Class

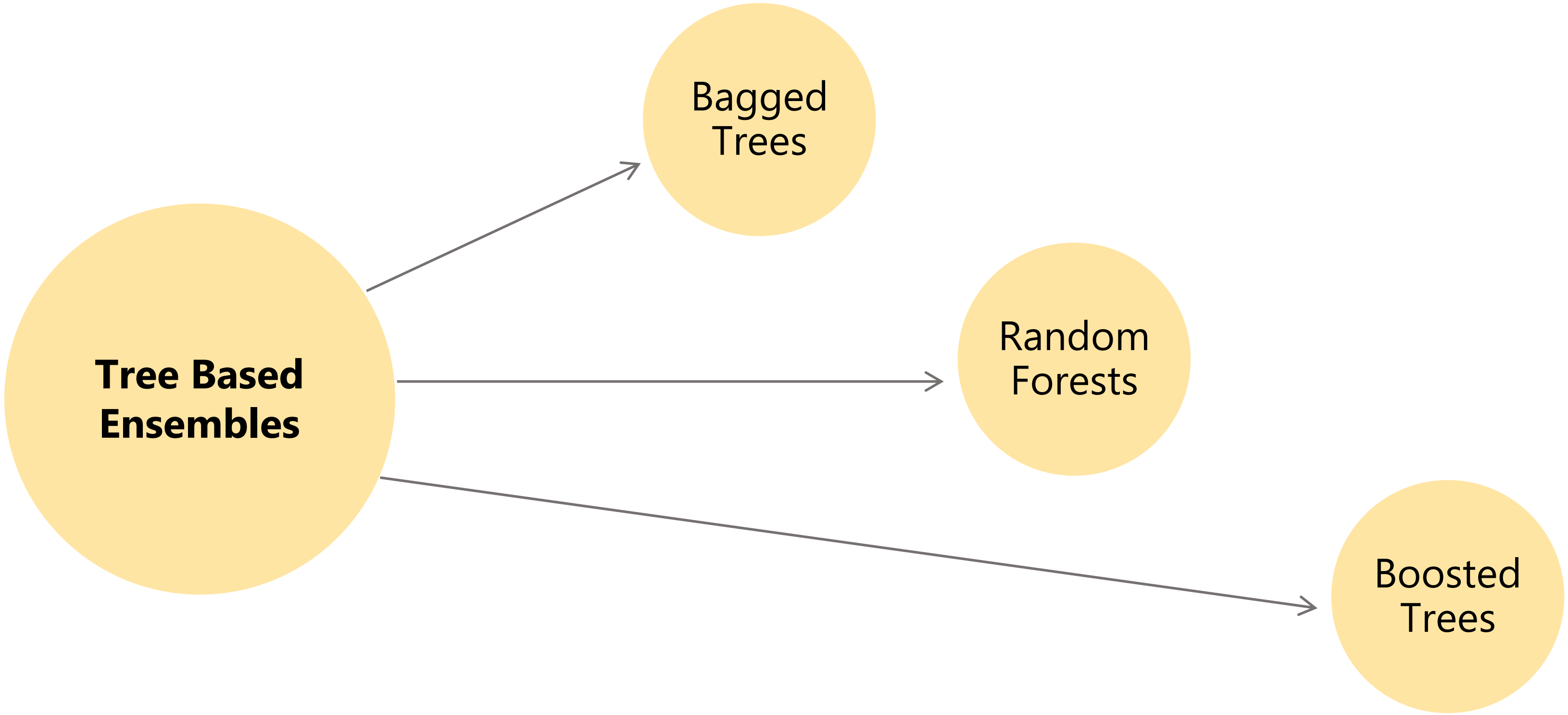
Tree Based Model

Topic

Tree Based Ensembles: Bagged Trees and Random Forests

Agenda

Another category of machine learning models → Ensemble Models



Tree Based Ensembles Overview

Single model

One good single regression model or one good single classification model

Ensemble model

