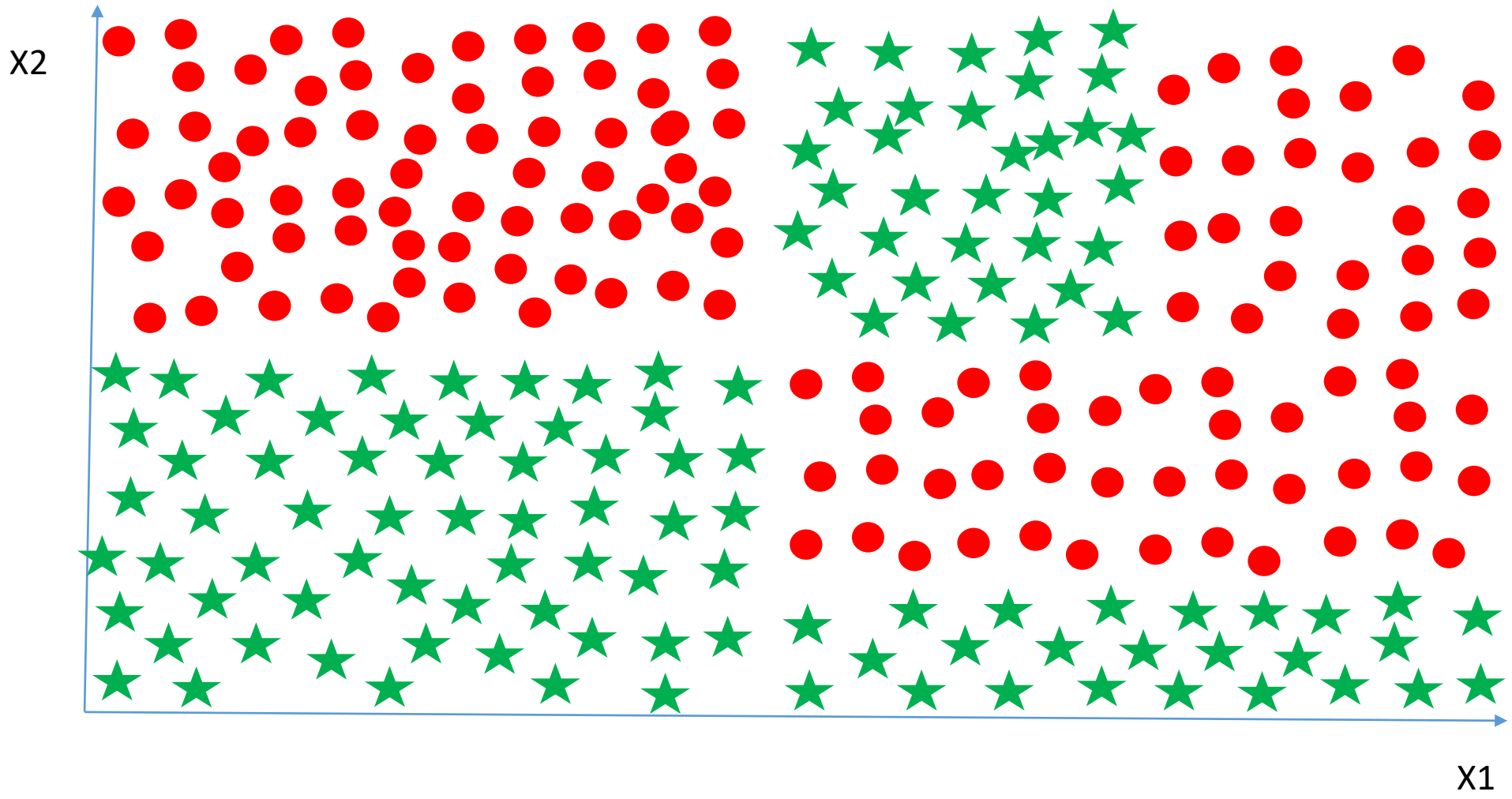
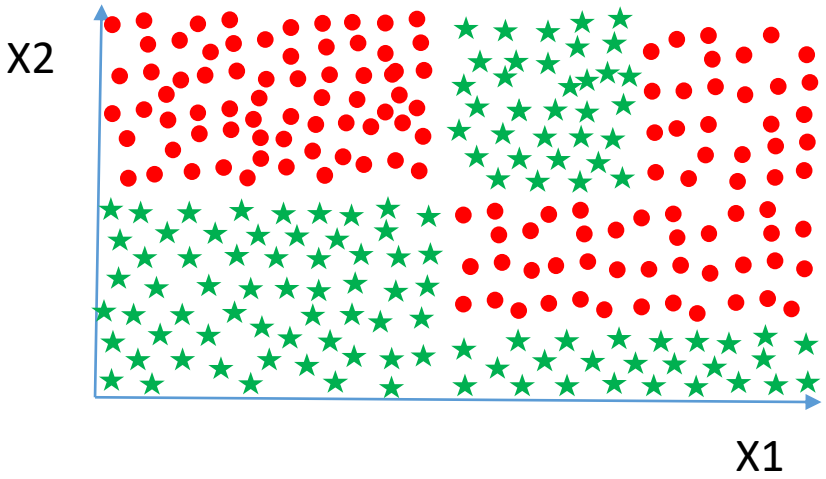
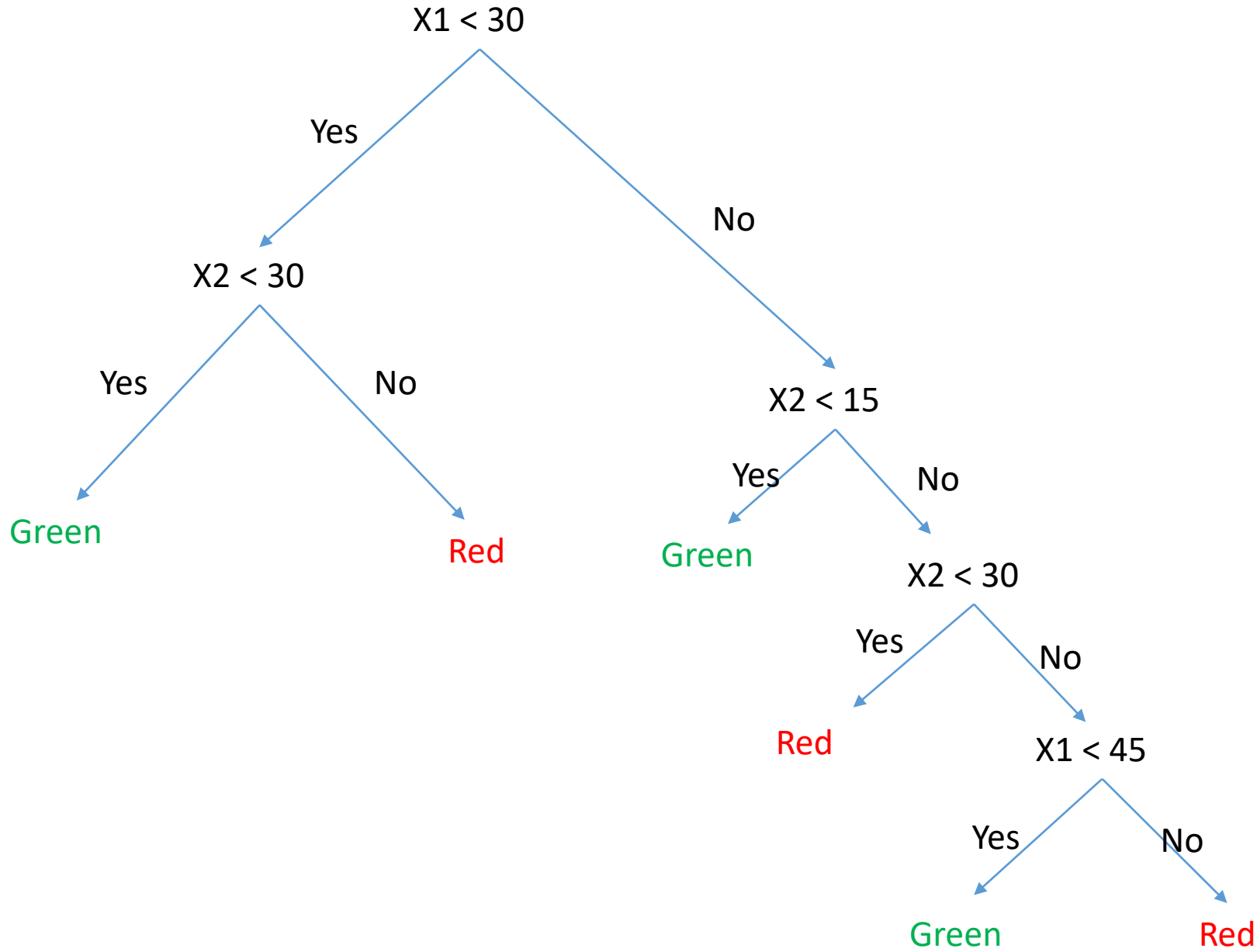
Decision Tree

