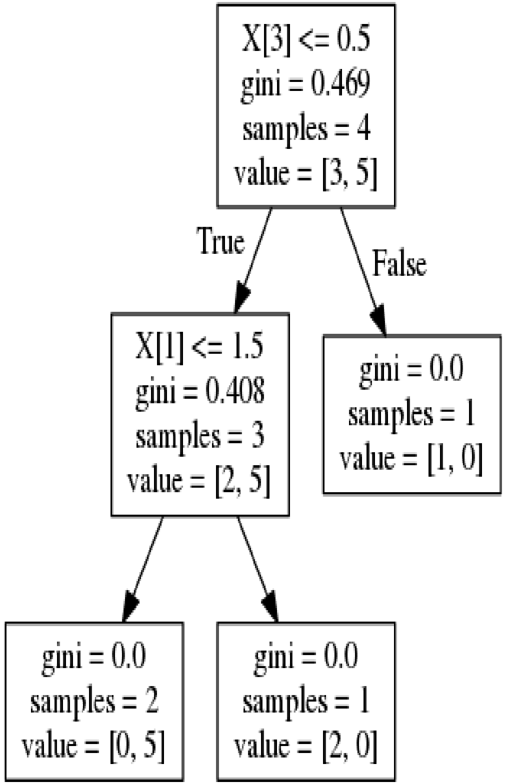
Tree 1={D6, D4, D7, D6, D5, D7, D3, D6}

Tree 2={D4, D6, D4, D5, D1, D1, D2, D6}

Tree 3={D1, D1, D4, D6, D8, D4, D4, D8}







Useful Features in Random Forest

- Out-of-bag error (OOB error) for feature importance
- Feature Importance: Useful in answer physical questions. This is not typically available in deep learning.

OOB Error

- OOB error is the mean prediction error on the training sample x_i using only trees that do not have x_i in their bootstrap sample.
- OOB error can be calculated only after obtaining a trained tree
- OOB error is an optional information used for tuning the RF parameters

Feature Importance

- Two approaches
- 1st approach: Mean decrease impurity
  - For each feature, look for nodes using that feature for the best split and then calculate the weighted average gain
  - Feature which has the largest averaged gain is the most important one
- 2nd approach: Mean decrease accuracy.
  - For each feature, permutes it across the training data. Calculate the OOB error for that feature, then compare this error to the original one
  - Feature which has the most change in the OOB error is the most important one.

