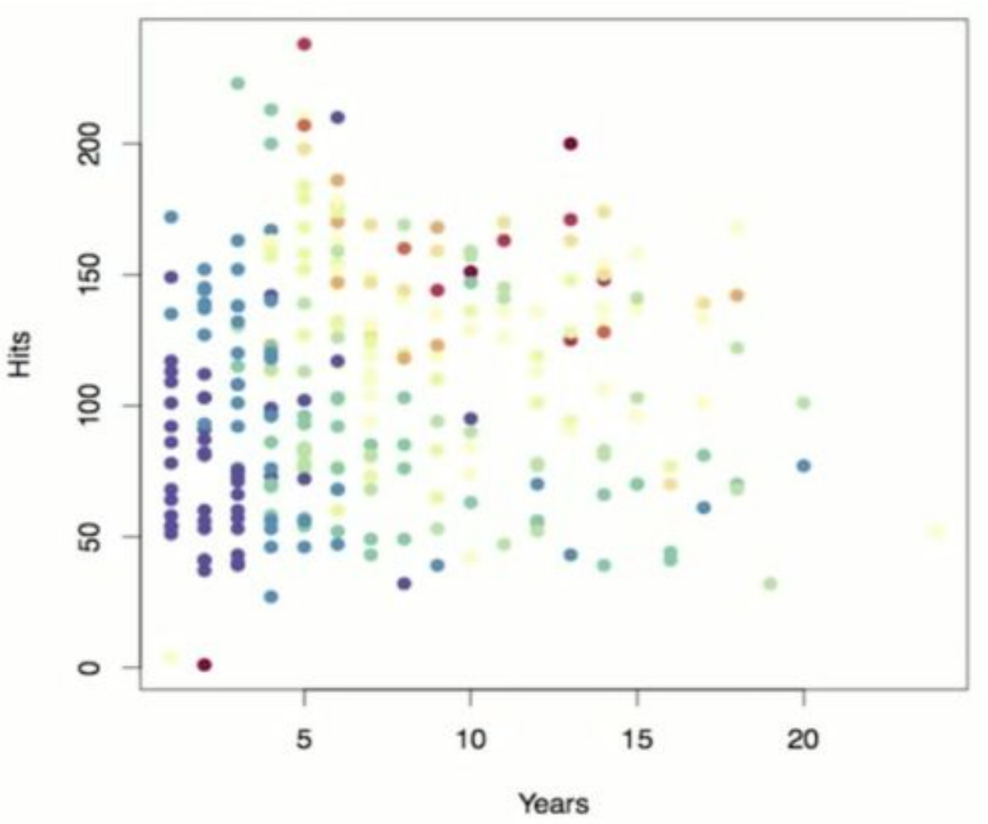
Decision trees and their Ensembles

RECAP: decision trees

- Stratifies or segments the predictor space into a number of simple regions.
- Uses *if-then-else* rules that provide branching for classification.
- Prediction for a given observation is typically made by using the **mean** or the **mode** of the training observations in the region to which it belongs (CART -- classification and regression trees).

RECAP: decision trees



Some dataset…

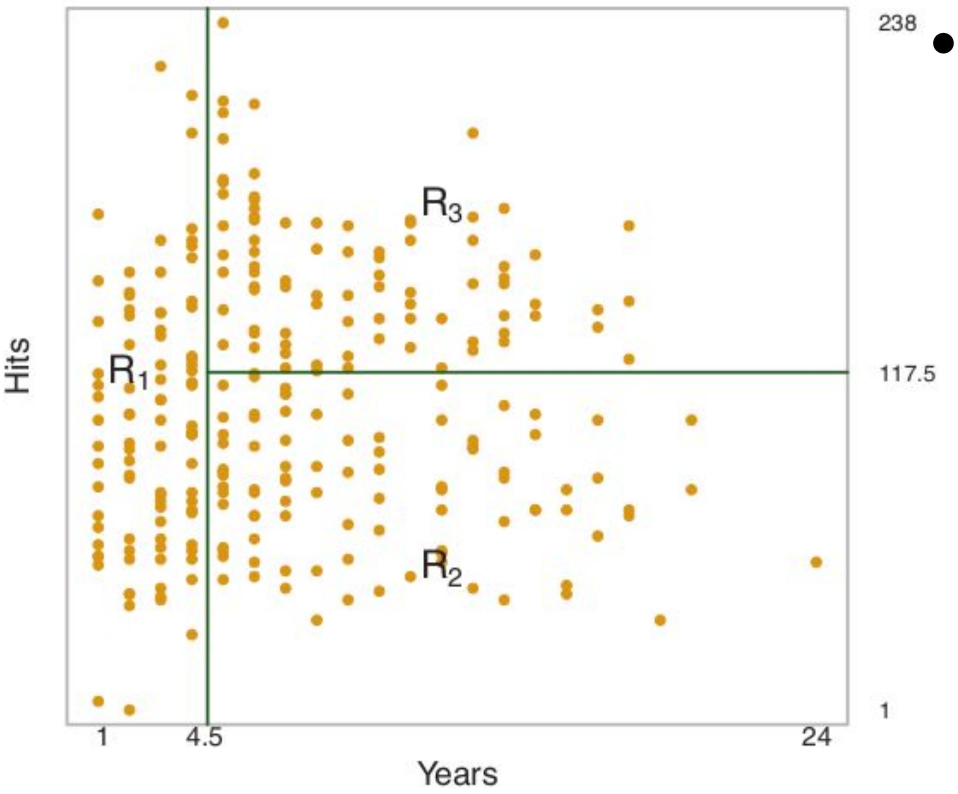
Y :Points -- baseball player's Salary

X_1 :Years -- the number of years that he has played in the major leagues

X_2 :Hits -- the number of hits that he made in the previous year

GOAL: We want to do regression of salary using decision trees

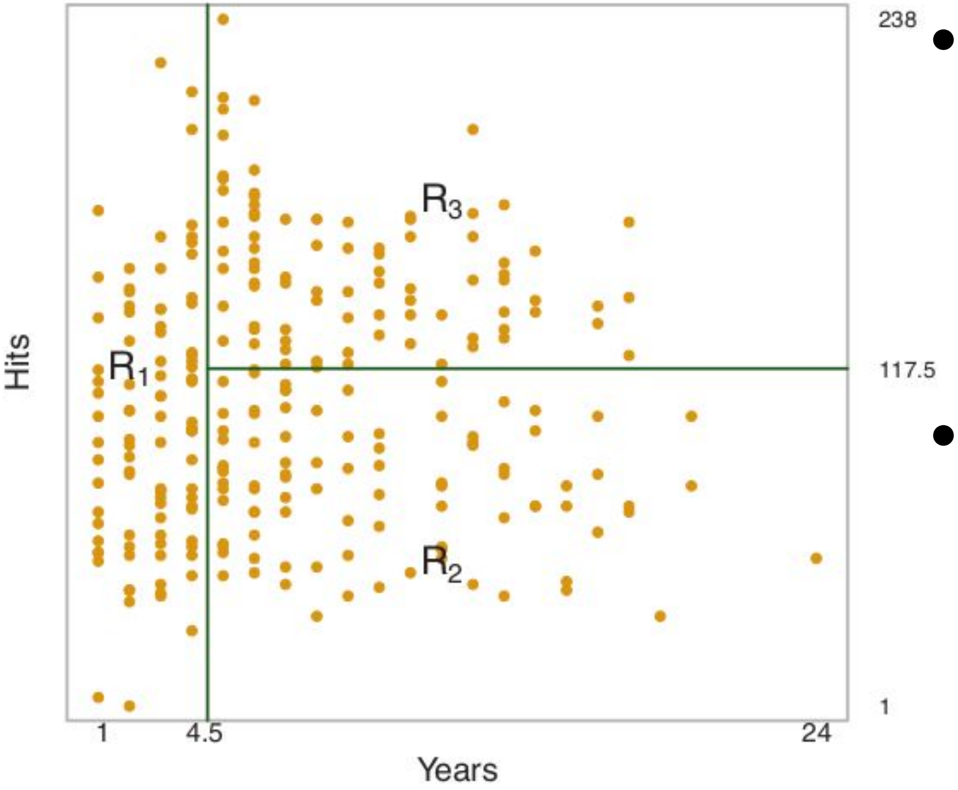
RECAP: decision trees



$x_1, x_2, x_3, \dots\ x_n$

- Segment the predictor space into a number of simple regions (R_1, R_2, …, R_m).
  R_1 ={X | Years<4.5}
  R_2 ={X | Years>=4.5, Hits<117.5}
  R_3 ={X | Years>=4.5 , Hits>=117.5}

RECAP: decision trees



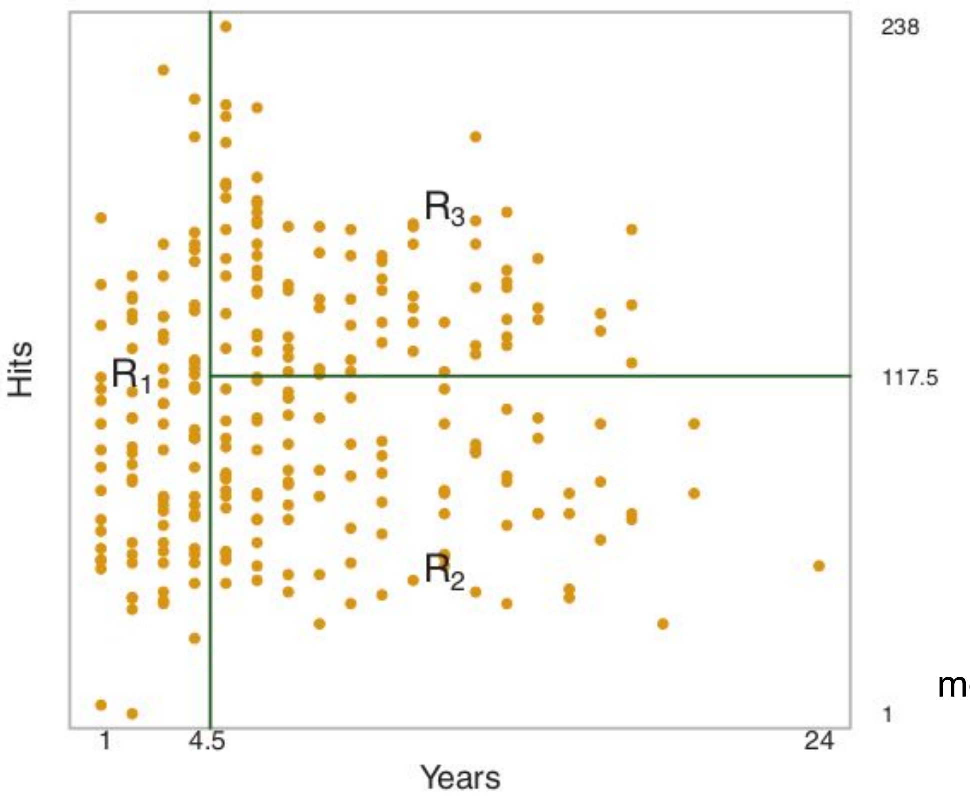
$x_1, x_2, x_3, \ldots\ x_p$

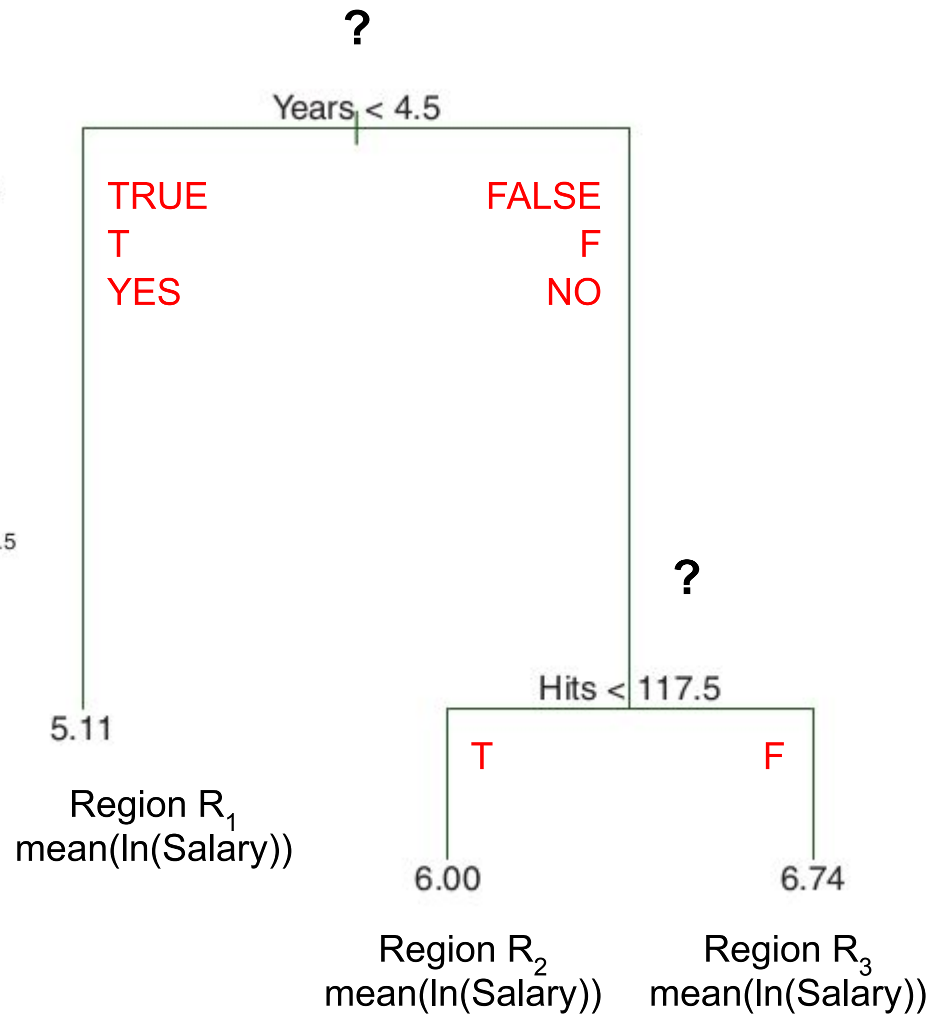
- Segment the predictor space into a number of simple regions ($R_1, R_2, \ldots, R_j$).
  R_1 ={X | Years<4.5}
  R_2 ={X | Years>=4.5, Hits<117.5}
  R_3 ={X | Years>=4.5 , Hits>=117.5}