Combine several simple models together to create one single meta model

Powerful models

Regression ensemble

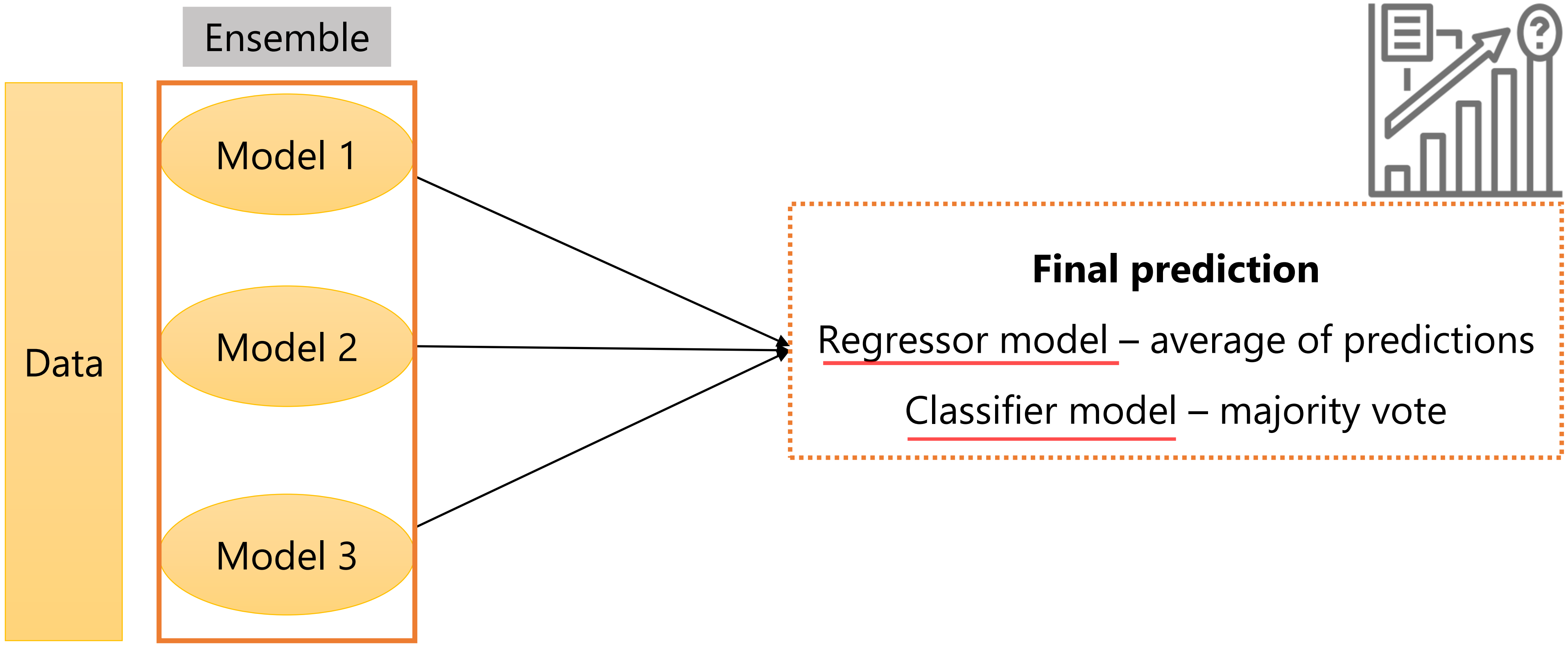
Final prediction is the average predictions from all the simple regression models in the ensemble

Classification ensemble

Final prediction is the majority vote out of all the predictions made by the constituent classifiers that make up this ensemble