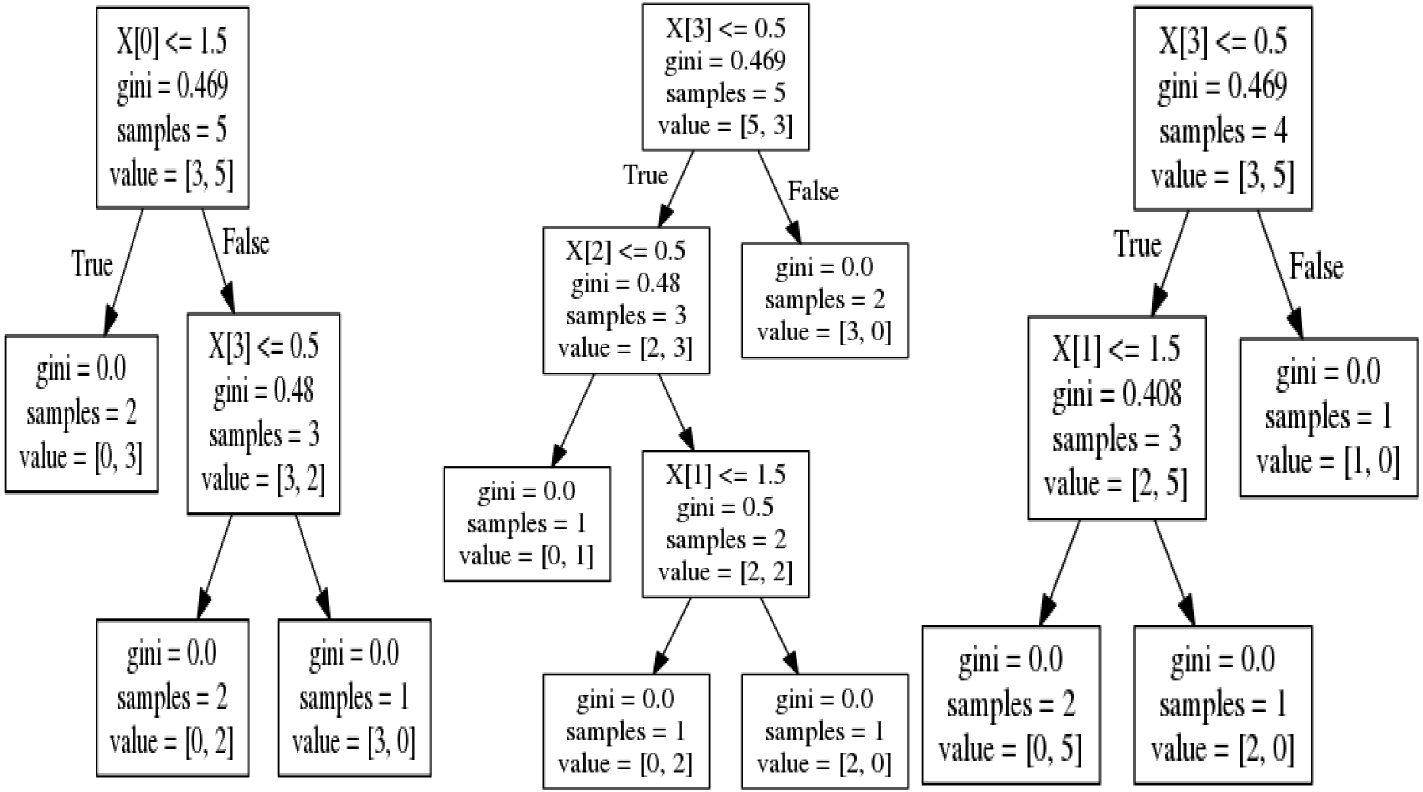
UNIVERSITY OF ARKANSAS

- A complete Random Forest for Example with number of trees = 3, max features = 2

Tree 1={D6, D4, D7, D6, D5, D7, D3, D6}

Tree 2={D4, D6, D4, D5, D1, D1, D2, D6}

Tree 3={D1, D1, D4, D6, D8, D4, D4, D8}



Feature Importance

- 1st approach: Mean decrease impurity
  - We determine the feature importance ranking of {outlook, temperature, humidity, wind}
  - Calculate the weighted average gain

X[0]: $\text{Outlook} = \left(0.469 - 0.48 \times \frac{5}{8}\right) \times \frac{8}{8} = 0.169$

X[1]: $\text{Temp} = 0.5 \times \frac{4}{8} + 0.408 \times \frac{7}{8} = 0.607$

X[2]: $\text{Humidity} = \left(0.48 - 0.5 \times \frac{4}{5}\right) \times \frac{5}{8} = 0.05$

X[3]: Wind

$$= 0.48 \times \frac{5}{8} + \left(0.469 - 0.48 \times \frac{5}{8}\right) + \left(0.469 - 0.408 \times \frac{7}{8}\right)$$

$$= 0.581$$

Temp > Wind > Outlook > Humidity

OOB Error

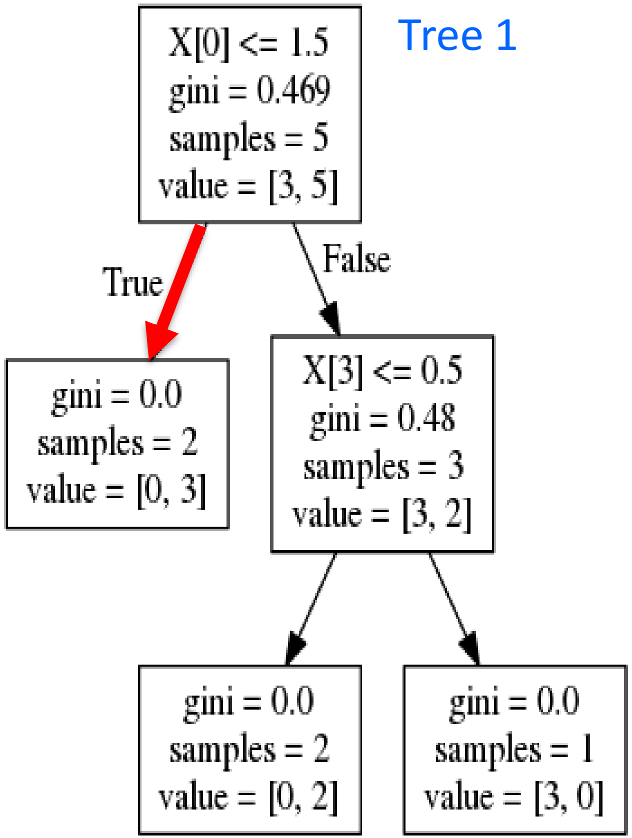
- <u>2nd approach:</u> Mean decrease accuracy
  - Calculate OOB error