- Use *if-then-else* rules that provide branching for classification:

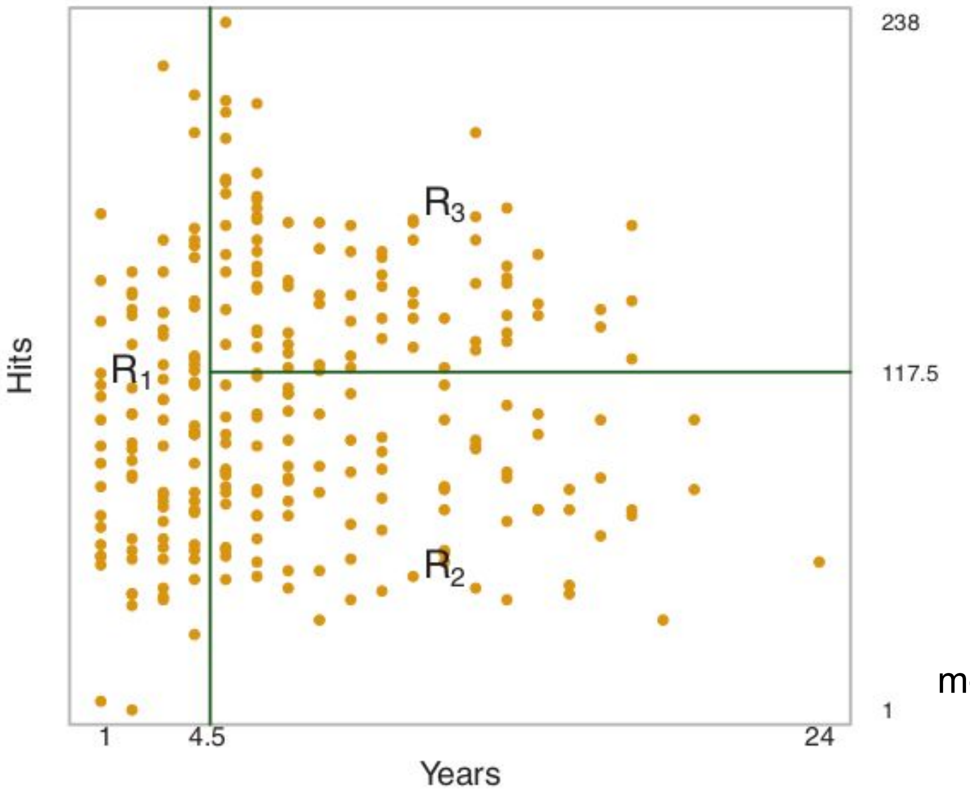
  If (Years < 4.5), then (R_1),
  else (R_2 or R_3)
  If (Hits < 117.5), then (R_2),
  else (R_3)