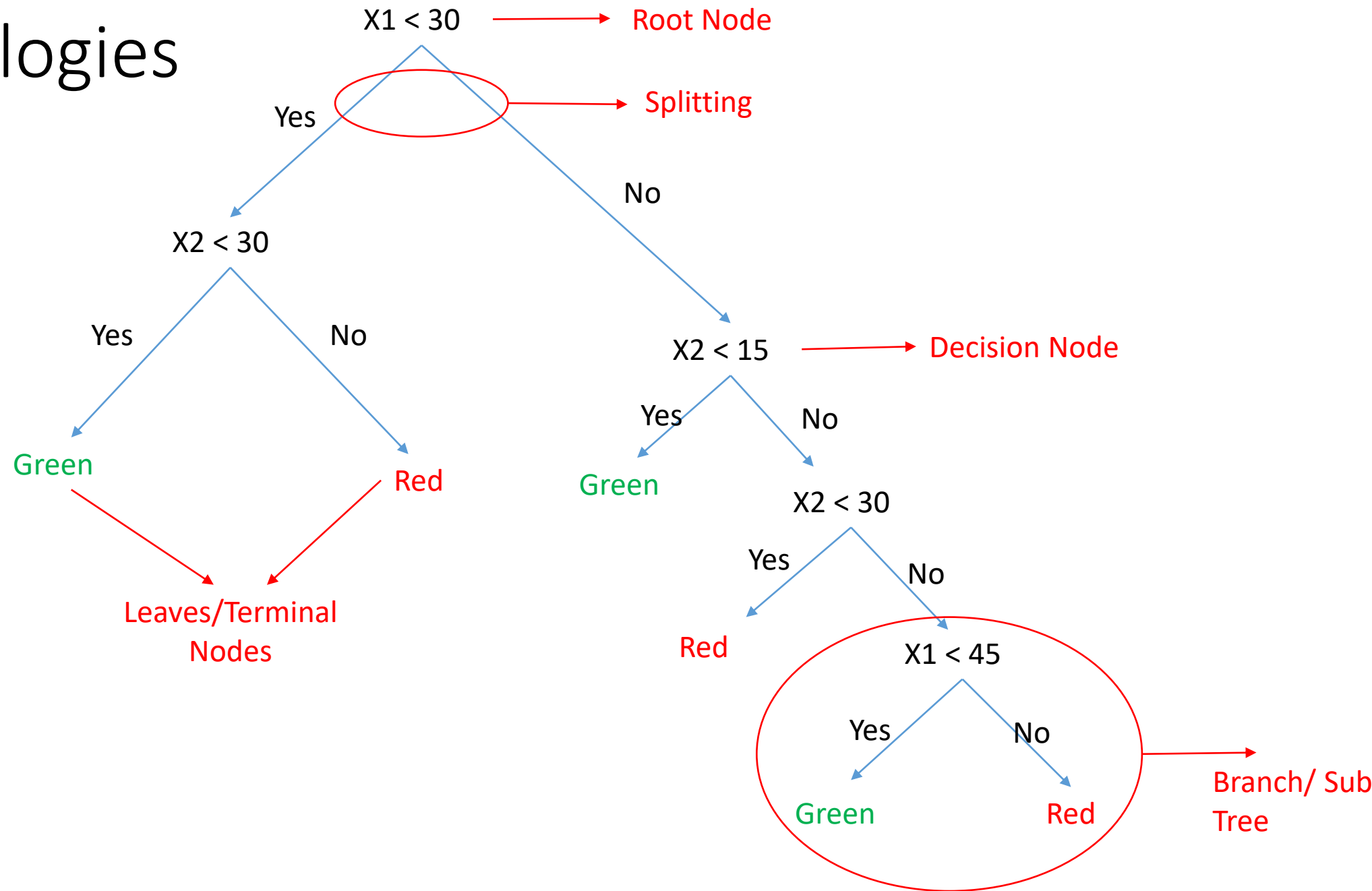
Decision Trees - Intuition



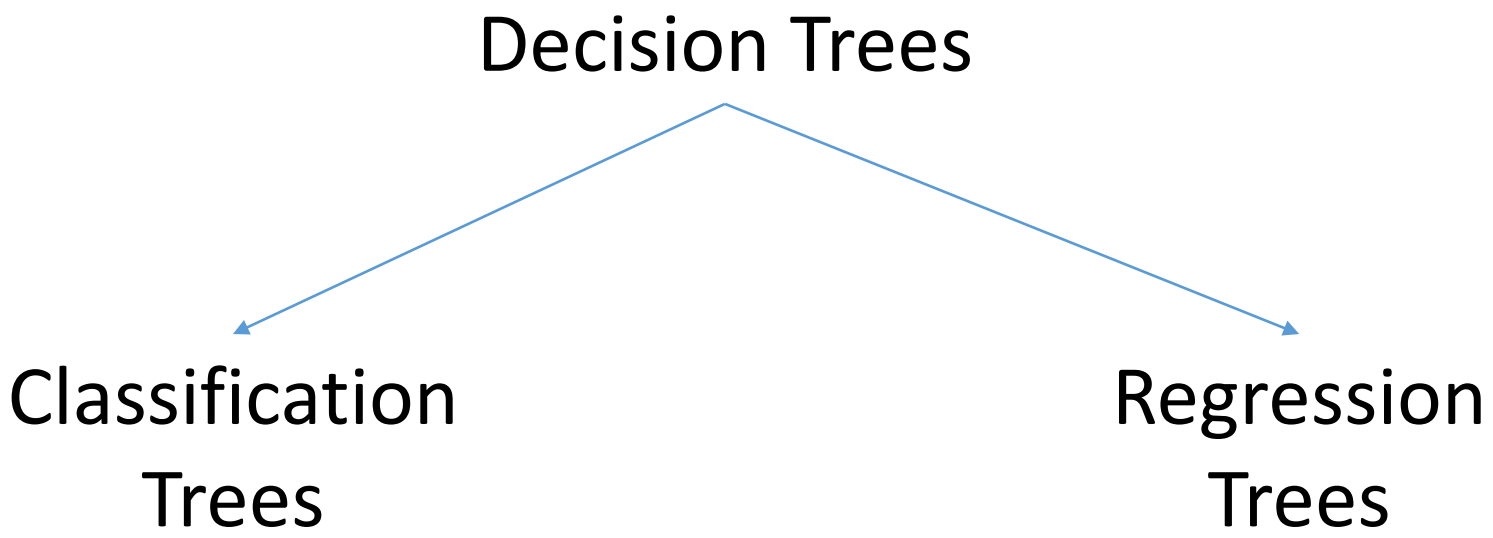

X2
X1


X1 < 30
Yes
No
X2 < 30
Yes
No
Green
Red
X2 < 15
Yes
No
Green
X2 < 30
Yes
No
Red
X1 < 45
Yes
No
Green
Red

Terminologies

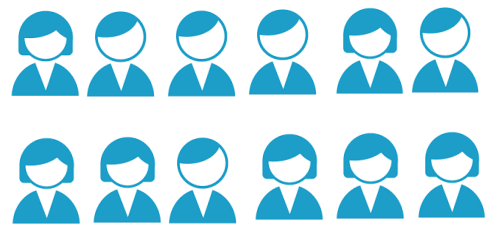


Types of Decision Trees



What if we have more dimensions?

Lets assume we are considering a sample of 30 people, 15 of which suffer from diabetes, and we want to create a model to predict those who are susceptible to diabetes. Lets suppose we are judging them on three parameters – Gender, Age and BMI



Population = 100
Diabetic = 50 (50%)

Split on Gender

Female (20)

Diabetic = 5 (25%)

Male (80)

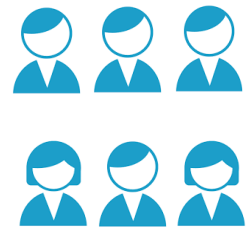
Diabetic = 40 (50%)

Split on Age

< 45 years (40)

Diabetic = 15 (38%)

>= 45 years (60)



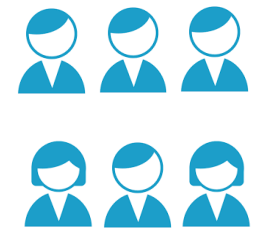
Diabetic = 20 (33%)

Split on BMI

< 23 (75)

Diabetic = 30 (40%)

>= 23 (25)



Diabetic = 15 (60%)

How does a Decision Tree decide where to split?

Decision Trees look for homogeneity. Decision tree splits the nodes on all available variables/features and then selects the split which results in most homogeneous sub-nodes.

Classification Trees:

- Gini Index
- Entropy

Regression Trees:

- Reduction in variance

Gini Index

- Measure of homogeneity
- Higher homogeneity = Higher Gini
- Gini = $p^2 + q^2$
- Calculate Gini index for each of the sub-nodes
- Finally, Gini for a split = weighted Gini scores for each of the sub-nodes

Entropy

- Measure of disorganization
- Higher homogeneity = Lower Gini
- Entropy = $- p\log_2 p - q\log_2 q$
- Calculate Entropy for each of the sub-nodes
- Finally, Entropy for a split = weighted Entropy for each of the sub-nodes

Decision Trees – Advantages & Disadvantages

Advantages

- Easy to understand
- When distribution is not linearly separable
- Useful for both classification & regression
- Not influenced by outliers or missing values
- Helps in identifying significant variables

Disadvantages

- Prone to overfitting

How to control overfitting in Decision Trees?