Tree 1: {D6, D4, D7, D6, D5, D7, D3, D6}

Tree 2: {D4, D6, D4, D5, D1, D1, D2, D6}

Tree 3: {D1, D1, D4, D6, D8, D4, D4, D8}

- Predict D1=($\mathbf{x}^{(1)}$, 0) using Tree 1,
  - $\mathbf{x}^{(1)} = (0,2,1,0)^T$
  - We get $\hat{y}^{(1)} = 1$, (wrong)



OOB Error

- 2nd approach: Mean decrease accuracy
  - Calculate OOB error

Tree 1: {D6, D4, D7, D6, D5, D7, D3, D6}

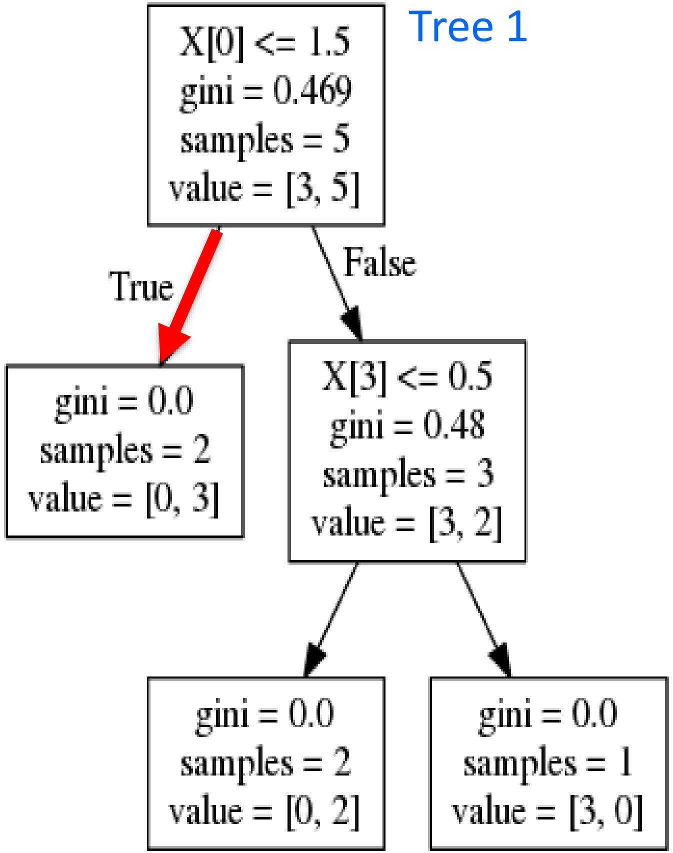
Tree 2: {D4, D6, D4, D5, D1, D1, D2, D6}

Tree 3: {D1, D1, D4, D6, D8, D4, D4, D8}

- Predict D2=($\mathbf{x}^{(2)}, 0$) using Tree 1 (and Tree 3):

$\mathbf{x}^{(2)} = (0,2,1,1)^T$

We get $\hat{y}^{(2)} = 1$ (wrong)



OOB Error

- <u>2nd approach:</u> Mean decrease accuracy
  - Calculate OOB error

Tree 1: {D6, D4, D7, D6, D5, D7, D3, D6}

Tree 2: {D4, D6, D4, D5, D1, D1, D2, D6}

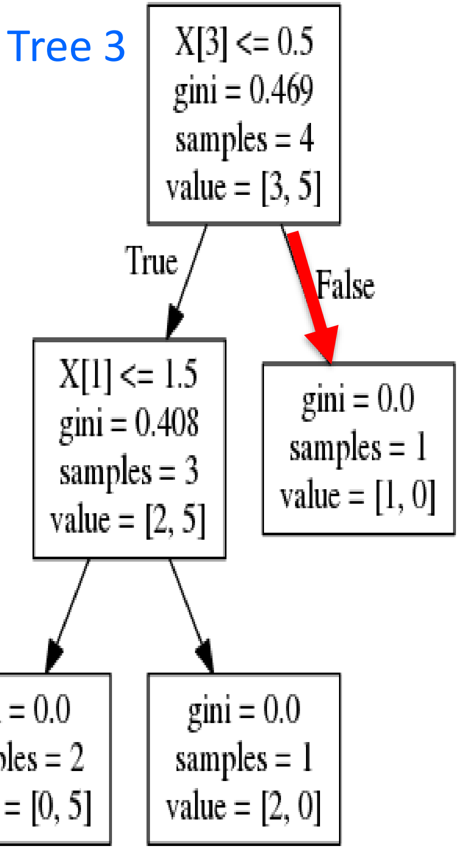
Tree 3: {D1, D1, D4, D6, D8, D4, D4, D8}