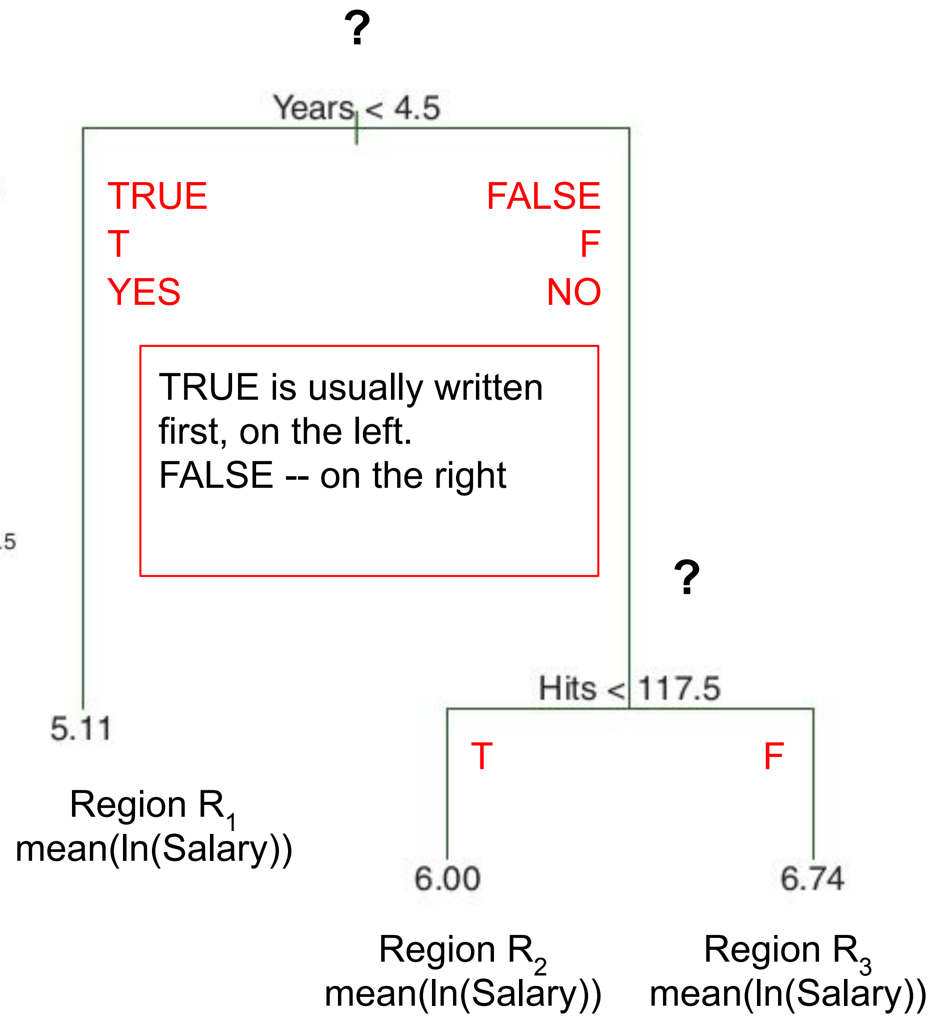
RECAP: decision trees





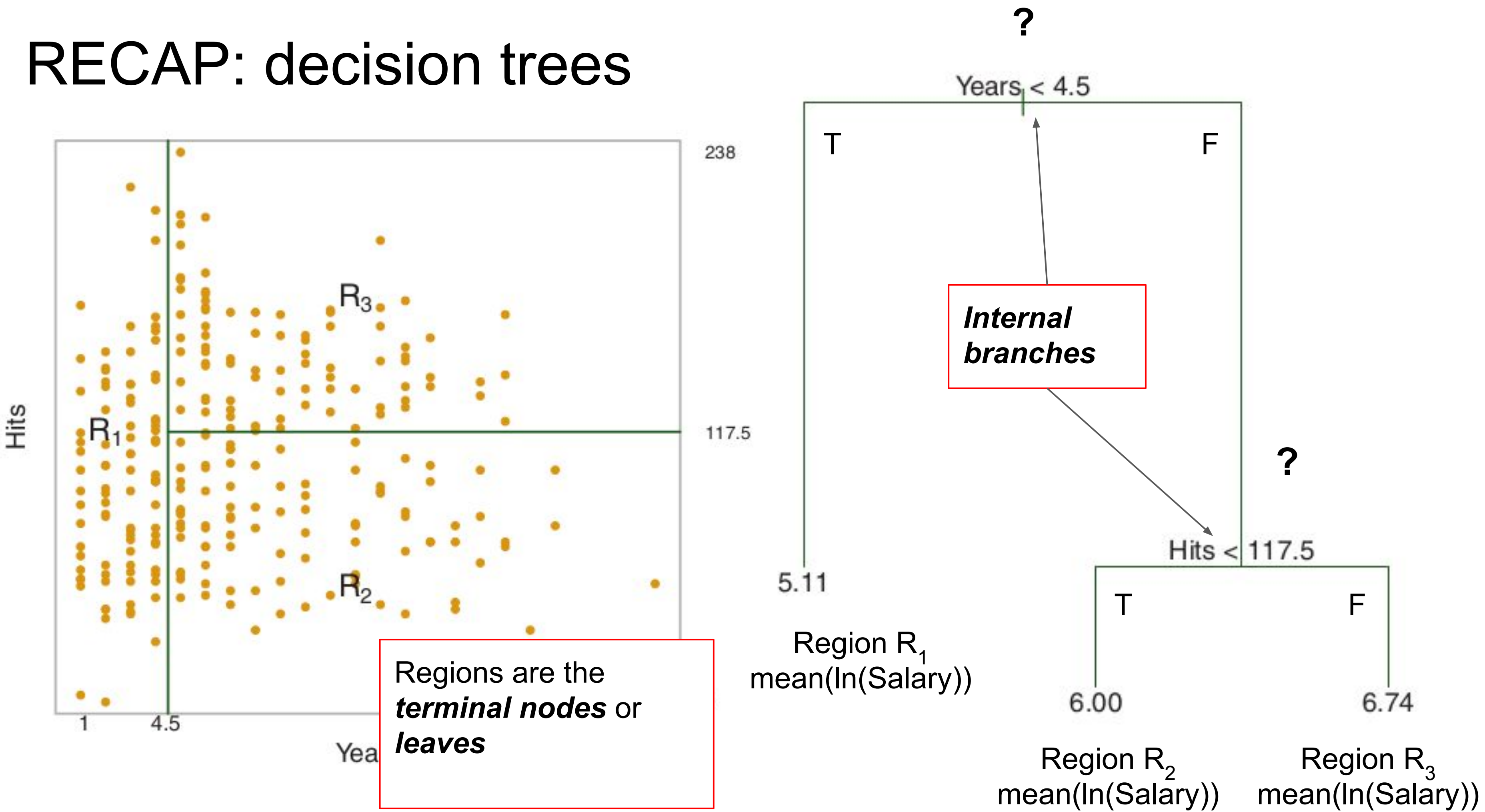
RECAP: decision trees



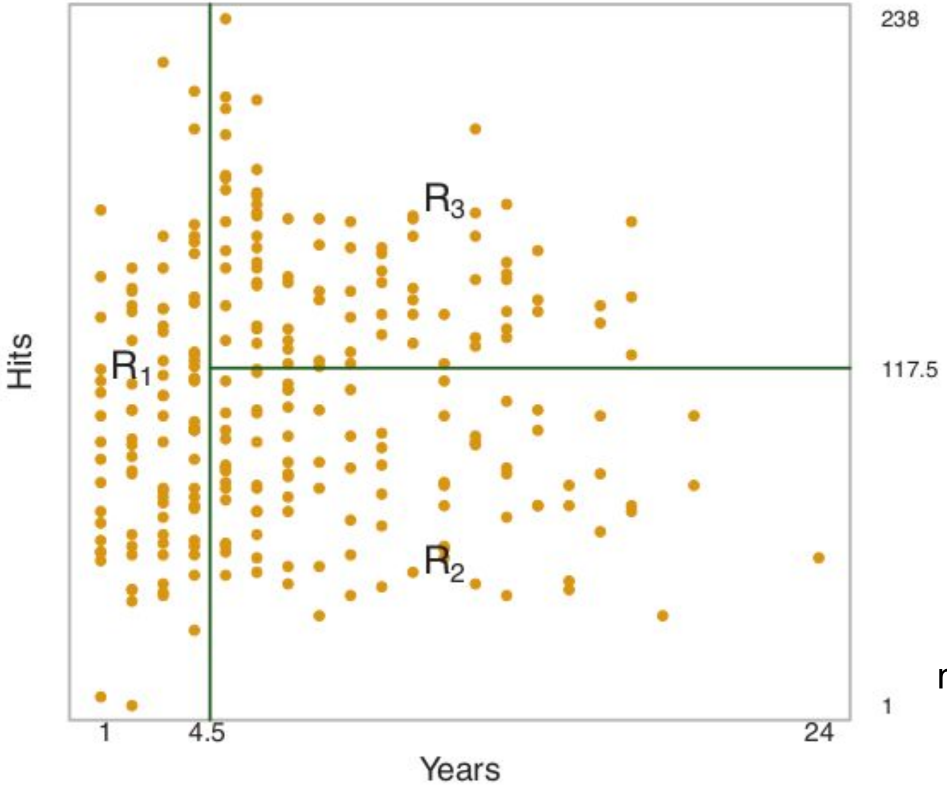


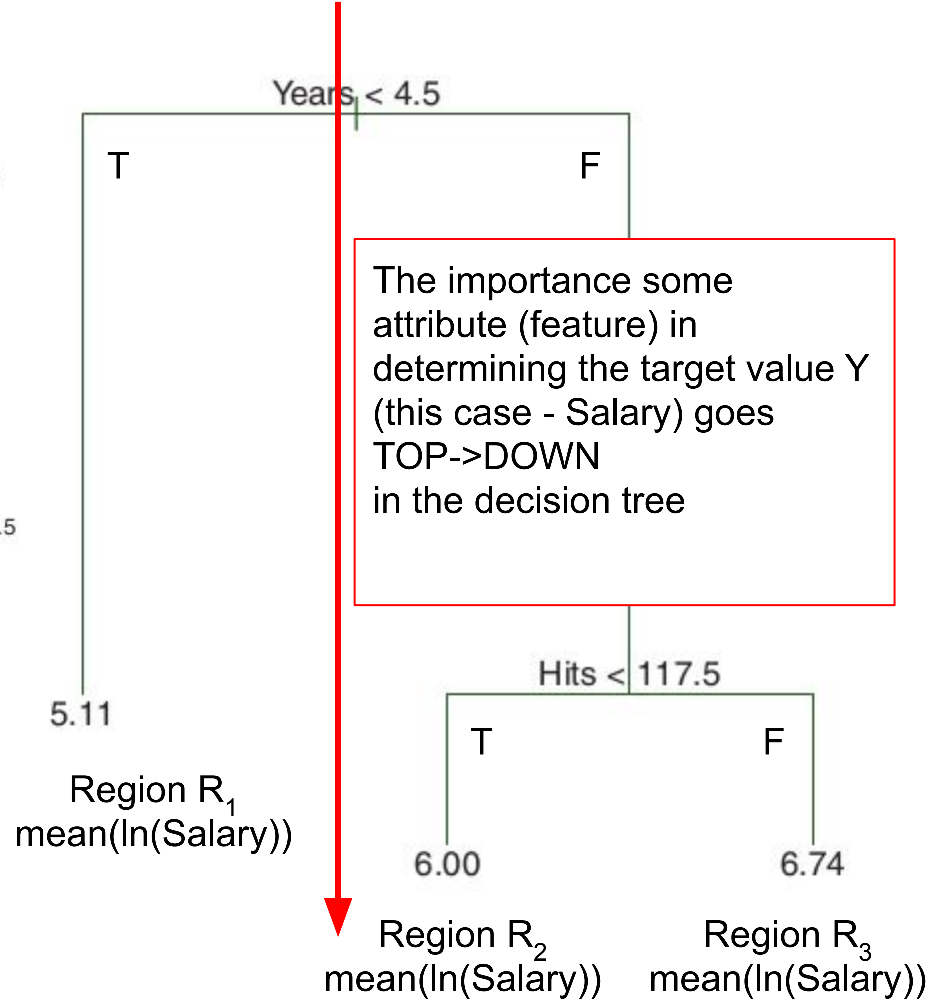
TRUE is usually written first, on the left.
FALSE -- on the right

RECAP: decision trees



RECAP: decision trees



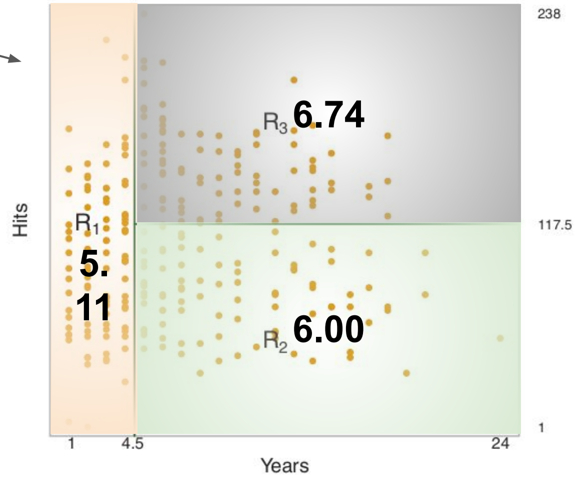


The importance some attribute (feature) in determining the target value Y (this case - Salary) goes TOP->DOWN in the decision tree

RECAP: decision trees

Decision tree creation is basically 2 steps:

1. Divide the predictor space—that is, the set of possible values for $X_1, X_2, ..., X_p$ - into J distinct and non-overlapping regions, $R_1, R_2, ..., R_J$.
2. For every observation that falls into the region R_j , we make the same prediction, which is simply the mean (or mode/majority) of the response values for the training observations in R_j



RECAP: decision trees

Decision tree creation is basically 2 steps:

1. Divide the predictor space—that is, the set of possible values for X_1, X_2, ..., X_p - into J distinct and non-overlapping regions, R_1, R_2, ..., R_J.
2. For every observation that falls into the region R_j , we make the same prediction, which is simply the mean (or mode/majority) of the response values for the training observations in R_j

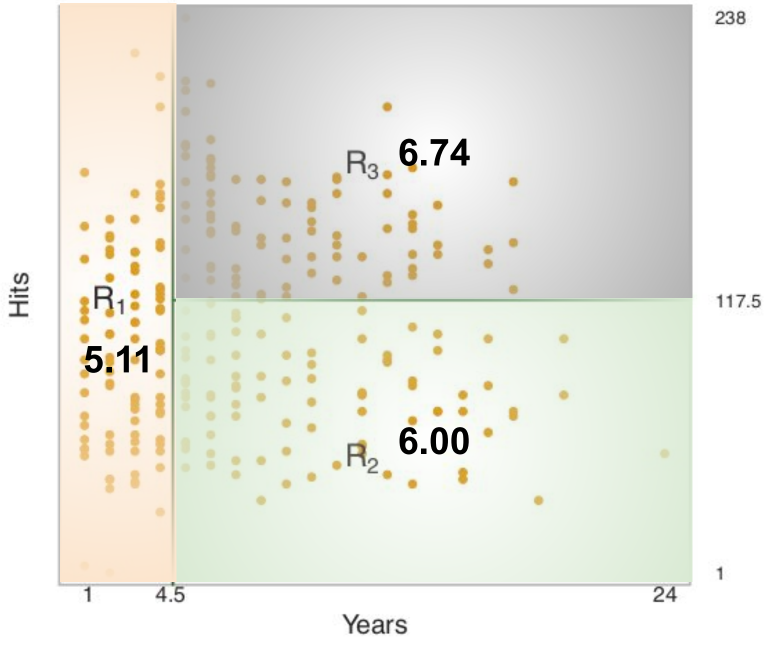
Q: How do we construct the regions R_1,...,R_J?

RECAP: decision trees

1. Divide the predictor space—that is, the set of possible values for X_1, X_2, ..., X_p - into J distinct and non-overlapping regions, R_1, R_2, ..., R_J.

Q: How do we construct the regions R_1,...,R_J?

RSS = $$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$



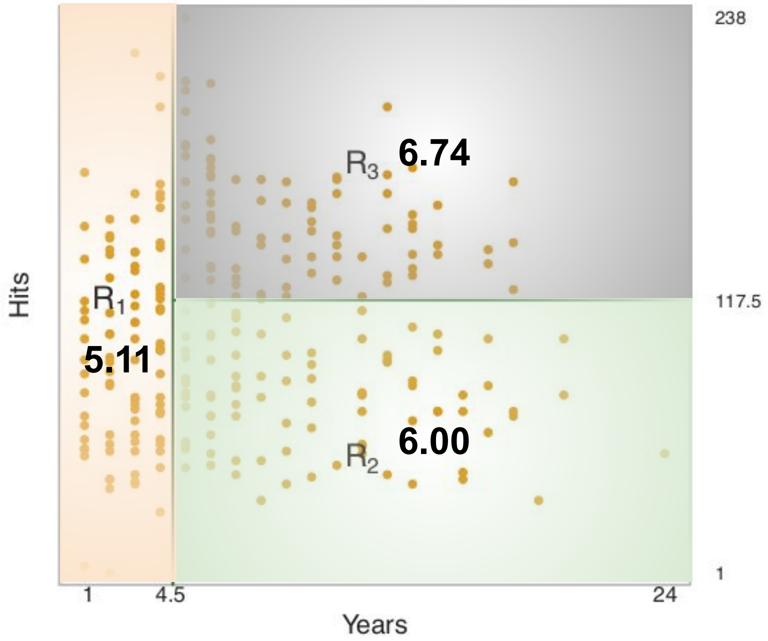
RECAP: decision trees

1. Divide the predictor space—that is, the set of possible values for X_1, X_2, ..., X_p - into J distinct and non-overlapping regions, R_1, R_2, ..., R_J.

Q: How do we construct the regions R_1,...,R_J?

RSS = $$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

Minimize it



RECAP: decision trees

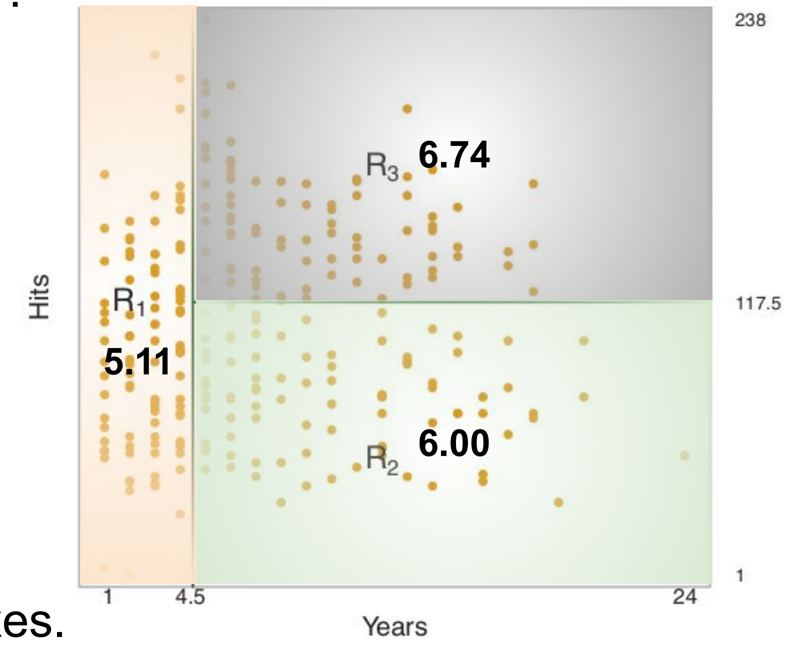
1. Divide the predictor space—that is, the set of possible values for $X_1, X_2, ..., X_p$ - into J distinct and non-overlapping regions, $R_1, R_2, ..., R_J$.

Q: How do we construct the regions $R_1,...,R_J$?

$$\text{RSS} = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

Minimize it

🙁 computationally infeasible to consider every possible partition of the feature space into J boxes.



RECAP: decision trees

1. Divide the predictor space—that is, the set of possible values for X_1, X_2, ..., X_p - into J distinct and non-overlapping regions, R_1, R_2, ..., R_J.