- Restricting tree size
- Pruning
- Random Forest

Overfitting - Restricting Tree Size

- Maximum Tree Depth (n)
- Maximum number of Terminal Nodes/Leaves (2^n)
- Minimum samples for node split
- Minimum samples for a Terminal Node/Leaf
- Maximum features to consider for a split

Overfitting - Pruning

- Restricting tree size is a greedy approach!
- What is greedy approach?

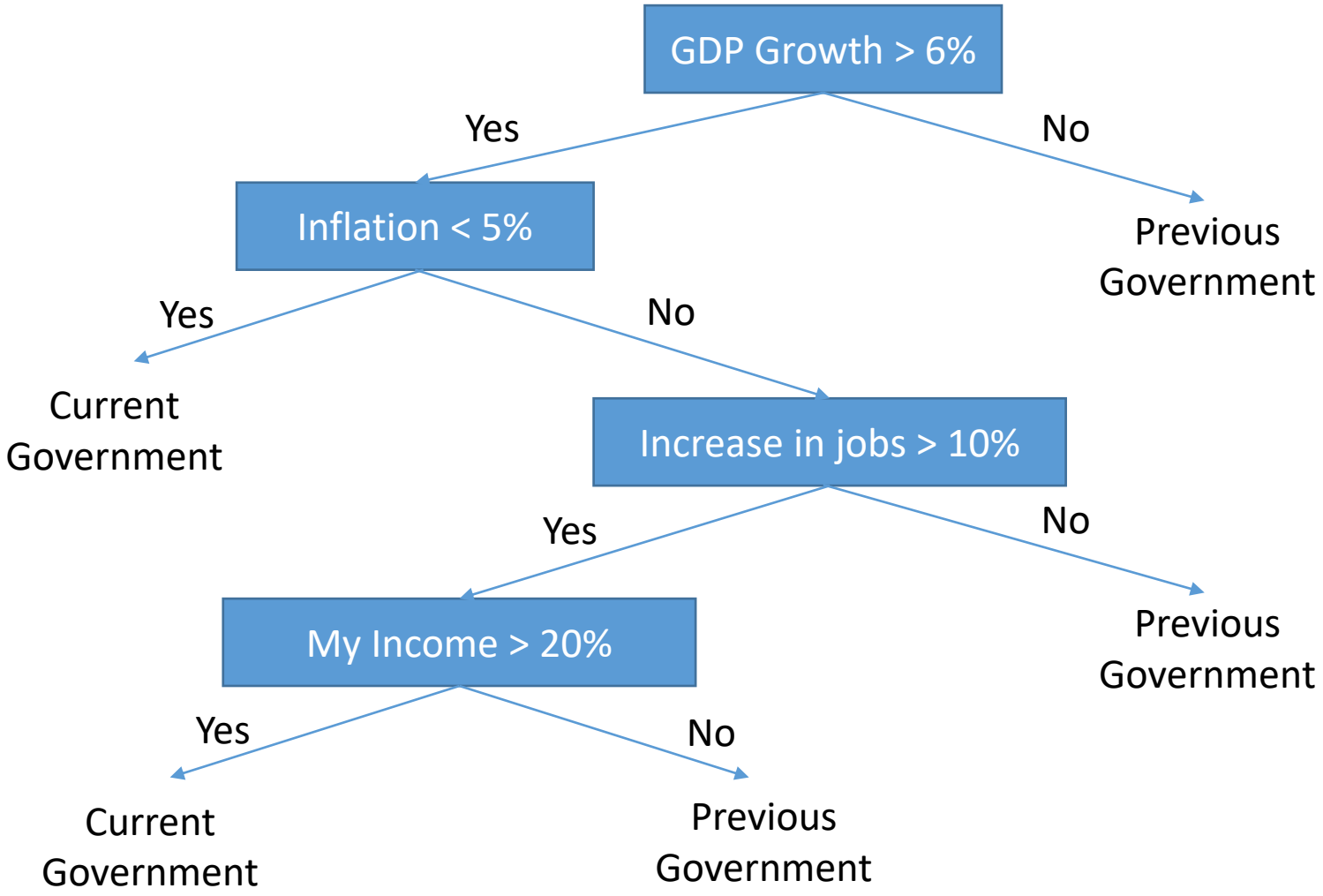
Overfitting - Pruning

- First make a decision tree grow to a large depth
- Then from bottom, start removing leaves which give negative returns compared to the top
- Suppose entropy for a split is 0.6 and for the next split is 0.2. If we use techniques of size restriction at the first split, we might stop at a non-optimal point
- Hence, consider pruning

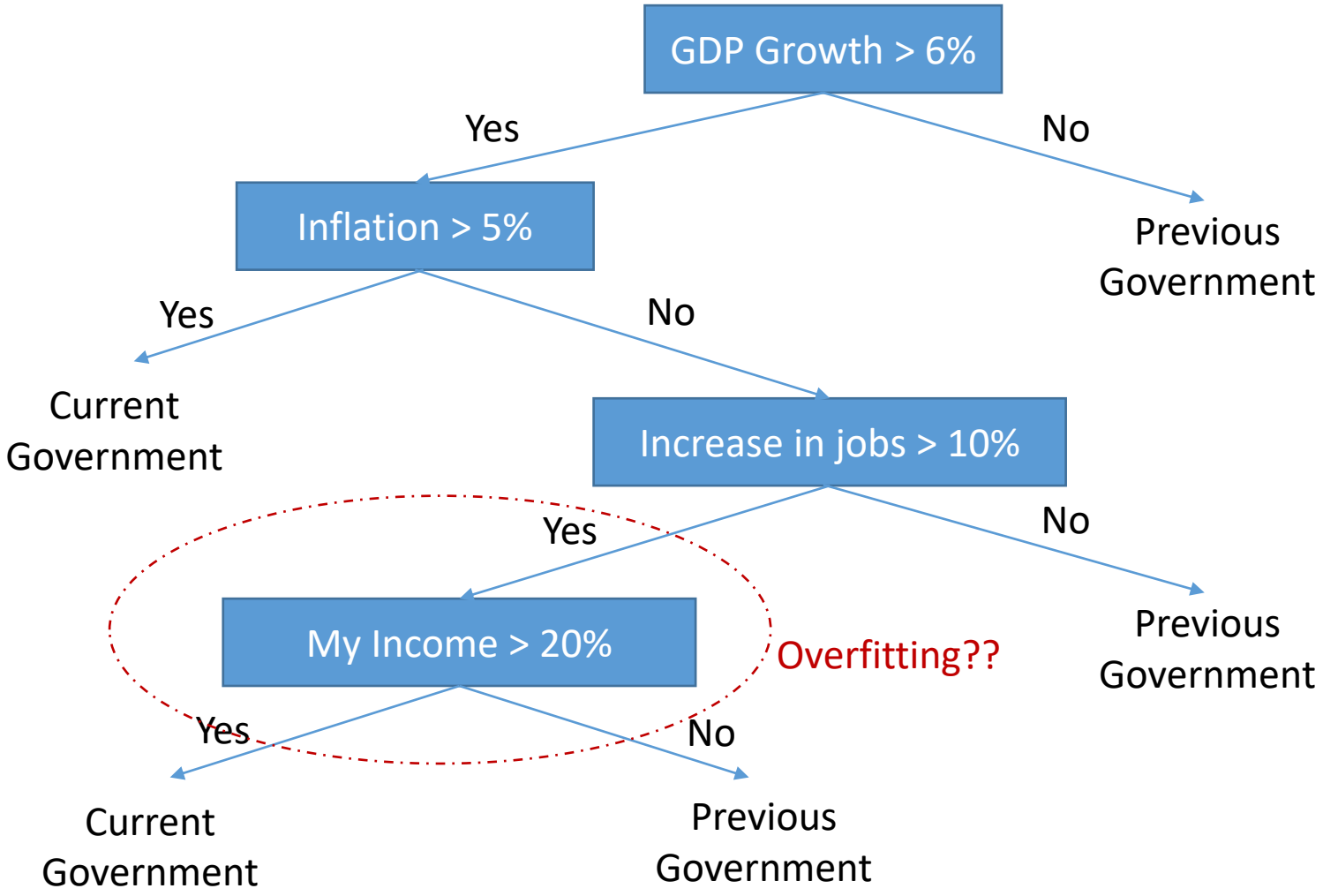
Overfitting – Random Forest

Random Forest

Who wins the next election?