- Predict D2=($\mathbf{x}^{(2)}$, 0) using Tree 3:

$\mathbf{x}^{(2)} = (0,2,1,1)^T$,

we get $\hat{y}^{(2)} = 0$, wrong

Tree 1 gives $\hat{y}^{(2)} = 1$ and Tree 3 gives $\hat{y}^{(2)} = 0$, we choose 1 (wrong) since the leaf node in Tree 1 has 3 samples while one in Tree 3 has only 1 sample. In the extreme case, the sample distribution is the same among the leaf nodes. We pick the positive label.



OOB Error

- 2nd approach: Mean decrease accuracy
  - Calculate OOB error -- Summary

Tree 1: {D6, D4, D7, D6, D5, D7, D3, D6}

Tree 2: {D4, D6, D4, D5, D1, D1, D2, D6}

Tree 3: {D1, D1, D4, D6, D8, D4, D4, D8}

- Predict D1=$(\mathbf{x}^{(1)}, 0)$ using Tree 1, we get $\hat{y}^{(1)} = 1$, wrong
- Predict D2=$(\mathbf{x}^{(2)}, 0)$ using Tree 1 and Tree 3 we get $\hat{y}^{(2)} = 1$, wrong
- Predict D3=$(\mathbf{x}^{(3)}, 1)$ using Tree 2 and Tree 3 we get $\hat{y}^{(3)} = 0$, wrong
- Predict D4=$(\mathbf{x}^{(4)}, 1)$ but no tree available. In general, if we generate a considerable large amount of trees, we rarely encounter this issue. However, in this situation, we may exclude D4 when calculating OOB error

OOB Error

- 2nd approach: Mean decrease accuracy
  - Calculate OOB error– Summary (continue)

Tree 1: {D6, D4, D7, D6, D5, D7, D3, D6}

Tree 2: {D4, D6, D4, D5, D1, D1, D2, D6}

Tree 3: {D1, D1, D4, D6, D8, D4, D4, D8}

- Predict D5=$(\mathbf{x}^{(5)}, 1)$ using Tree 3 we get $\hat{y}^{(5)} = 1$, correct
- Predict D6=$(\mathbf{x}^{(6)}, 0)$: not applicable
- Predict D7=$(\mathbf{x}^{(7)}, 1)$ using Tree 2 and 3 we get $\hat{y}^{(7)} = 0$, wrong
- Predict D8=$(\mathbf{x}^{(8)}, 1)$ using Tree 1 and 2 we get $\hat{y}^{(8)}=1$, correct

So OOB_error =4/6

Mean Decrease Accuracy

- Permute Outlook feature

Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	0 → 2	2	1	0	0
D2	0 → 2	2	1	1	0
D3	1 → 0	2	1	0	1
D4	2 → 1	1	1	0	1
D5	2 → 1	0	0	0	1
D6	2 → 0	0	0	1	0
D7	1 → 0	0	0	1	1
D8	0 → 2	0	0	0	1

- Calculate the change in OOB error

$$\Delta OOB_{error}^{Outlook} = \left| OOB_{error}^{Outlook} - OOB_{error} \right|$$

- Similarly, one can determine $\Delta OOB_{error}^{Temp}, \Delta OOB_{error}^{Humidity}, \Delta OOB_{error}^{Humidity}, \Delta OOB_{error}^{Wind}$
- We rank the features based on ΔOOB_{error}^{*}

Discussions

- It is simple and easy to use but might not be the most accurate method.
- Overfitting is typically not a problem.
- Unsupervised learning with Random forest
- ExtraTrees: *extremely randomized trees:* additionally the top-down splitting in the tree learner is randomized. Instead of computing the locally *optimal* feature/split using GINI.
- Related to KNN (why?)
- It can be extended to multinomial logistic regression and naive Bayes classifiers
- Kernel random forest