Q: How do we construct the regions R_1,...,R_J?

recursive binary splitting --

top-down, (begins at the top of the tree (at which point all observations belong to a single region)

greedy (at each step of the tree-building process, the best split is made at that particular step, not looking what split could lead to better split in some future step)

approach to find those R_j regions

RECAP: decision trees

recursive binary splitting --

top-down,

greedy

approach to find those R_j regions

We split a complicated problem to a single more simple problems that we can solve at the time

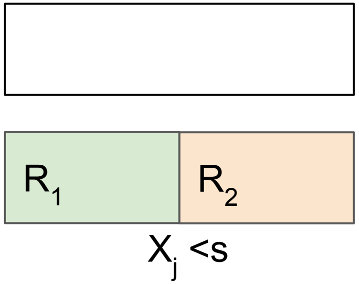
We need:

Some predictor *j*

Cutpoint value *s*

$$\sum_{i:\ x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\ x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

Easier to minimize



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

RECAP: decision trees

recursive binary splitting --

top-down,

greedy

approach to find those R_j regions

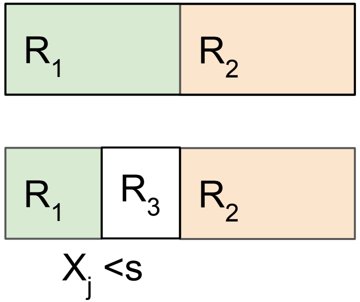
We split a complicated problem to a single more simple problems that we can solve at the time

We need:

Some new predictor *j*

New cutpoint value *s*

$$\sum_{i:\ x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \in R_3(j,s)} (y_i - \hat{y}_i)^2$$



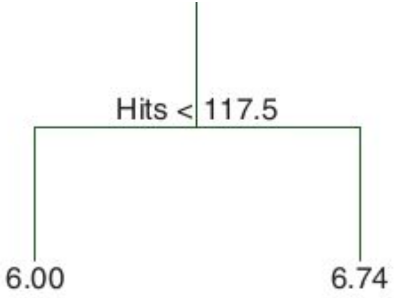
Easier to minimize

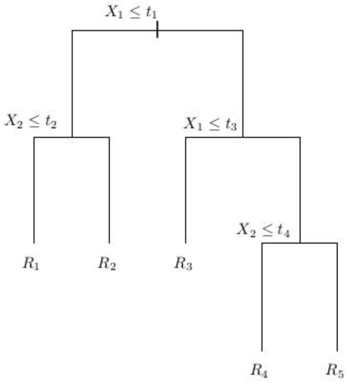
$$R_1(j,s) = \{X | X_j < s\} \text{ and } R_3(j,s) = \{X | X_j \geq s\}$$

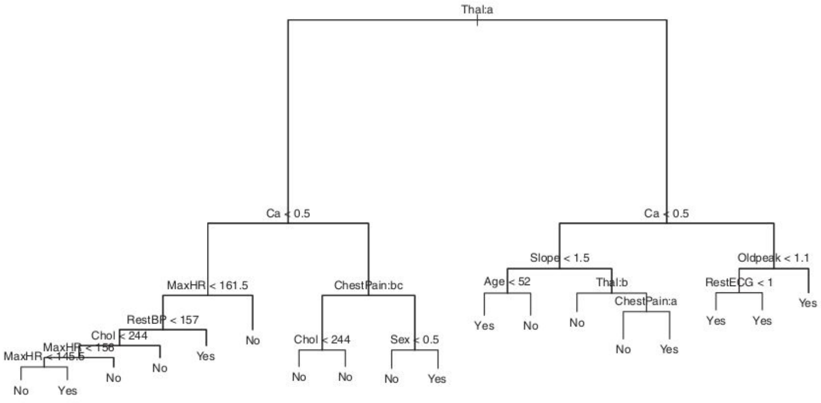
Stopping criterion e.g.:

- # of observations in R_j
- decrease in the RSS due to each split < threshold

RECAP: decision trees



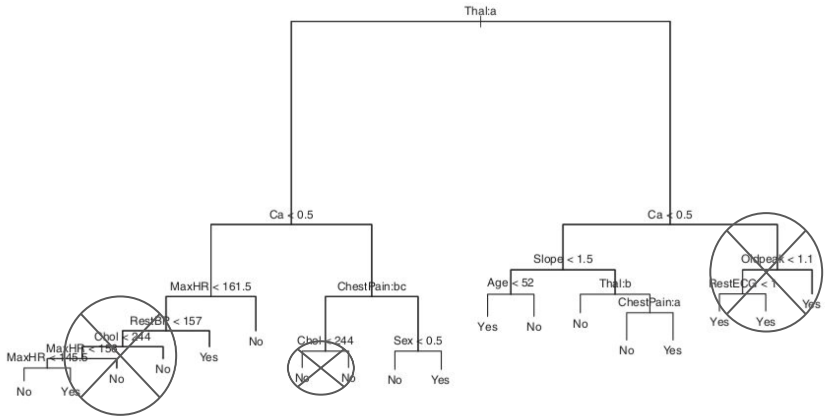




Simple
Rigid
Could underfit

Complex
Flexible
Could overfit

Decision tree pruning



How?

Intuitive solution:
select a subtree that leads to the lowest test error rate (using CV)

Cost complexity pruning
(a.k.a **weakest link pruning**)

Decision tree pruning

Cost complexity pruning
(a.k.a **weakest link pruning**)

the number of terminal nodes of the tree T

nonnegative tuning parameter α

$$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

mth terminal node

mean of the training observations in R_m

- When α = 0, then the subtree T will simply equal full tree T_0
- As α increases, there is a price to pay for having a tree with many terminal nodes

Decision tree pruning

Cost complexity pruning
(a.k.a **weakest link pruning**)

the number of terminal nodes of the tree T

nonnegative tuning parameter α

$$\underbrace{\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2}_{\text{RSS}} + \underbrace{\alpha |T|}_{\text{Penalty}}$$

mth terminal node

mean of the training observations in R_m

- When α = 0, then the subtree T will simply equal full tree T_0
- As α increases, there is a price to pay for having a tree with many terminal nodes

Decision tree pruning

Cost complexity pruning
(a.k.a **weakest link pruning**)

the number of terminal nodes of the tree T

nonnegative tuning parameter α

$$\underbrace{\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2}_{\text{RSS}} + \underbrace{\alpha|T|}_{\text{Penalty}}$$

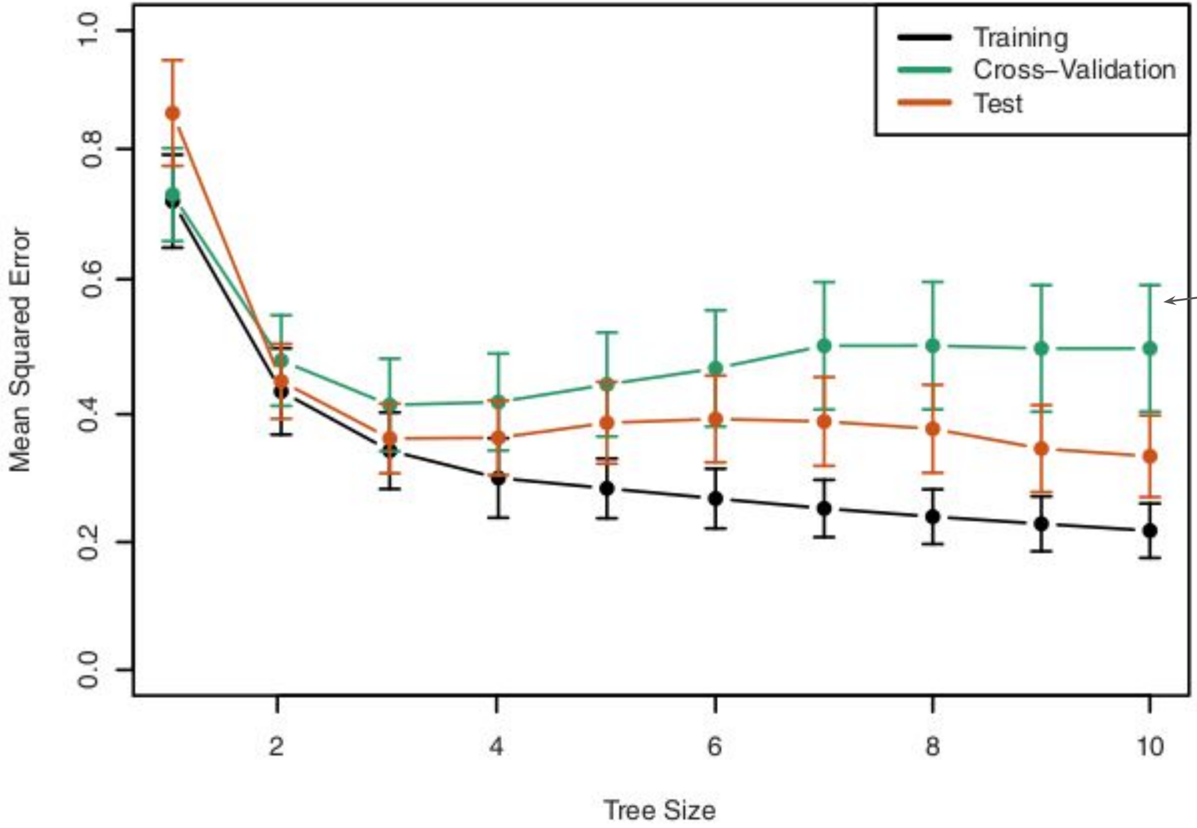
mth terminal node

mean of the training observations in R_m

- When α = 0, then the subtree T will simply equal full tree T_0
- As α increases, there is a price to pay for having a tree with many terminal nodes
- branches get pruned from the tree in a nested and predictable fashion

We can use Cross Validation to find best α value

Decision tree pruning



Standart error bars

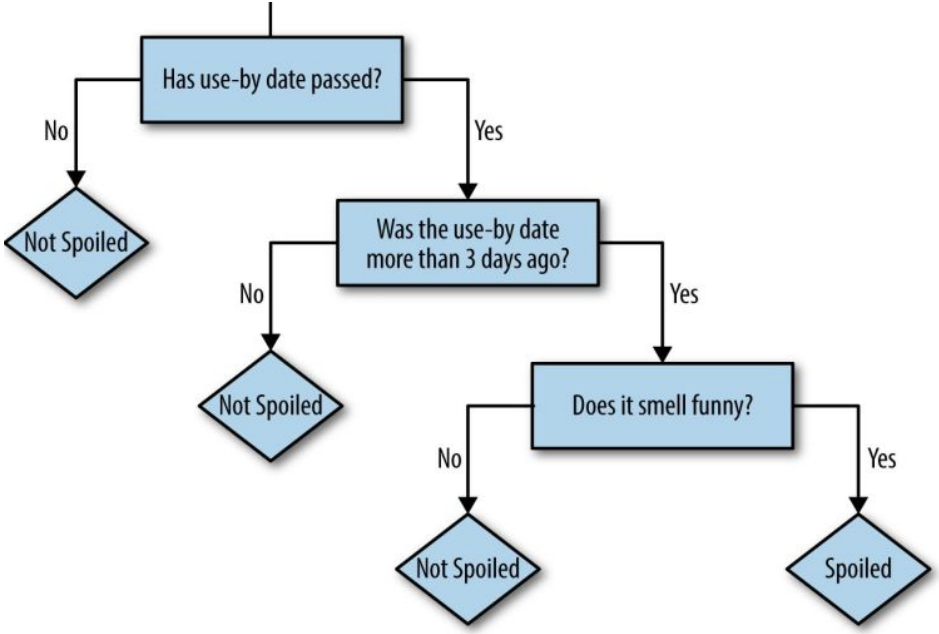
Classification Trees

similar to regression tree:

- Recursive binary splitting to grow a classification tree
- Tree pruning is performed

differences:

- **Y**: qualitative *instead of* quantitative
- **predicted response**: most commonly occurring class of training observations *instead of* mean response of training obs.
- **Criteria to minimise**: classification error rate, Gini index or entropy *instead of* RSS



Classification Trees

$\hat{p}_{mk}$ - proportion of training observations in the mth region that are from the kth class.

$$0 \leq \hat{p}_{mk} \leq 1$$

Classification error

$$E = 1 - \max_k(\hat{p}_{mk}).$$

fraction of the training observations in that region that do not belong to the most common class.