Tree Based Ensembles Overview

Schematic working of ensembles



Tree Based Ensembles Overview

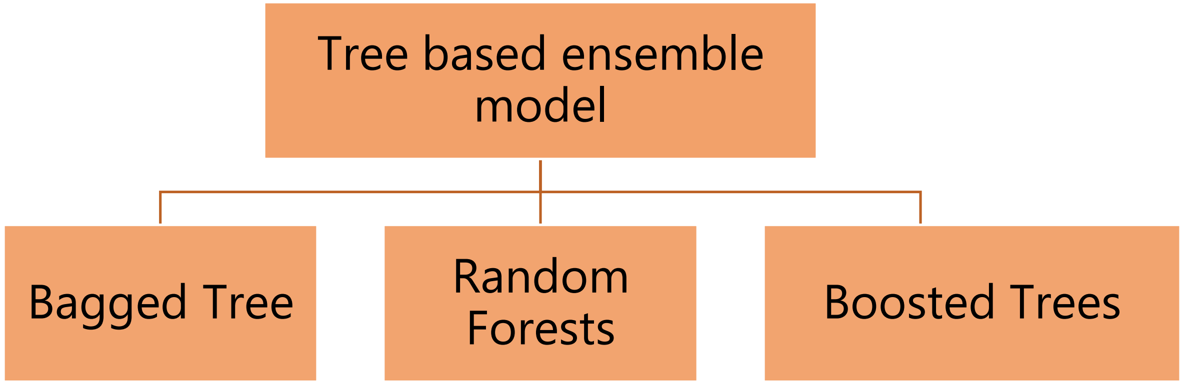
Tree based ensemble model

- More popular choice
- Only a subset of total data is used by each base learner

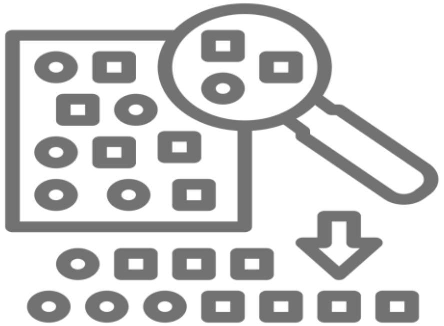
The way this data set is fed into each of the base learners is based on a **data sampling scheme**

Different sampling schemes give rise to different types of tree based ensembles

Tree Based Ensemble Models



Sampling Scheme	Bootstrap Sampling	Bootstrap Sampling + Feature Sampling	Data Reweighing
Base Learner	Tree	Tree	Tree



Bagged Trees

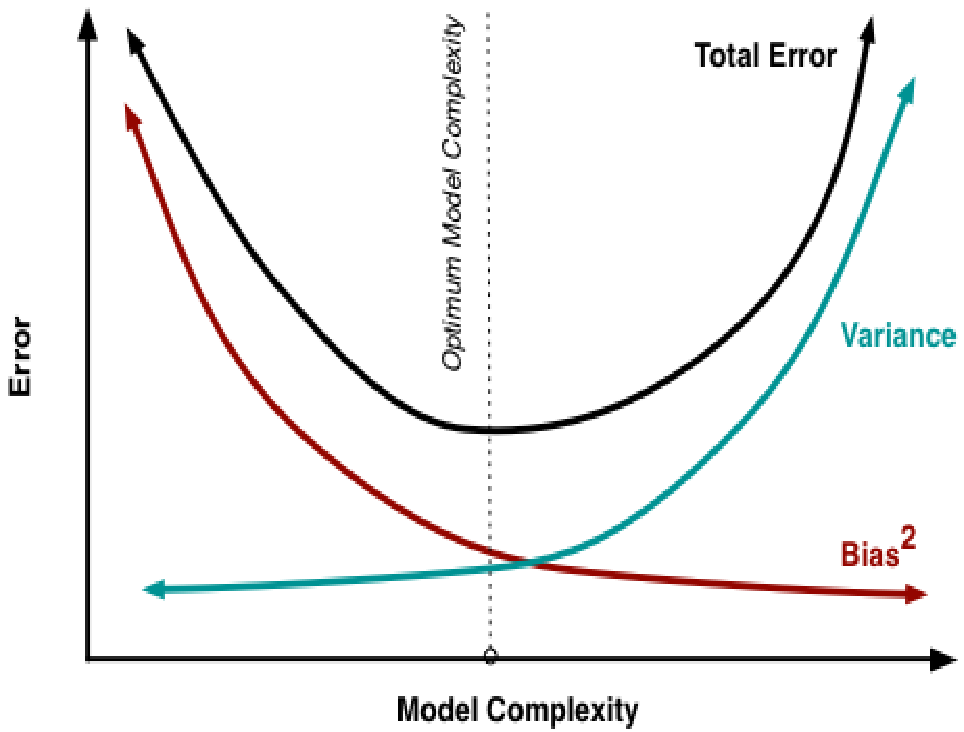
The total error due to any machine learning algorithm = In-sample error + Out-of-sample error