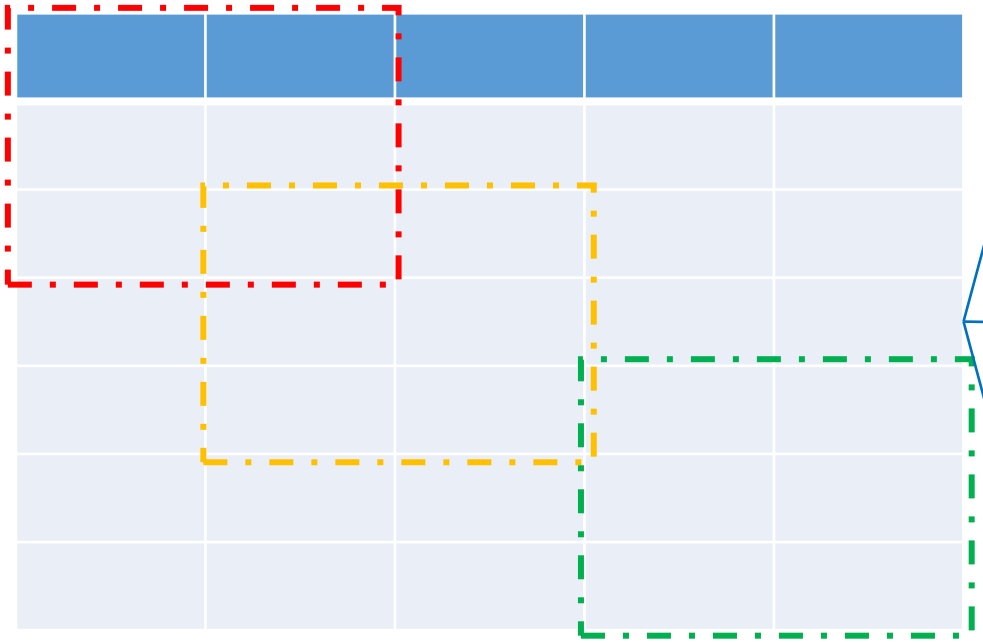
Who wins the next election?



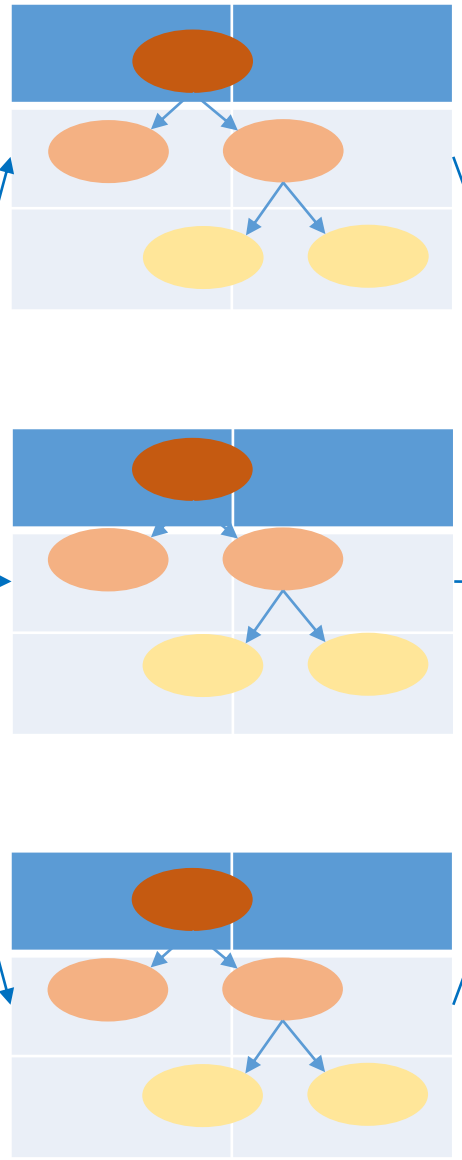
Random Forest

- Grow multiple decision trees (weak learners)
- Example of Bagging
- Technique of combining multiple classifiers modelled on different sub-samples of the same dataset
- Helps in reducing variance of predictions

Random Forest



Step 1 – Create multiple sub-samples of the dataset



Step 2 – Create multiple classifiers

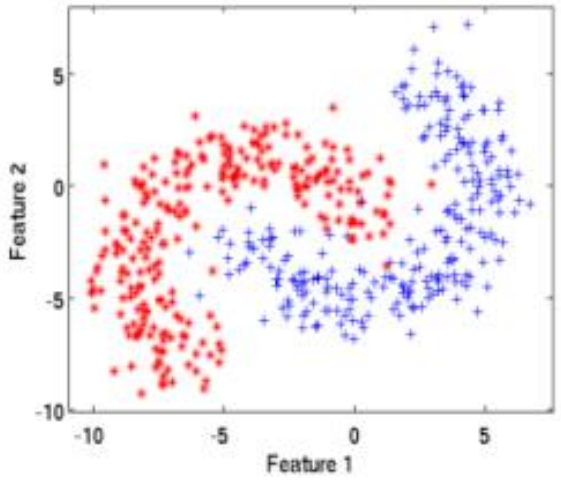
Step 3 – Combine classifiers

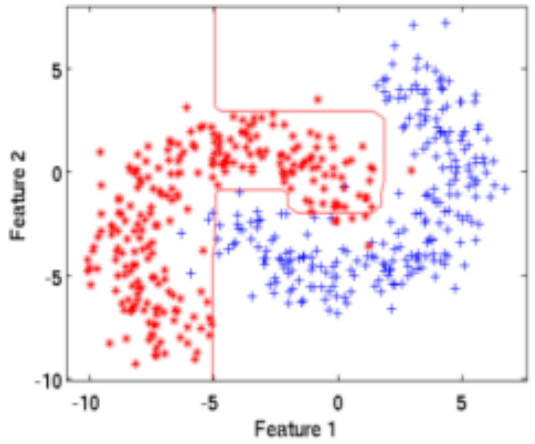
Step 4 – Predict based on voting

Random Forest - Algorithm

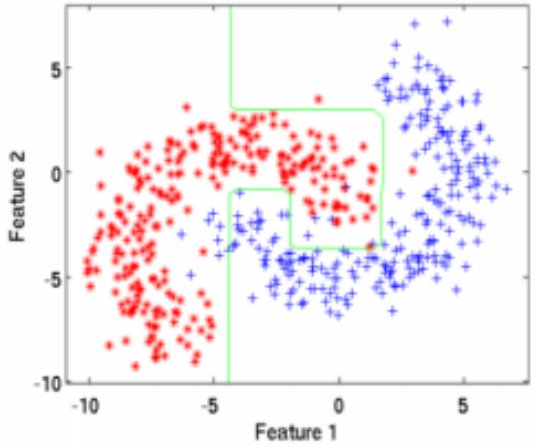
1. Assuming the dataset has size M x N (M rows, N columns). Select M rows at random with replacement
2. Select n < N columns at random. Select the best split
3. Grow the trees to the largest extent without any restriction/pruning
4. Predict by aggregating results of the trees (voting for classification, mean for regression)