Entropy

Gini Index

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

measure of total variance across the K classes.
Gini index takes on a small value if all of the $\hat{p}_{mk}$'s are close to zero or one

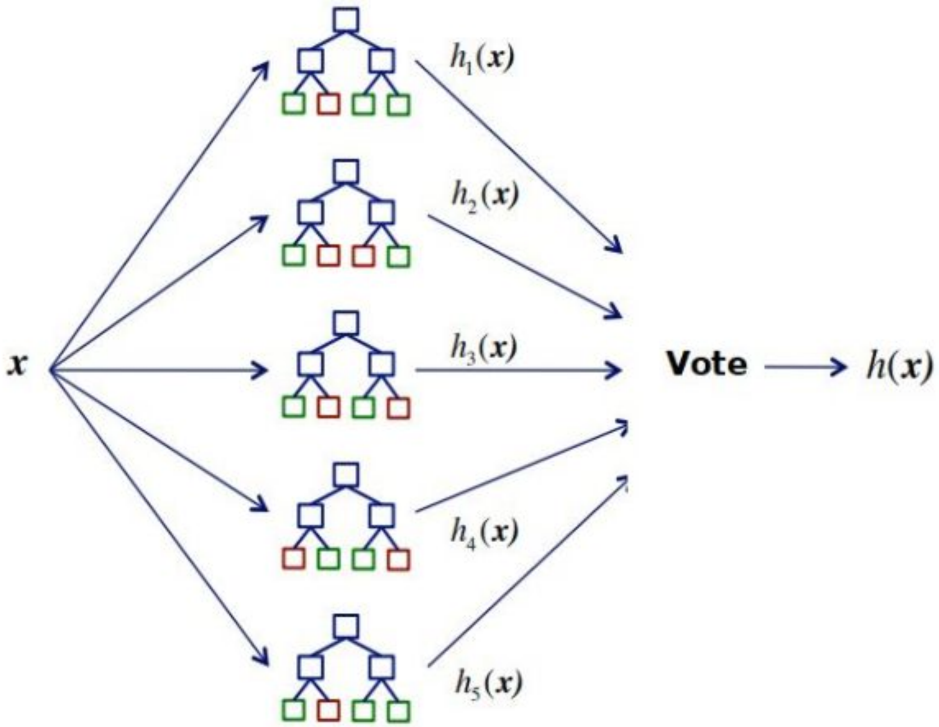
Classification Trees

- Trees can be displayed graphically & are easily interpreted even by a non-expert (especially if they are small)
- Trees can easily handle qualitative predictors without the need to create dummy variables
- Not robust to small changes in the data.
- High error (compared to other approaches)

Ensemble

- Single tree -- not very good
- Combinations of separate trees -- dramatic improvements in prediction accuracy, at the expense of some loss in interpretation

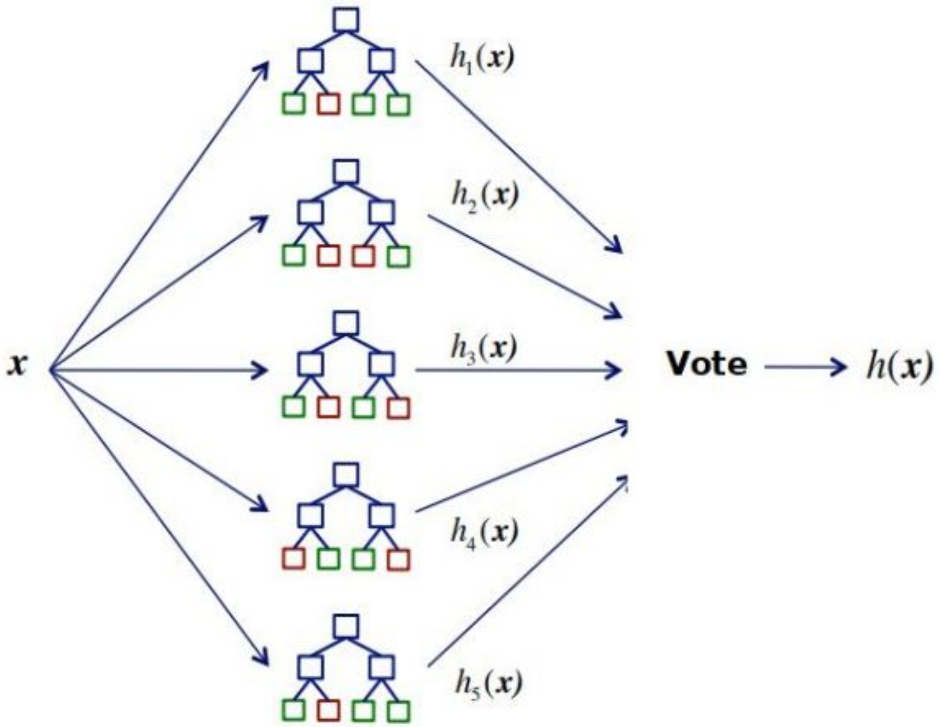
a group of weak learners can combine together to construct a strong learner !



Ensemble

Decision trees aggregation methods:

- Bagging
- Random forest
- Boosting



Bagging (a.k.a Bootstrap aggregation)

- general-purpose procedure for reducing the variance of a statistical learning method

Bootstrap

Powerful statistical tool that can be used to quantify the uncertainty associated (accuracy) with a given estimator or statistical learning method

Some kind of statistics/parameter:

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}},$$

Do not have information about whole population, so have to make do with sample estimates

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}$$

Q: How well does ^α represent true α of a population?

Bootstrap

Powerful statistical tool that can be used to quantify the uncertainty associated (accuracy) with a given estimator or statistical learning method

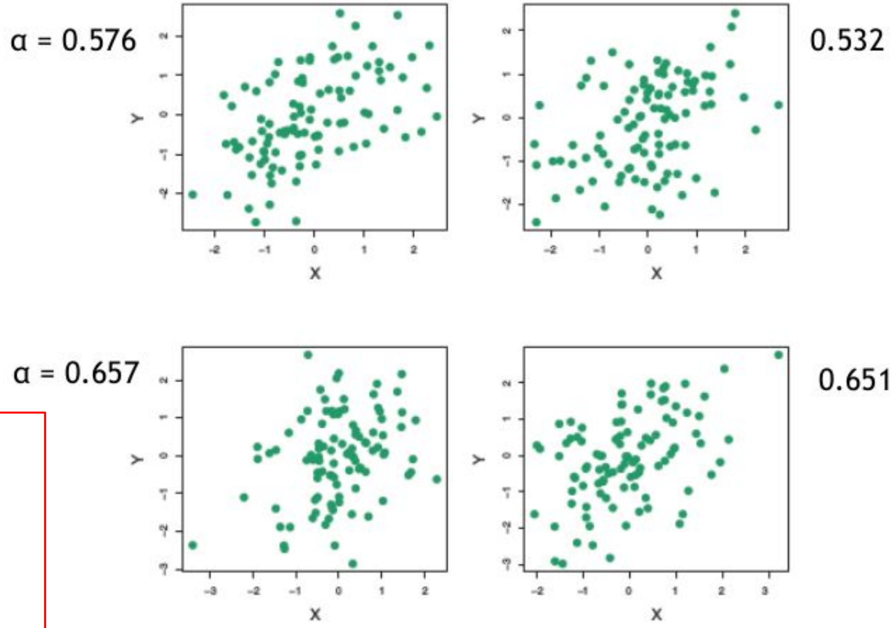
Some kind of statistics/parameter:

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

Do not have information about whole population, so have to make do with sample estimates

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \cdots}{\hat{\sigma}_X^2 + \hat{\sigma}\cdots}$$

Q: How well does ^α represent true α of a population?

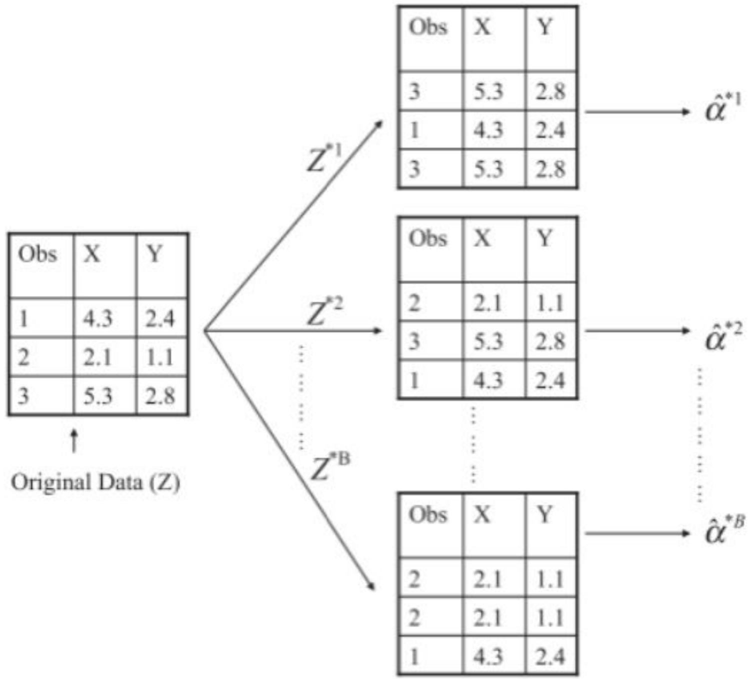


Bootstrap

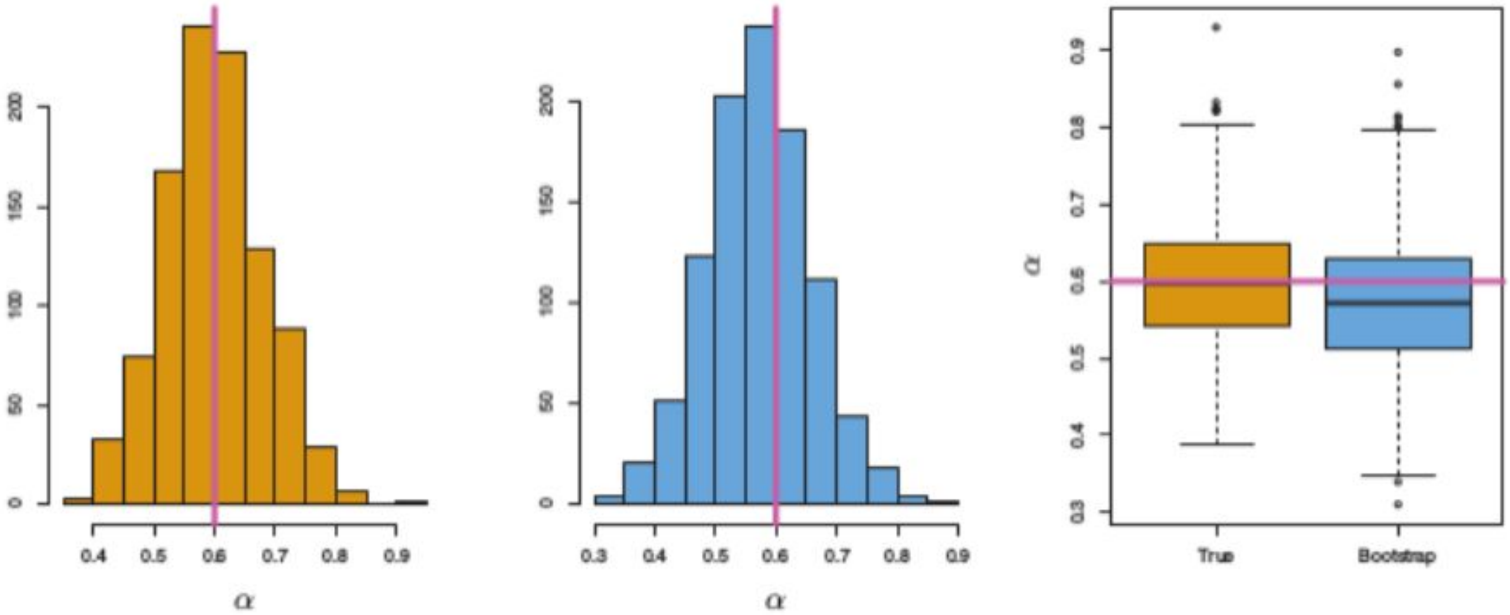
Powerful statistical tool that can be used to quantify the uncertainty associated (accuracy) with a given estimator or statistical learning method

Random sampling with replacement

n = 3
Z* – bootstrap data set
B – number of sampling iterations



Bootstrap -- random sampling with replacement

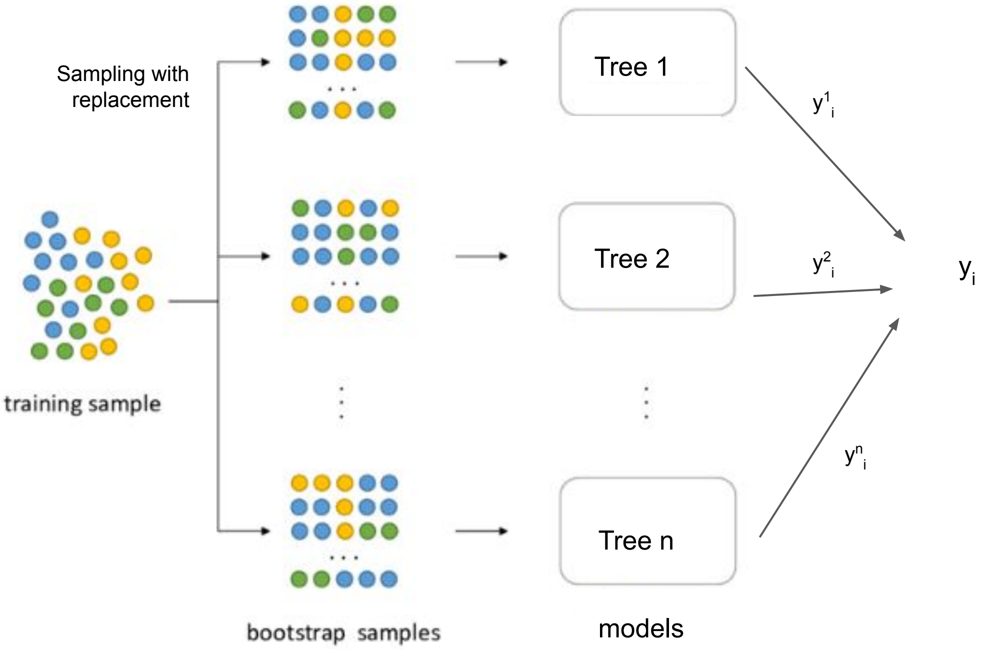


1,000 simulated data sets from **true population**

1,000 **bootstrap** samples from a single data set

Bagging (a.k.a Bootstrap aggregation)