In-sample error: Inability of an ML algorithm to fit the training data well

Out-of-sample error: Inability of a model to generalize well

Good predictions on training data but is unable to do so on unseen data

Bagged Trees



Error = Bias + Variance = In-sample Error + Out-of-sample Error

Trade offs between the model complexity and the error in models

More complicated models have very **low in-sample error but have a high out-of-sample error**

Simpler models have **low out-of-sample error but high in-sample error**

Theoretically, there is a limit to minimum error that can be achieved

Reduce error further by decreasing in-sample error and out-of-sample error simultaneously

Bagged Trees

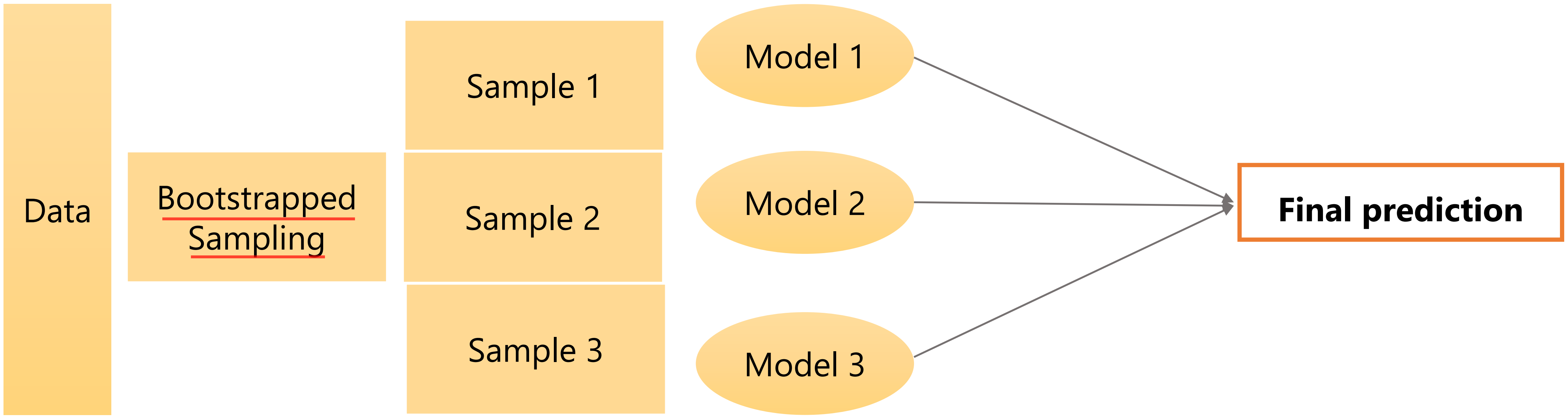
Use tree based models as base learners to reduce in-sample error

While training a tree based ensemble the constituent tree models are allowed to grow **many levels deep**

The intricacies of training data are captured intimately, thereby reducing the in-sample error

Bagged Trees

In the case of **Bagged trees** each of the unpruned trees are fed bootstrapped samples of original data set