Random Forest – Example 1

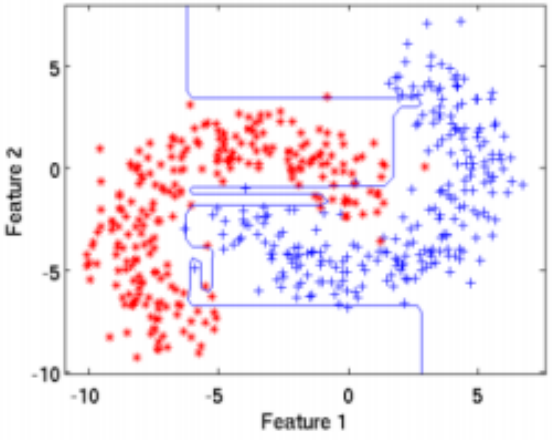




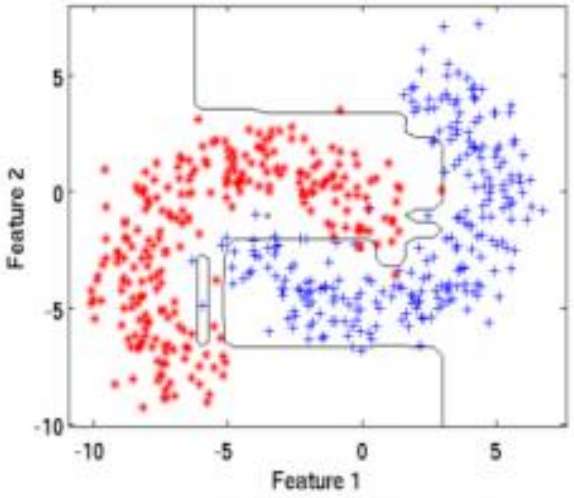
Decision boundary produced by one tree



Decision boundary produced by a second tree

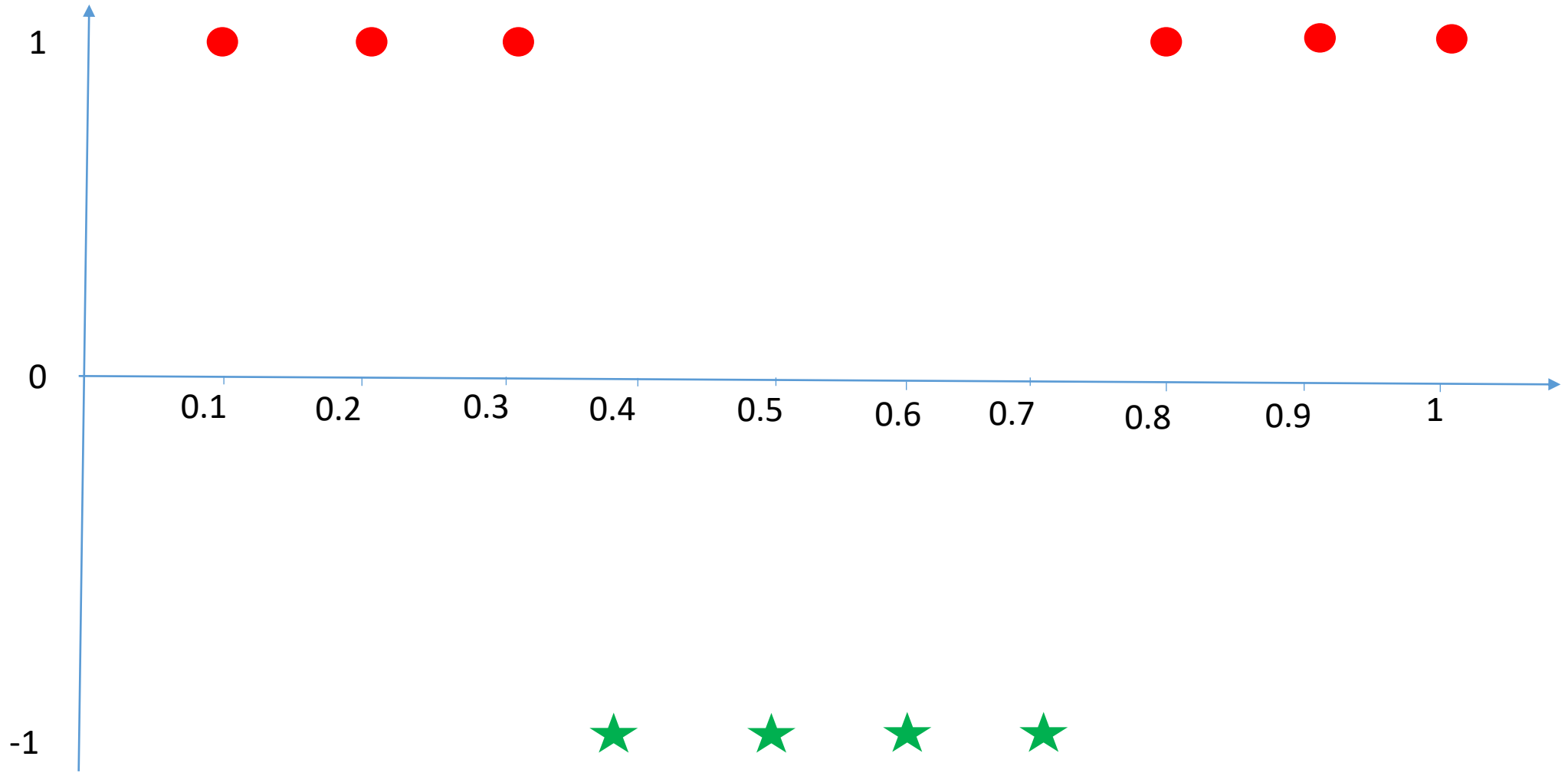


Decision boundary produced by a third tree



Final result from bagging all trees.

Random Forest – Example 2



Random Forest – Example 2

Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 2:

x	0.1	0.2	0.3	0.4	0.5	0.8	0.9	1	1	1
y	1	1	1	-1	-1	1	1	1	1	1

x <= 0.65 ==> y = 1
x > 0.65 ==> y = 1

Bagging Round 3:

x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 4:

x	0.1	0.1	0.2	0.4	0.4	0.5	0.5	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.3 ==> y = 1
x > 0.3 ==> y = -1

Bagging Round 5:

x	0.1	0.1	0.2	0.5	0.6	0.6	0.6	1	1	1
y	1	1	1	-1	-1	-1	-1	1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 6:

x	0.2	0.4	0.5	0.6	0.7	0.7	0.7	0.8	0.9	1
y	1	-1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 7:

x	0.1	0.4	0.4	0.6	0.7	0.8	0.9	0.9	0.9	1
y	1	-1	-1	-1	-1	1	1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 8:

x	0.1	0.2	0.5	0.5	0.5	0.7	0.7	0.8	0.9	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 9:

x	0.1	0.3	0.4	0.4	0.6	0.7	0.7	0.8	1	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 10:

x	0.1	0.1	0.1	0.1	0.3	0.3	0.8	0.8	0.9	0.9
y	1	1	1	1	1	1	1	1	1	1

x <= 0.05 ==> y = -1
x > 0.05 ==> y = 1

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	1	1	1	-1	-1	-1	-1	-1	-1	-1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	-1	-1	-1	-1	-1	-1	-1
4	1	1	1	-1	-1	-1	-1	-1	-1	-1
5	1	1	1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	1	1	1
7	-1	-1	-1	-1	-1	-1	-1	1	1	1
8	-1	-1	-1	-1	-1	-1	-1	1	1	1
9	-1	-1	-1	-1	-1	-1	-1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
Sum	2	2	2	-6	-6	-6	-6	2	2	2
Sign	1	1	1	-1	-1	-1	-1	1	1	1
True Class	1	1	1	-1	-1	-1	-1	1	1	1

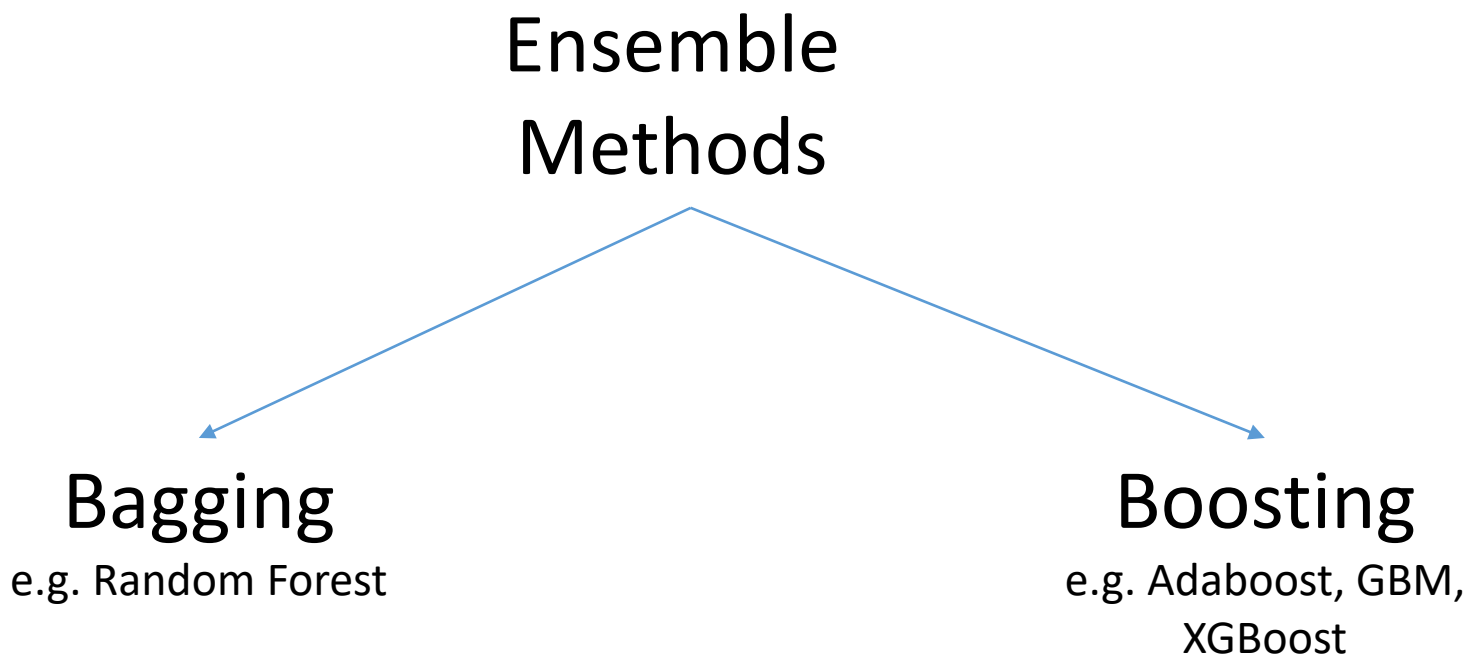
Random Forest– Advantages & Disadvantages

Advantages

- Useful for large, high dimensional data with high degree of non-linearity
- Useful for both classification & regression
- Provides relative importance of features

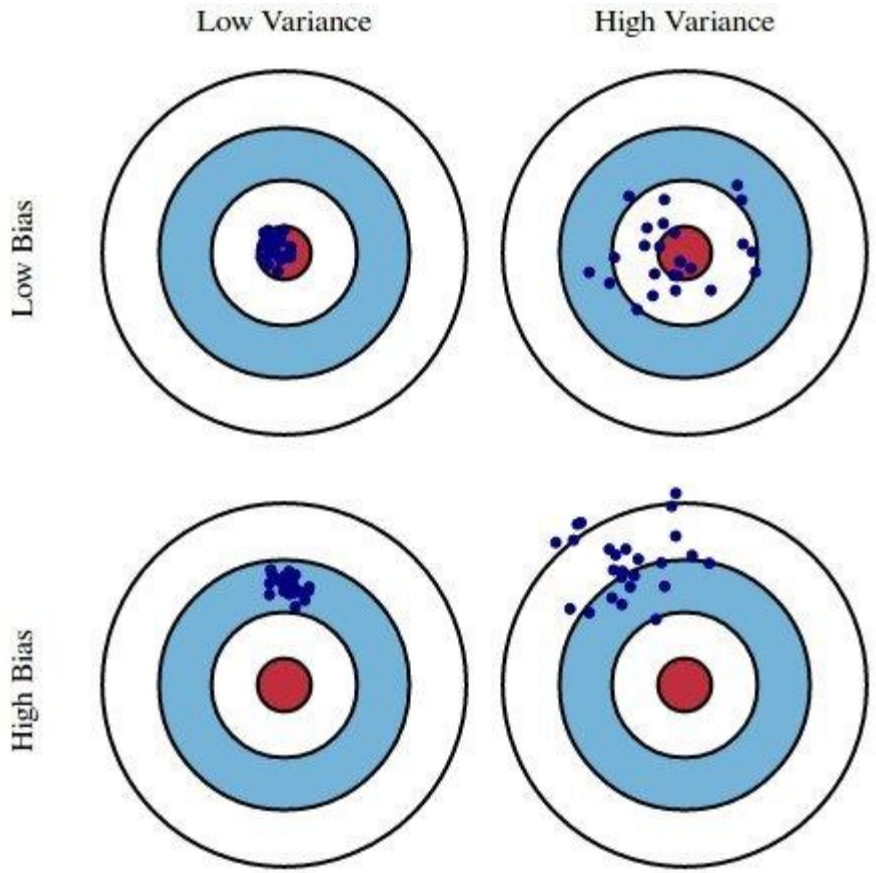
Disadvantages

- Black-box model
- Doesn’t yield great results for regression due to discontinuous predictions