- general-purpose procedure for reducing the variance of a statistical learning method

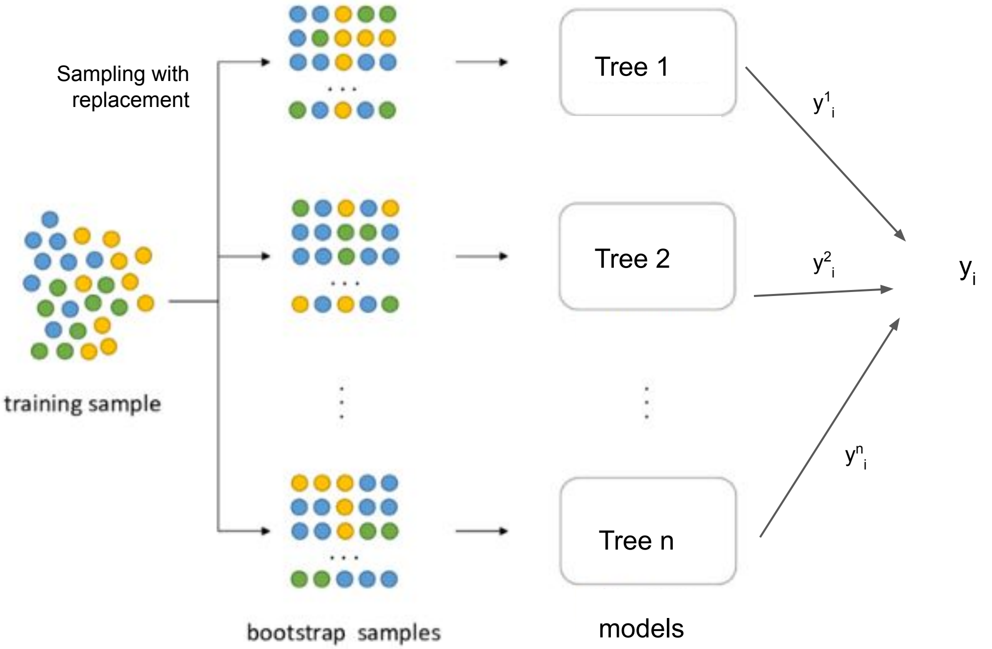


Bagging (a.k.a Bootstrap aggregation)

- general-purpose procedure for reducing the variance of a statistical learning method

Steps:

1. Create bootstrap sets
2. Fit models
3. Average predictions



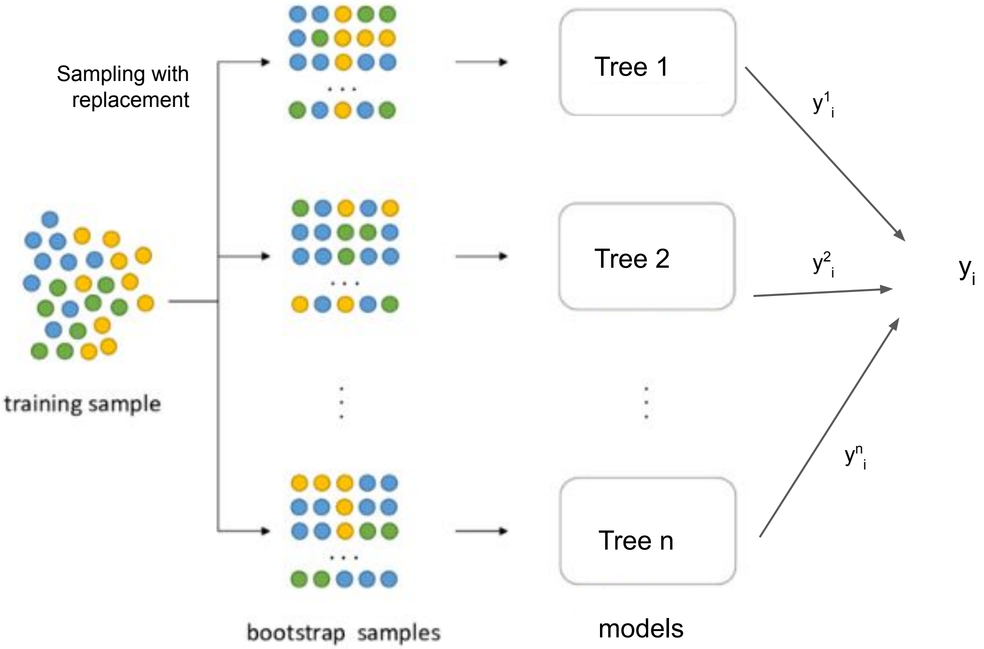
Bagging (a.k.a Bootstrap aggregation)

- general-purpose procedure for reducing the variance of a statistical learning method

Steps:
1. Create bootstrap sets
2. Fit models
3. Average predictions

Each tree has high variance, but low bias

Averaging these N trees reduces the variance

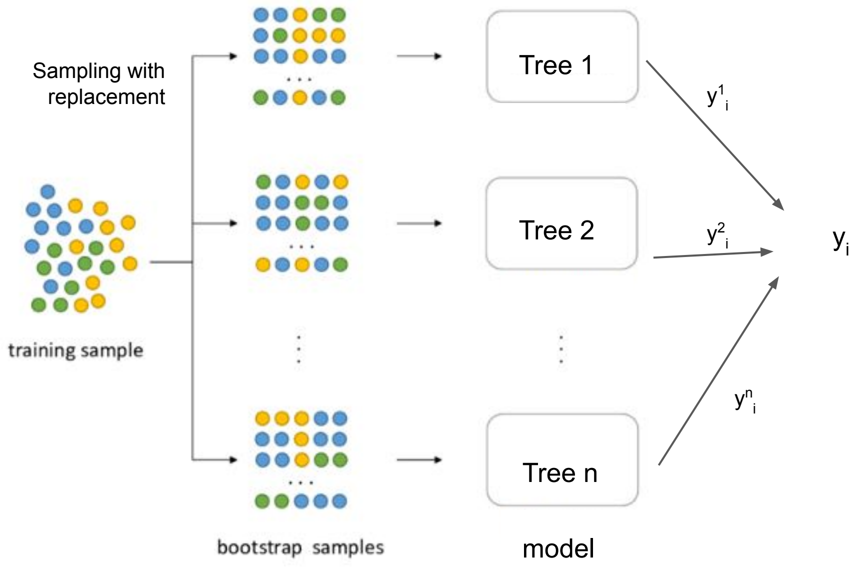


Out-of-Bag Estimate

- On average only 2/3 of observations are used in each sample.
- The remaining 1/3 of the observations not used to fit a given bagged
- Tree are referred to as the out-of-bag (OOB) observations.

Using that we can compute:

Overall OOB Mean Square Error
(for a regression problem)
or
Overall OOB classification error rate
(for a classification problem)



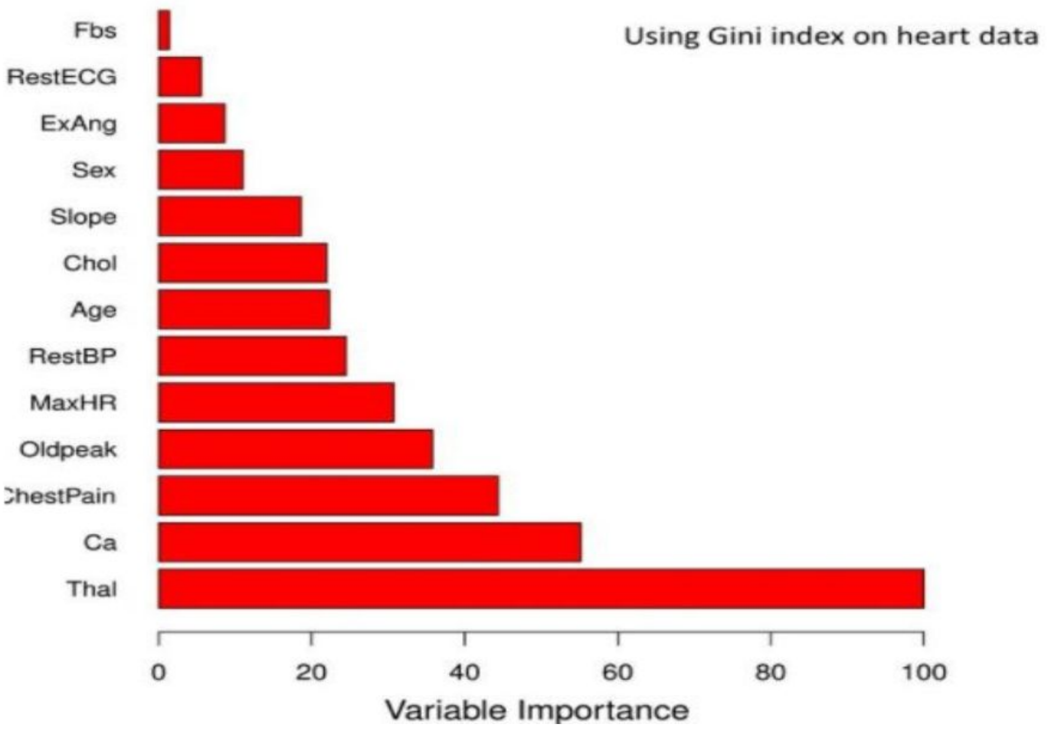
Interpretability of Bagged Trees

Bagging improves prediction accuracy at the expense of interpretability

Can obtain an overall summary of the importance of each predictor:

- RSS (for bagging regression trees) -- record the total amount that the RSS decreases due to splits over a given predictor, averaged over all N trees.
- Gini index (for bagging classification trees) -- total amount that the Gini index decreases by splits over a given predictor, averaged over all N trees.

Interpretability of Bagged Trees



Problem with Bagging

One very strong predictor + a number of moderately strong predictors

- All of the bagged trees will look quite similar.
- The predictions from the bagged trees will be highly correlated.

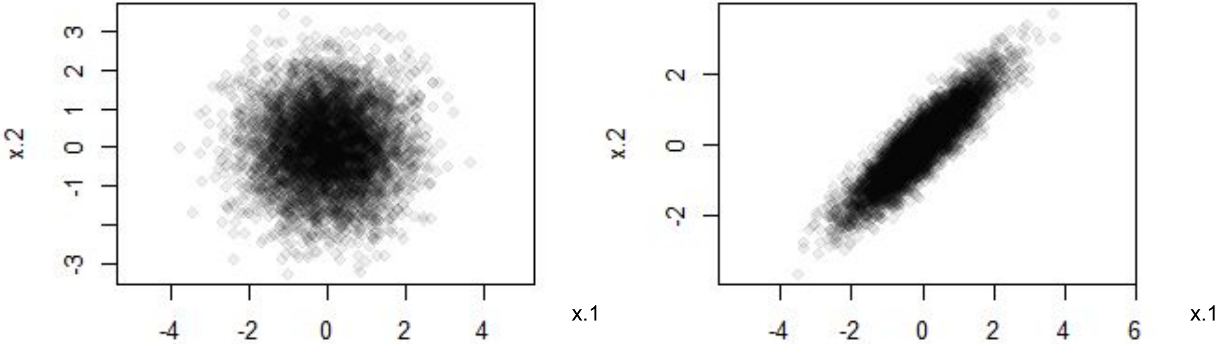
Averaging many highly correlated quantities doesn’t lead to great reduction in variance.

Problem with Bagging

One very strong predictor + a number of moderately strong predictors

- All of the bagged trees will look quite similar.
- The predictions from the bagged trees will be highly correlated.

Averaging many highly correlated quantities doesn’t lead to great reduction in variance.



Problem with Bagging

One very strong predictor + a number of moderately strong predictors

- All of the bagged trees will look quite similar.
- The predictions from the bagged trees will be highly correlated.

averaging many highly correlated quantities does not lead to as large of a reduction in variance as averaging many uncorrelated quantities

Option: Random forests provides an improvement over bagged trees by using a small tweak that decorrelates the trees.

Random Forest

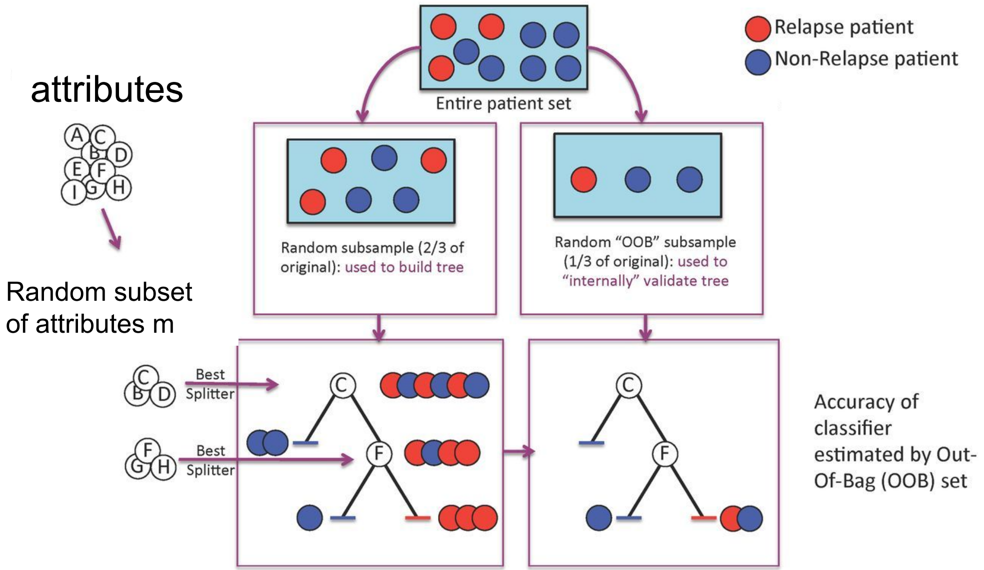
DATA: One very strong predictor + a number of moderately strong predictors

RANDOM FOREST: for each split takes the random selection of m features rather than using all p

Random Forest

DATA: One very strong predictor + a number of moderately strong predictors

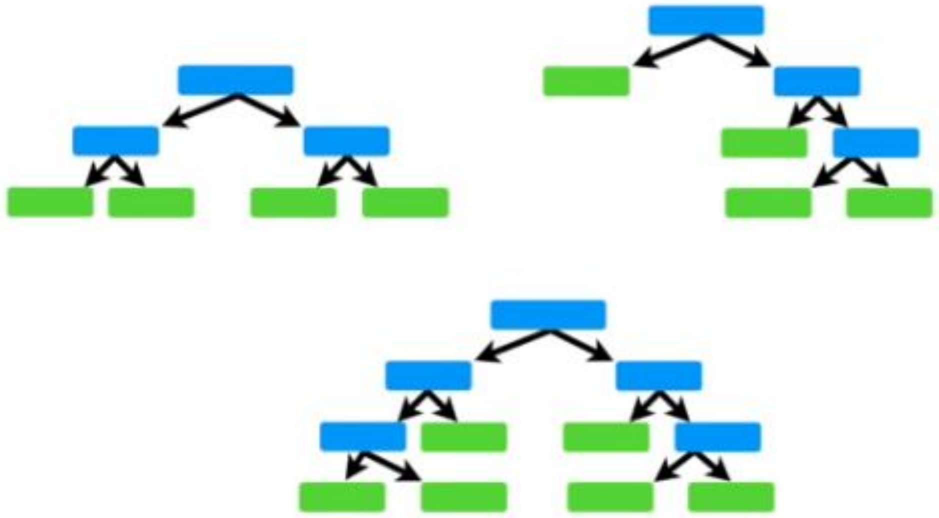
RANDOM FOREST: for each split takes the random selection of *m* features rather than using all *p*



Random Forest

DATA: One very strong predictor + a number of moderately strong predictors

RANDOM FOREST: for each split takes the random selection of *m* features rather than using all *p*



- We force trees to take up different topology and that way become uncorrelated

- The average of the resulting trees becomes less variable and hence more reliable

Random Forest

DATA: One very strong predictor + a number of moderately strong predictors

RANDOM FOREST: for each split takes the random selection of *m* features rather than using all *p*

- Random forest is good when we have a lot of correlated predictors
- When m = p, random forest becomes ...?

Boosting

- boosting is a general approach that can be applied to many statistical learning methods for regression or classification
- While bagging and random forest aims to decrease variance, boosting aims to decrease bias
- Boosted trees try to improve the model fit by considering past fit

Boosting

- boosting is a general approach that can be applied to many statistical learning methods for regression or classification
- While bagging and random forest aims to decrease variance, boosting aims to decrease bias
- Boosted trees try to improve the model fit by considering past fit

$\boxed{\text{HOW?}}$

Boosting

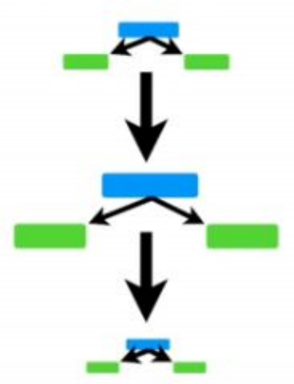
- boosting is a general approach that can be applied to many statistical learning methods for regression or classification
- While bagging and random forest aims to decrease variance, boosting aims to decrease bias
- Boosted trees try to improve the model fit by considering past fit

HOW?

- trees are grown sequentially: each tree is grown using information from previously grown trees
- Original dataset is modified with each tree fit

Boosting

1. Each of the created trees is small (few terminal nodes, quite usual to have only two terminal nodes)
2. Each tree is made sequentially
3. Some trees get “more to say” i.e.more weight in the final decision



Tree with 2 terminal nodes is called a stump

Adaboost

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

Create best stump

Adaboost

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

Create best stump

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

Increase weight for observations that were classified incorrectly

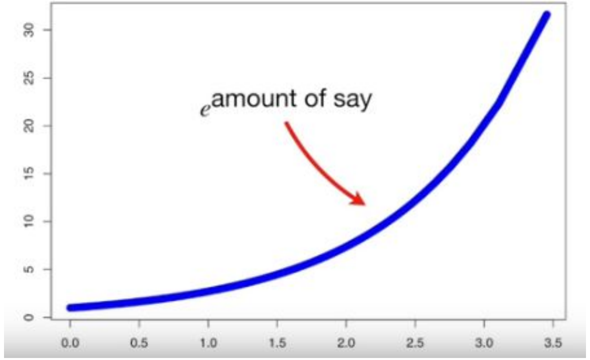
Decrease the weight for correctly classified observations

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

$$\text{New Sample Weight} = \text{sample weight} \times e^{\text{amount of say}}$$

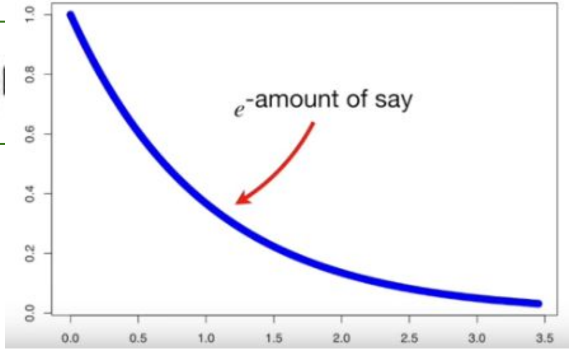
$$\text{New Sample Weight} = \text{sample weight} \times e^{-\text{amount of say}}$$

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8



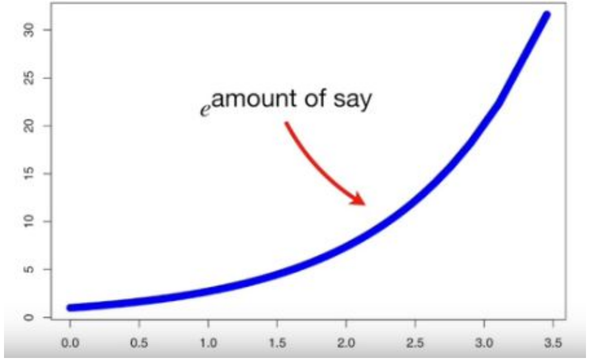
$$\text{New Sample Weight} = \text{sample weight} \times e^{\text{amount of say}}$$

New Sample Weight = sample weig



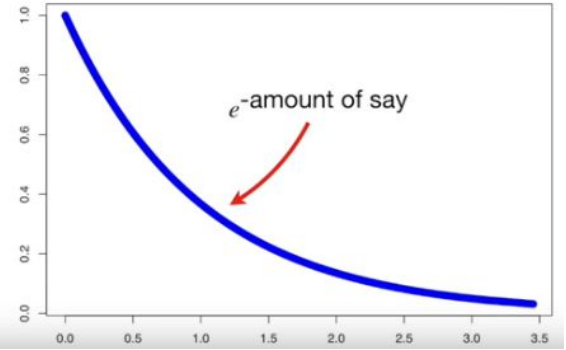
$$\text{Amount of Say} = \frac{1}{2}\log\left(\frac{1 - \text{Total Error}}{\text{Total Error}}\right)$$

Chest Pain	Blocked Arteries	Patient Weight	Heart Disease	Sample Weight
Yes	Yes	205	Yes	1/8
No	Yes	180	Yes	1/8
Yes	No	210	Yes	1/8
Yes	Yes	167	Yes	1/8
No	Yes	156	No	1/8
No	Yes	125	No	1/8
Yes	No	168	No	1/8
Yes	Yes	172	No	1/8

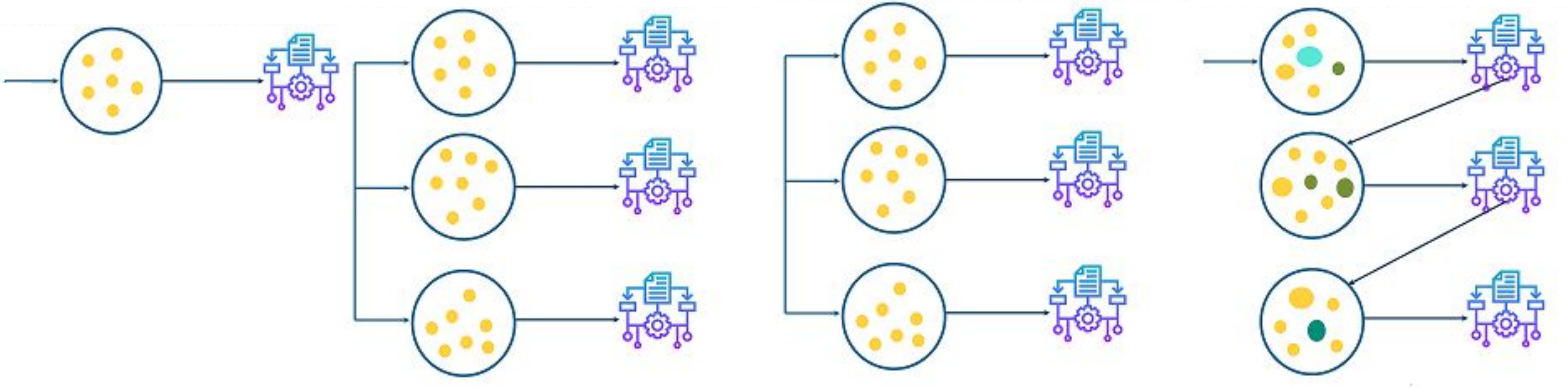


$$\text{New Sample Weight} = \text{sample weight} \times e^{\text{amount of say}}$$

New Sample Weight = sample weigl



Comparison



Single Tree:

- Complete training set

Random Forest:

- Tree fitting - parallel
- Random sampling with replacement
- Use random sample of m features for each split

Reduces variance and decorrelated trees

Bagging:

- Tree fitting - parallel
- Random sampling with replacement
- Use all p features

Reduces variance

Boosting:

- Tree fitting - sequential
- Random sampling with replacement over weighted data

Reduces Bias

Ensemble (stacking)

- Works not only for trees
- Can use different models that capture different properties of data

a group of **weak learners** can combine together to construct a strong learner !