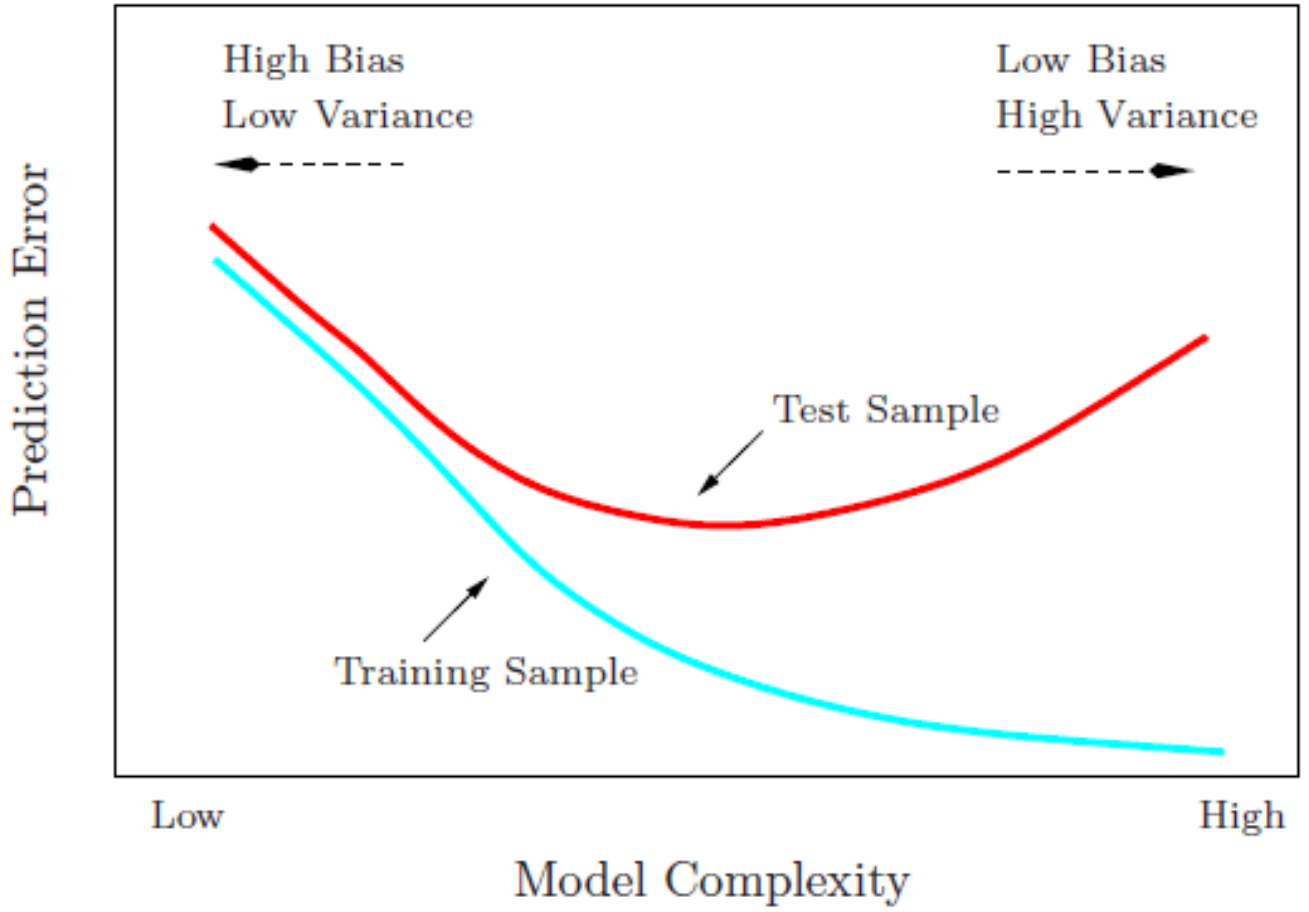

Ensemble Methods
Bagging
e.g. Random Forest
Boosting
e.g. Adaboost, GBM, XGBoost

Ensemble Methods

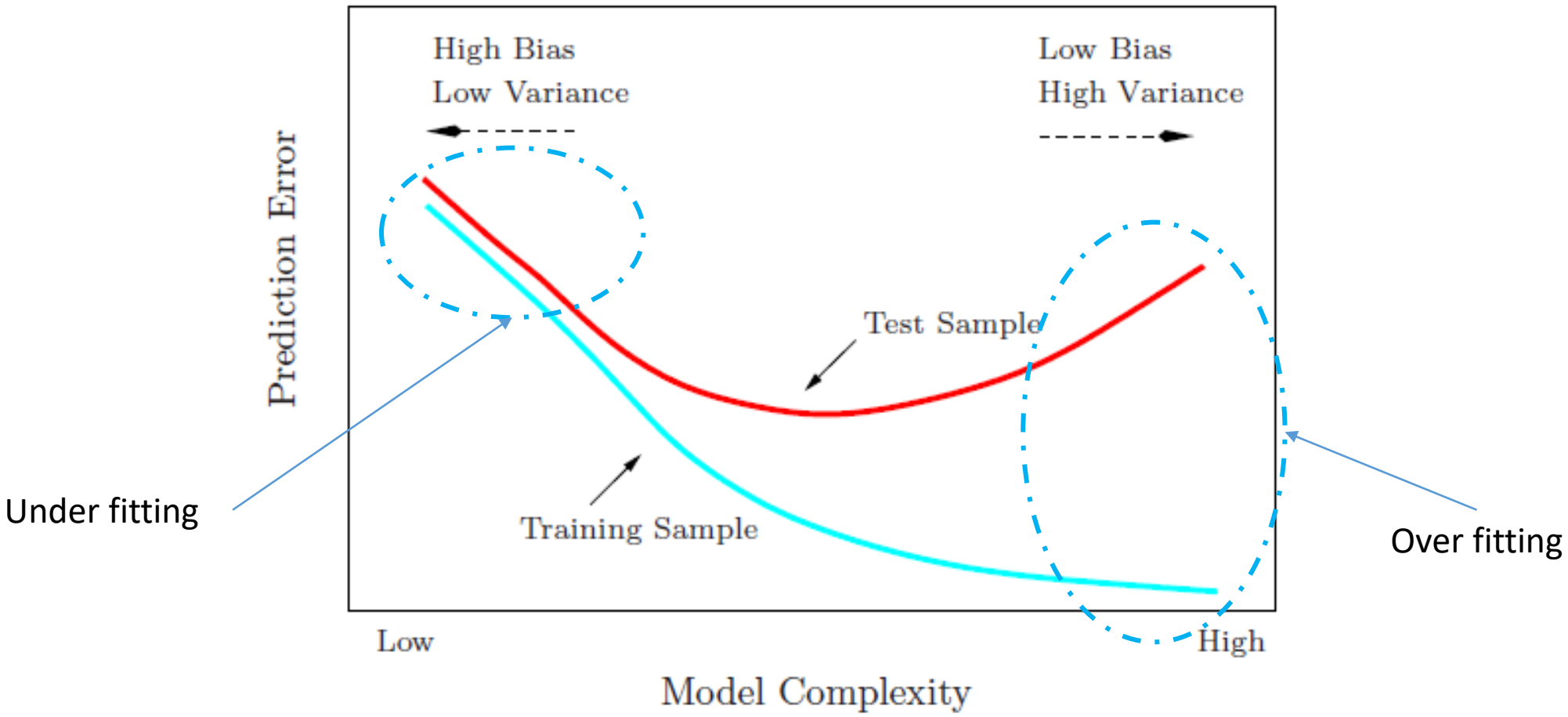
- Ensemble = Group
- Refers to employing a group of (weaker) predictive models to achieve better test accuracy (bias) and stability (variance)