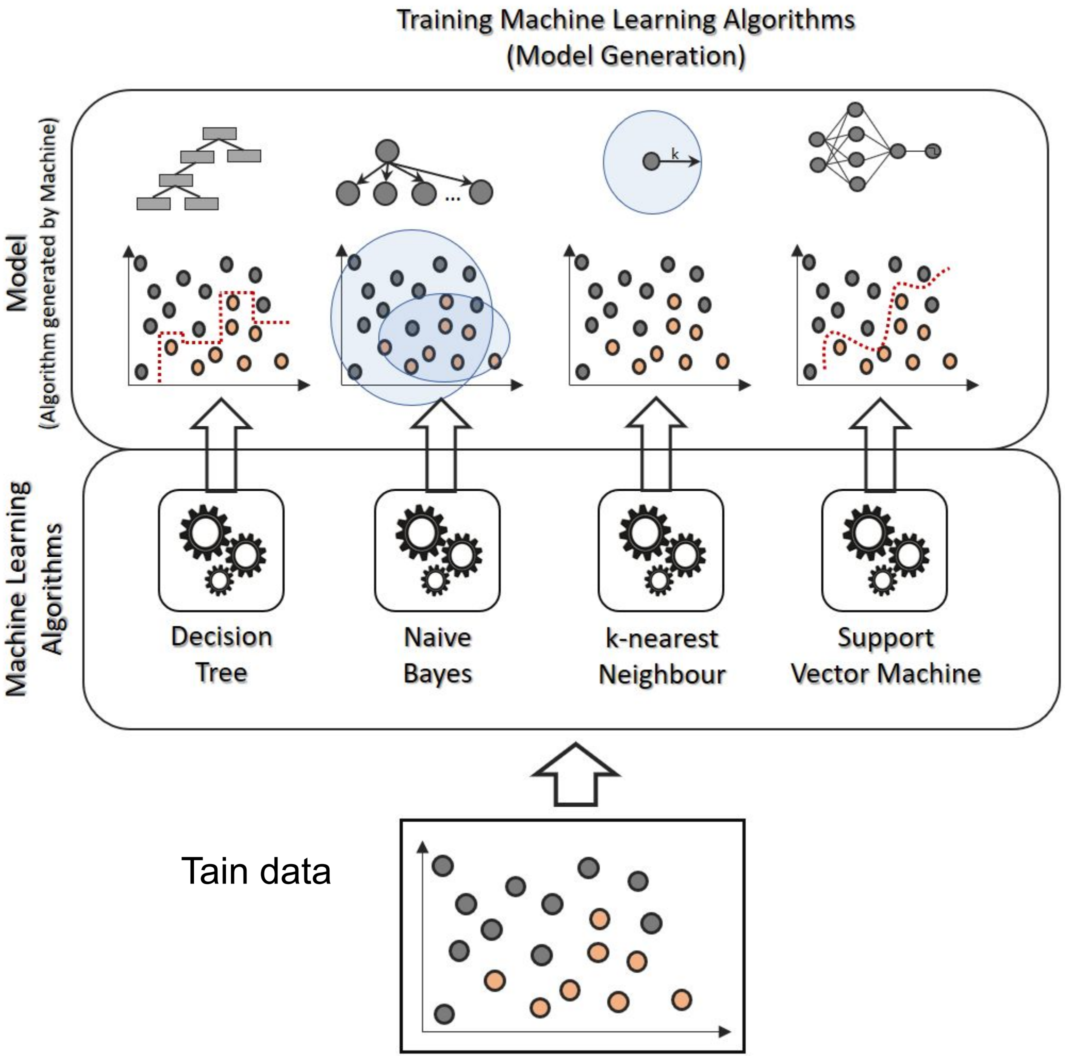
a group of **specific/specialized learners** can combine together to construct a strong learner !

Ensemble (stacking) of methods

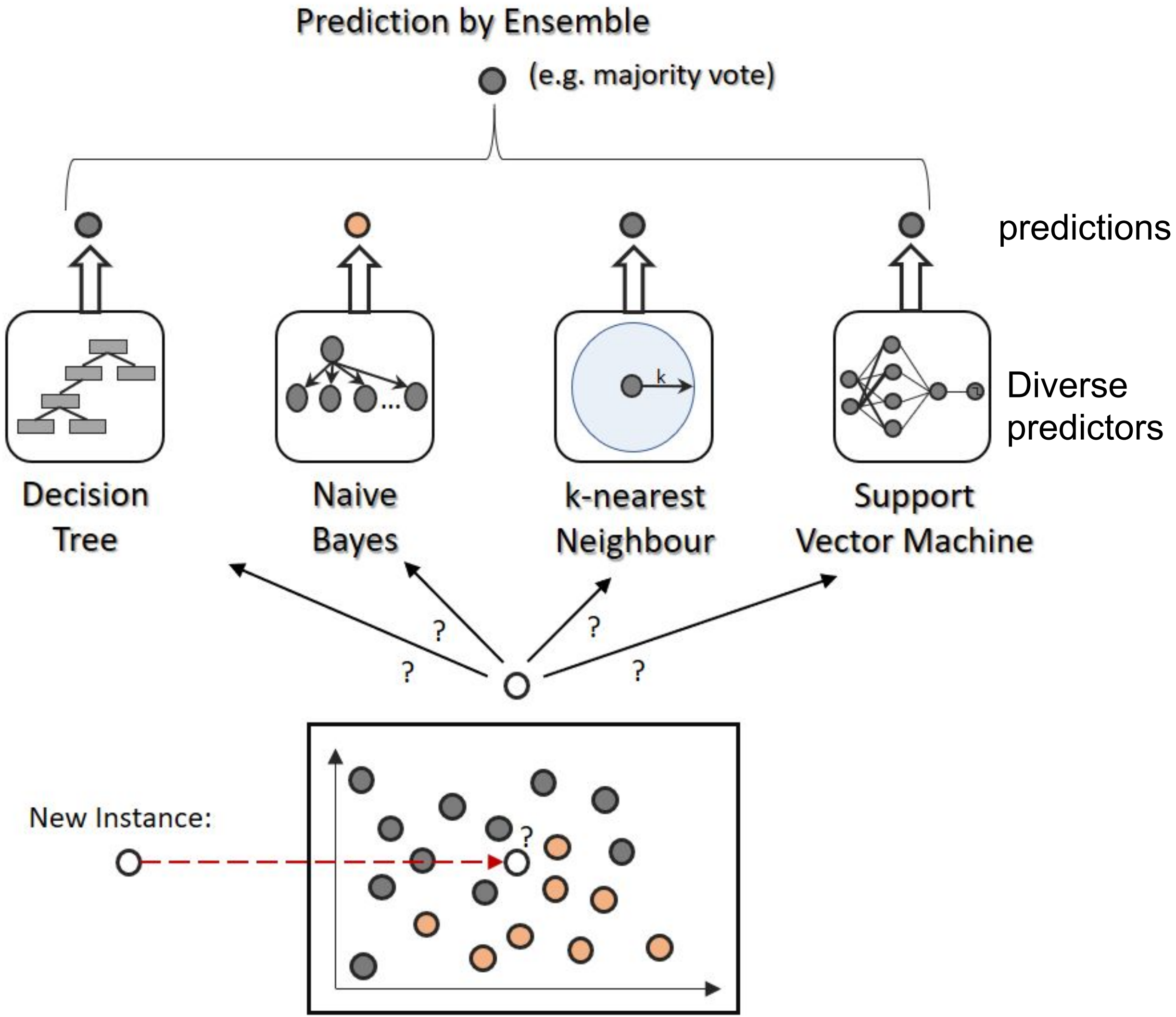
Fit diverse models:

- Complexity
- Decision boundaries

GOAL: improve prediction



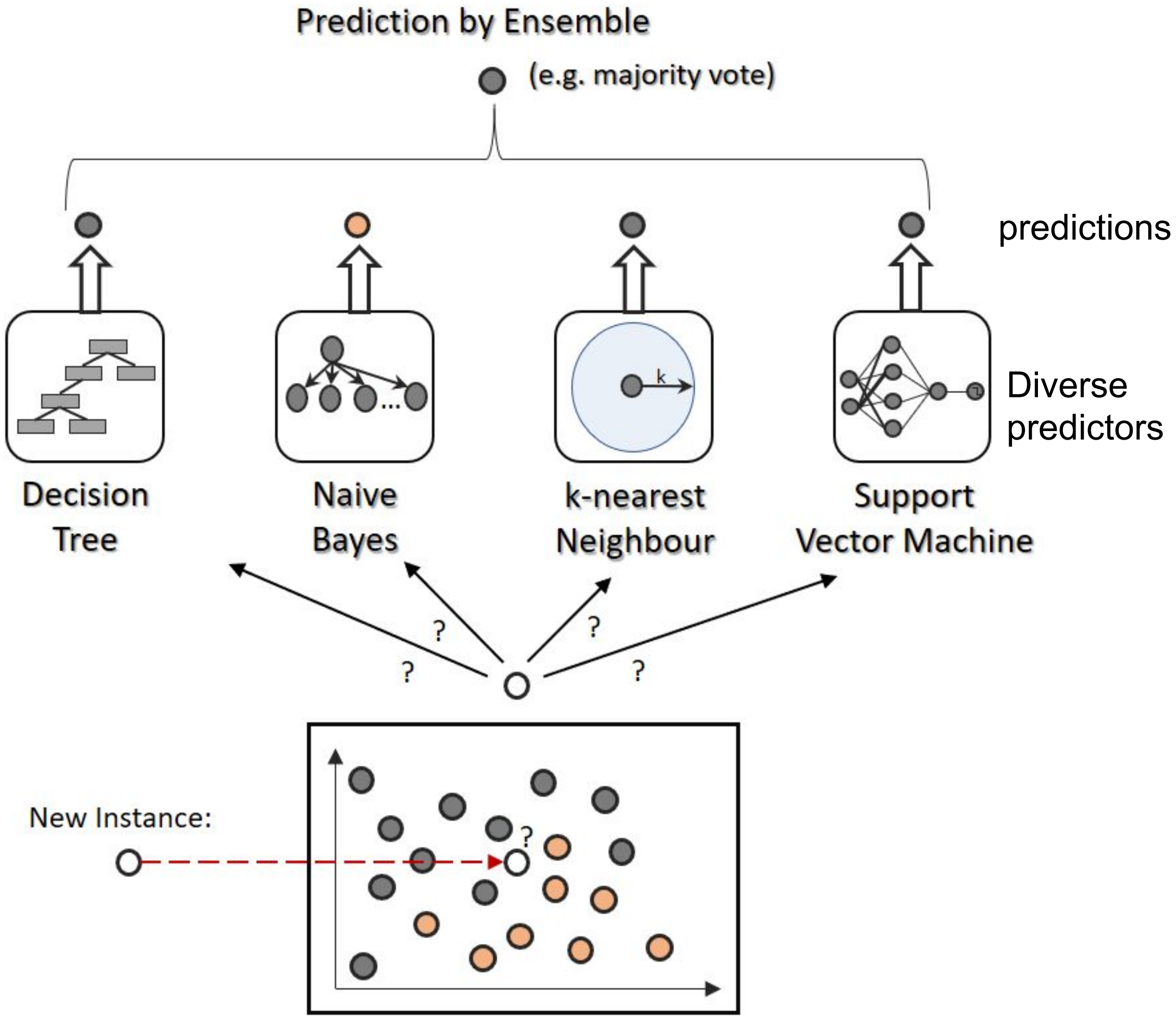
Ensemble (stacking) of methods



Ensemble

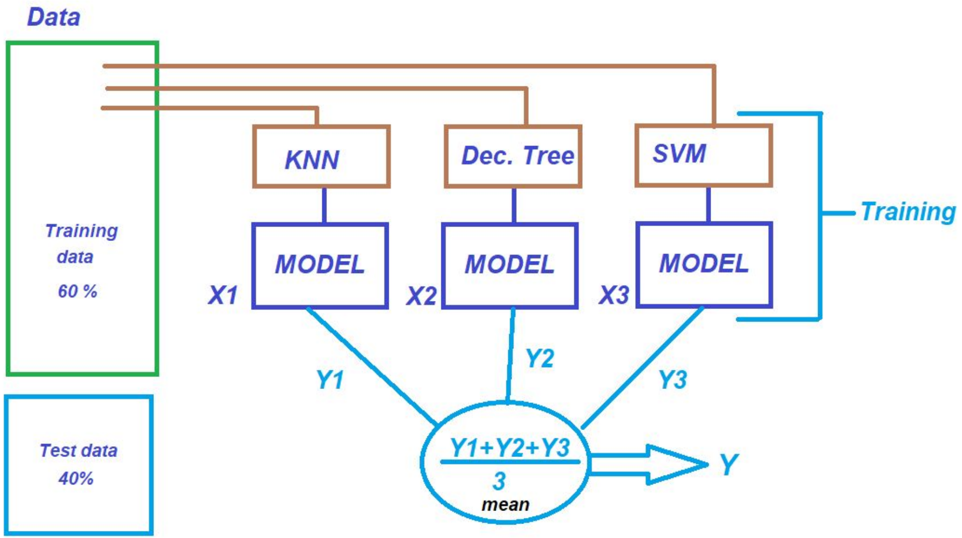
variety of ensembling methods:

- voting or averaging the predictions
- Linear models (multiple regression, logistic regressions)
- k-nearest neighbours
- boosting trees



Ensemble (stacking) of methods

- Often beat state-of-the-art academic benchmarks and are widely used to win Kaggle competitions. Improves prediction.
- Usually computationally expensive
- Easy to overfit if not careful



More information about Ensemble methods (if interested):

https://mlwave.com/kaggle-ensembling-guide/

http://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml04.icdm06long.pdf

http://www.columbia.edu/~rsb2162/PBGH-SIGKDDExp.pdf

https://www.netflixprize.com/assets/GrandPrize2009_BPC_BigChaos.pdf

…

CART

If you prefer reading books:

1. **An Introduction to Statistical Learning with Applications in R** chapter 8
2. Principles of data mining, David Hand, chapter 10.5

3. Pattern Classification, R.O.Duda, P.E.Hart, D.G Stork chapter 8
4. Elements of statistical learning, T.Hastie et.al, chapter 9-10

Really in depth

HOMEWORK:

Intro to Neural Networks
https://www.youtube.com/watch?v=aircAruvnKk&list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi

Classification algorithm overview

	2 classes	>= 2 classes
parametric	LDA, QDA, logistic regression, support vector machines	Naive bayes
non-parametric		Decision trees, KNN