Ensemble Methods



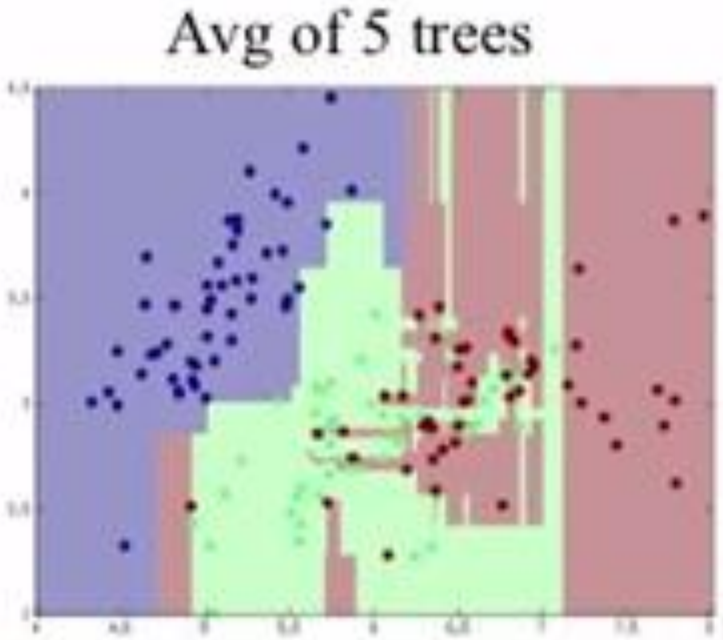
Ensemble Methods

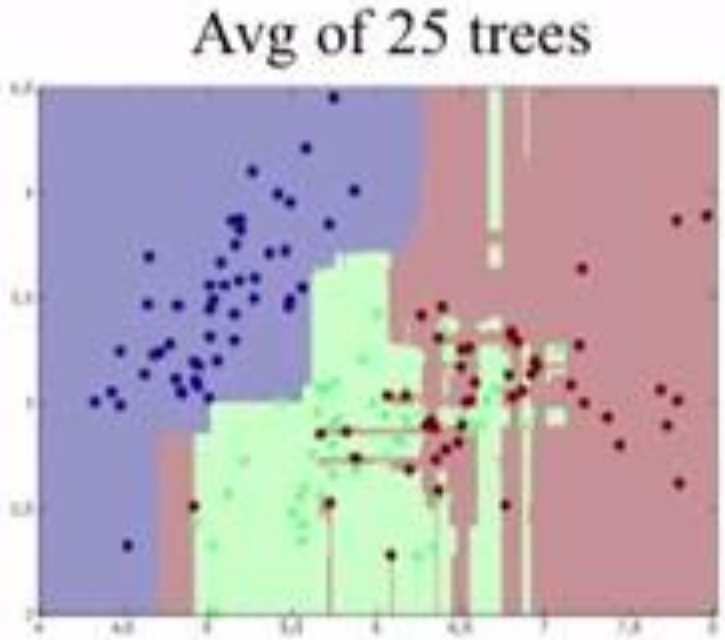


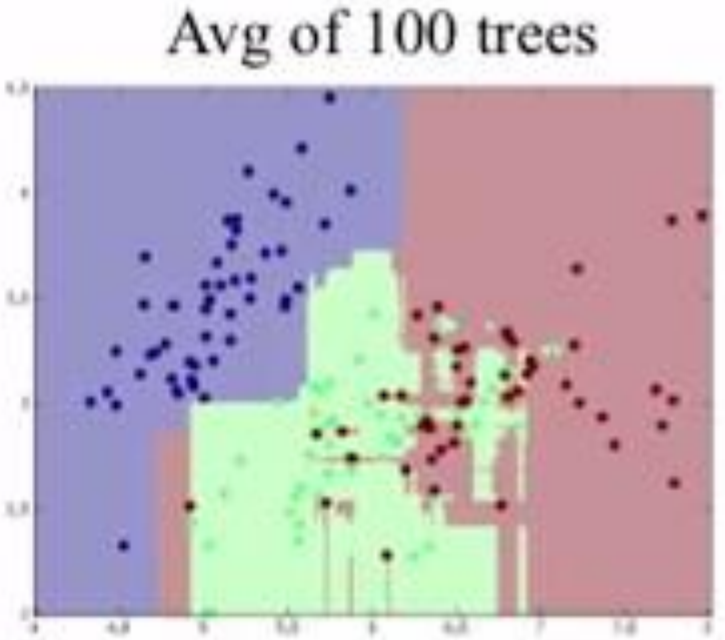
Bagging – Bootstrap Aggregating

- Operates via equal weighing of models
- Settles on result using majority votes
- Uses multiple instances of the same classifier for one dataset
- Builds models on smaller datasets using sampling with replacement
- Works best when classifier is unstable, i.e. has high variance

Bagging – Visual Example



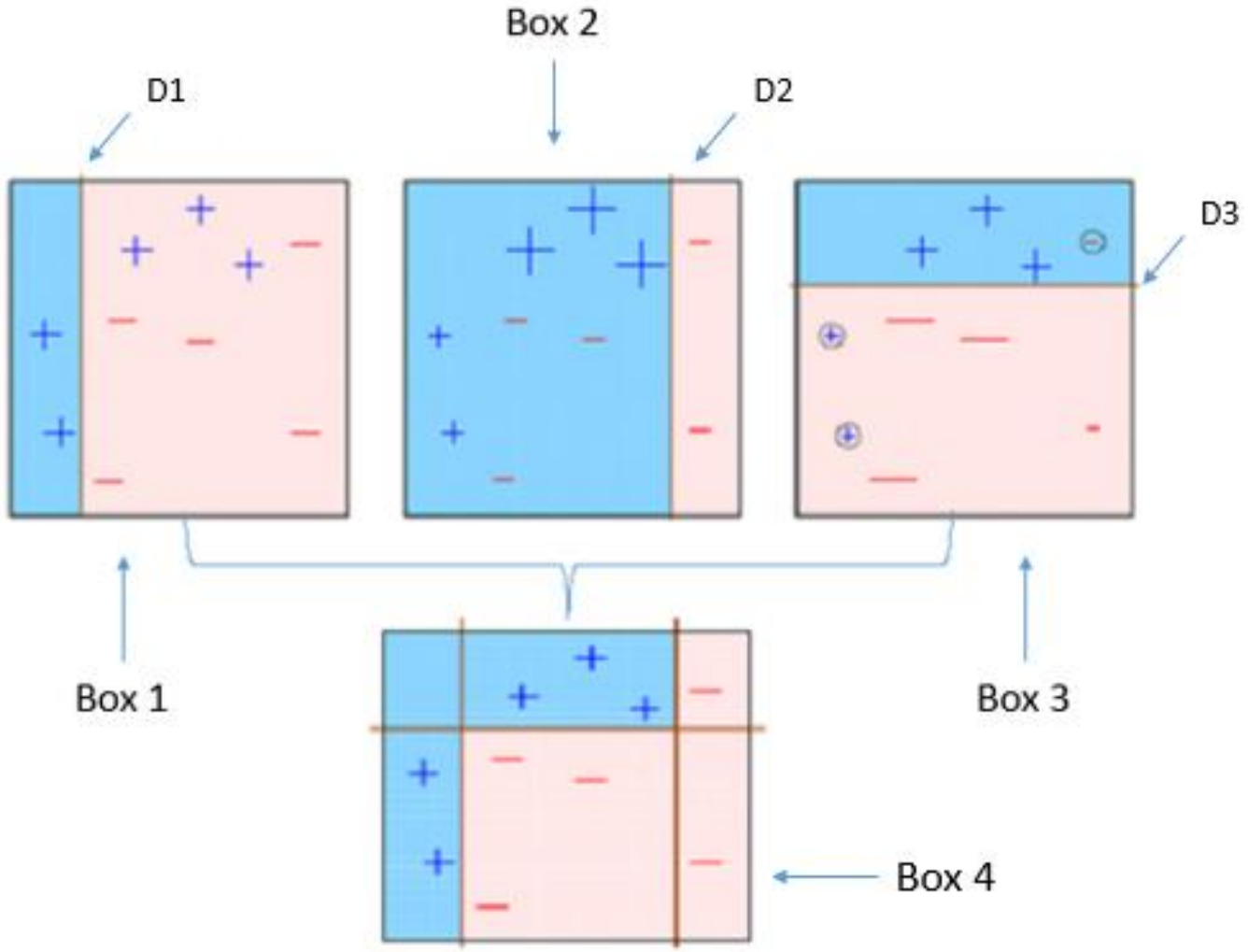




Boosting

- Similar to bagging with one conceptual difference
- Assigns varying weights to classifiers, instead of equal weights
- Algorithms proceeds iteratively, new models influenced by previous ones
- First a model is built on the training data. Then, a second model is created that attempts to correct errors from the first model
- New classifiers become experts for instances classified incorrectly by previous classifiers

Boosting – AdaBoost (Adaptive Boosting)



How Does AdaBoost Work?

Boosting – GBM (Gradient Boosted Machines)

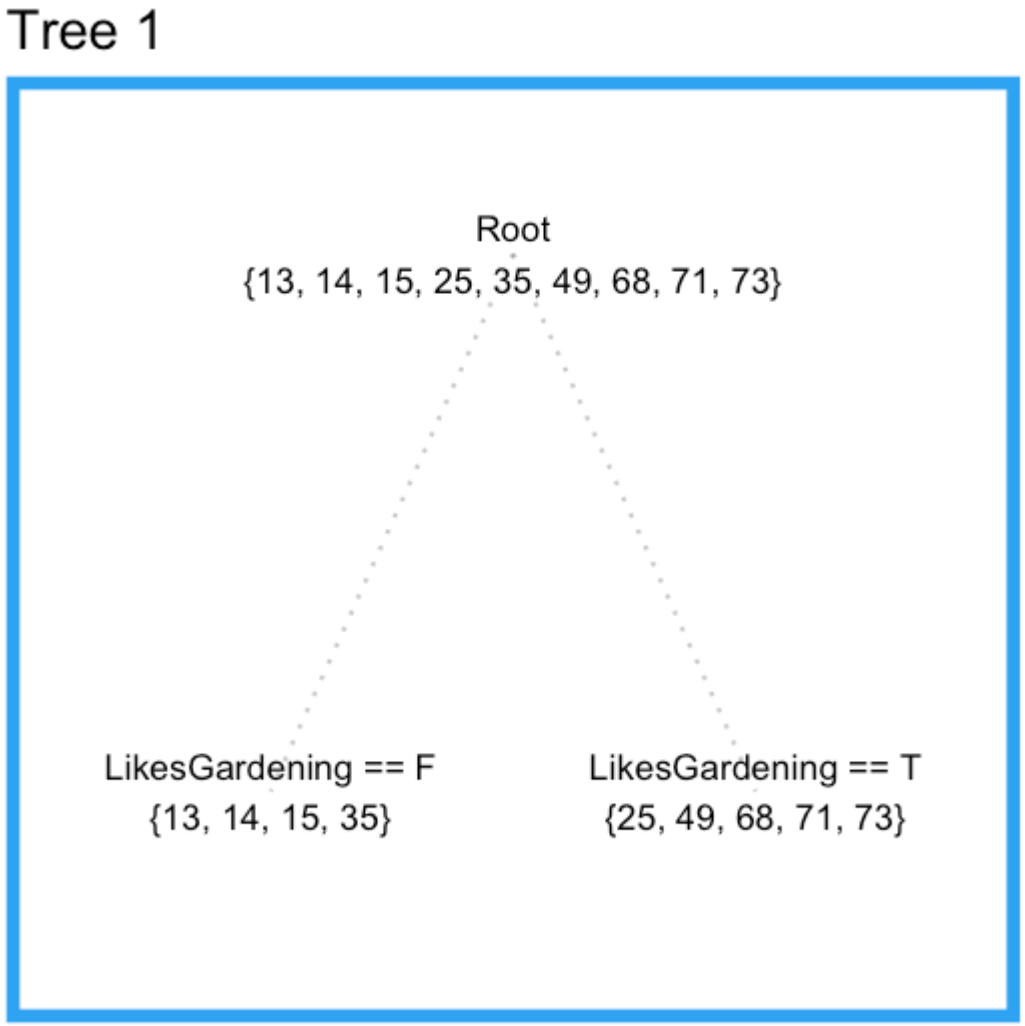
- Similar to AdaBoost, with a modification
- Instead of using varying weights, the focus of each new classifier is to gradually minimize the loss of the whole system using Gradient Descent method
- A weak learner is introduced to compensate the “shortcomings” of an existing weak learner
- “Shortcomings” are identified by gradients (instead of high-weight data points)
- So what is the Gradient Descent method?

How do GBM work?

PersonID	Age	LikesGardening	PlaysVideoGames
1	13	FALSE	TRUE
2	14	FALSE	TRUE
3	15	FALSE	TRUE
4	25	TRUE	TRUE
5	35	FALSE	TRUE
6	49	TRUE	FALSE
7	68	TRUE	TRUE
8	71	TRUE	FALSE
9	73	TRUE	FALSE

Lets suppose we are trying to predict a person's age based on whether they play video games and enjoy gardening.

How do GBM work?



Lets model the data with a regression tree.

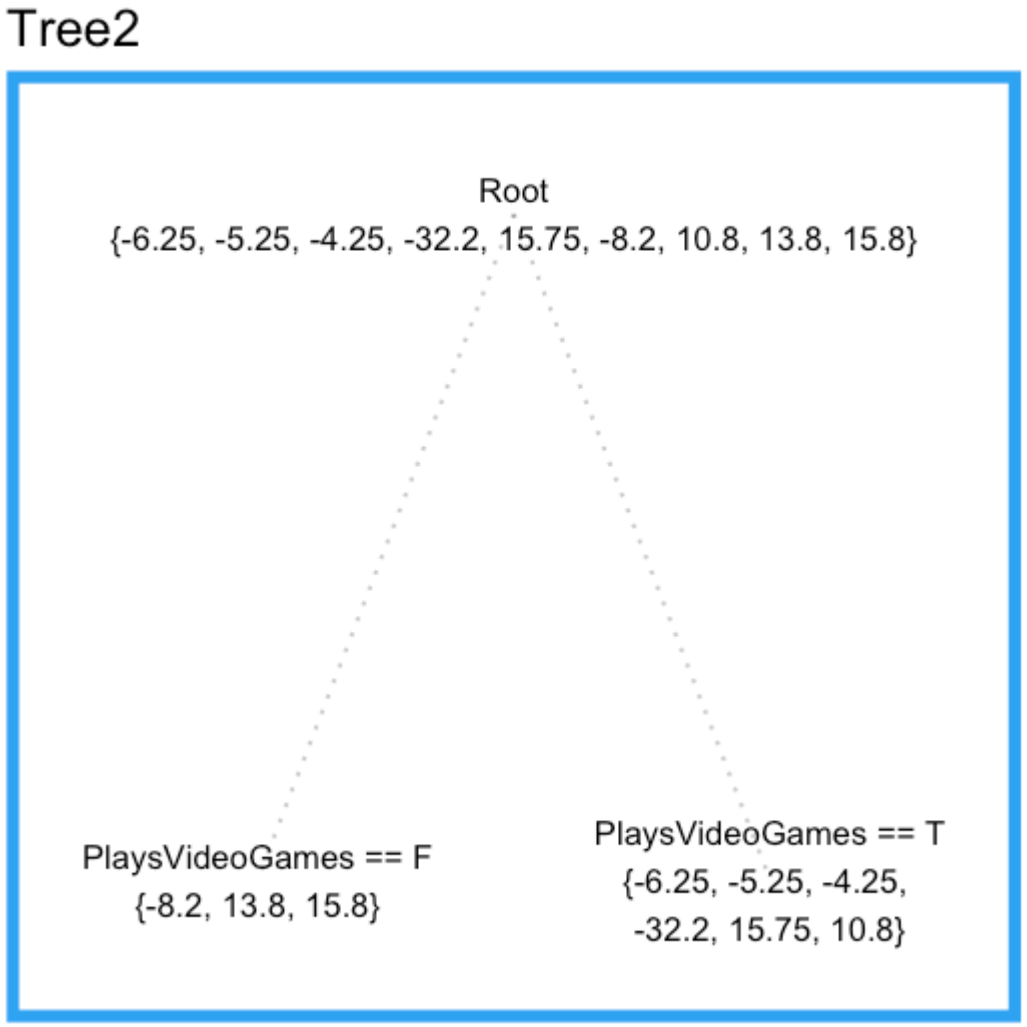
- Tree 1 prediction for node 1 = (13+14+15+35)/4 = 19.25 years
- Tree 1 prediction for node 2 = (25+49+68+71+73)/5 = 57.2 years

How do GBM work?

PersonID	Age	Tree1 Prediction	Tree1 Residual
1	13	19.25	-6.25
2	14	19.25	-5.25
3	15	19.25	-4.25
4	25	57.2	-32.2
5	35	19.25	15.75
6	49	57.2	-8.2
7	68	57.2	10.8
8	71	57.2	13.8
9	73	57.2	15.8

- Lets check the training errors from our first tree.
- Clearly, this tree has large errors (residuals). We can build a better predictor by minimizing these residuals.
- Lets fit a second regression tree to the residuals of the first tree

How do GBM work?



- Tree 2 prediction for node 1 = (-8.2+13.8+15.8)/3 = 7.133
- Tree 2 prediction for node 2 = (-6.25-5.25-4.25-32.2+15.75+10.8)/6 = -3.567

How do GBM work?

PersonID	Age	Tree1 Prediction	Tree1 Residual	Tree2 Prediction	Combined Prediction	Final Residual
1	13	19.25	-6.25	-3.567	15.68	2.683
2	14	19.25	-5.25	-3.567	15.68	1.683
3	15	19.25	-4.25	-3.567	15.68	0.6833
4	25	57.2	-32.2	-3.567	53.63	28.63
5	35	19.25	15.75	-3.567	15.68	-19.32
6	49	57.2	-8.2	7.133	64.33	15.33
7	68	57.2	10.8	-3.567	53.63	-14.37
8	71	57.2	13.8	7.133	64.33	-6.667
9	73	57.2	15.8	7.133	64.33	-8.667

Clearly, the predictions from our first tree can be improved by adding the “error-correcting” predictions from the second tree.
Lets now try to generalize this…

Boosting - XGBoost

- The most popular boosting technique
- Similar to GBM, but also includes the regularization
- Also leverages hardware to improve compute time and memory usage through parallel processing
- Allows users to define custom optimization functions
- Uses pruning instead of restriction of tree size
- Has built-in cross-validation
- http://xgboost.readthedocs.io/en/latest/model.html

Cross Validation