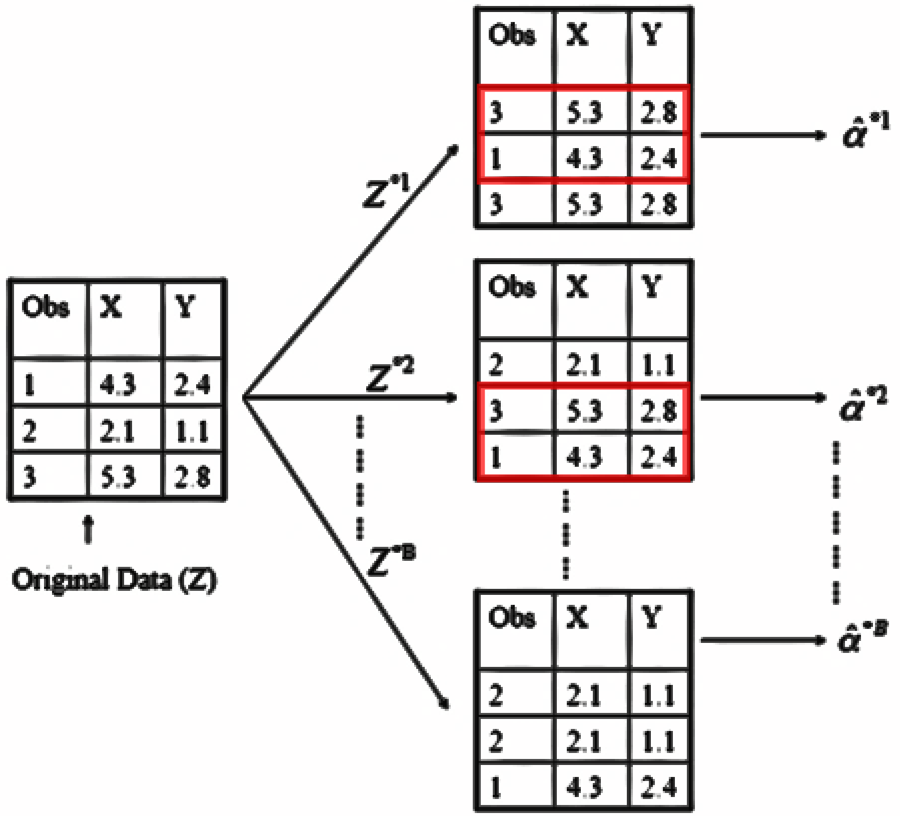
Bootstrapped Sampling

Bootstrapped sampling simply refers to **sampling by replacement**

Blue and red dots are repeated more often in the samples than they are present in the original data

Bootstrapped Sampling

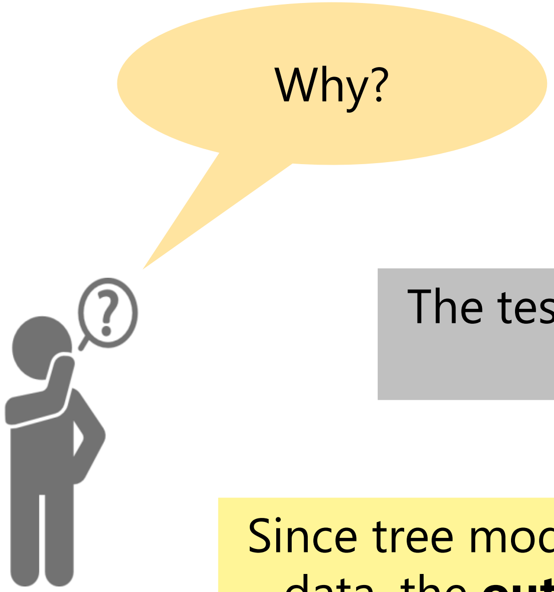
Bootstrapped Sampling at the data level



Bootstrapped Sampling

Bootstrapped Sampling helps in reducing out of sample error

If an unpruned decision tree is fit into any data set then the model will have **very high out-of-sample error**



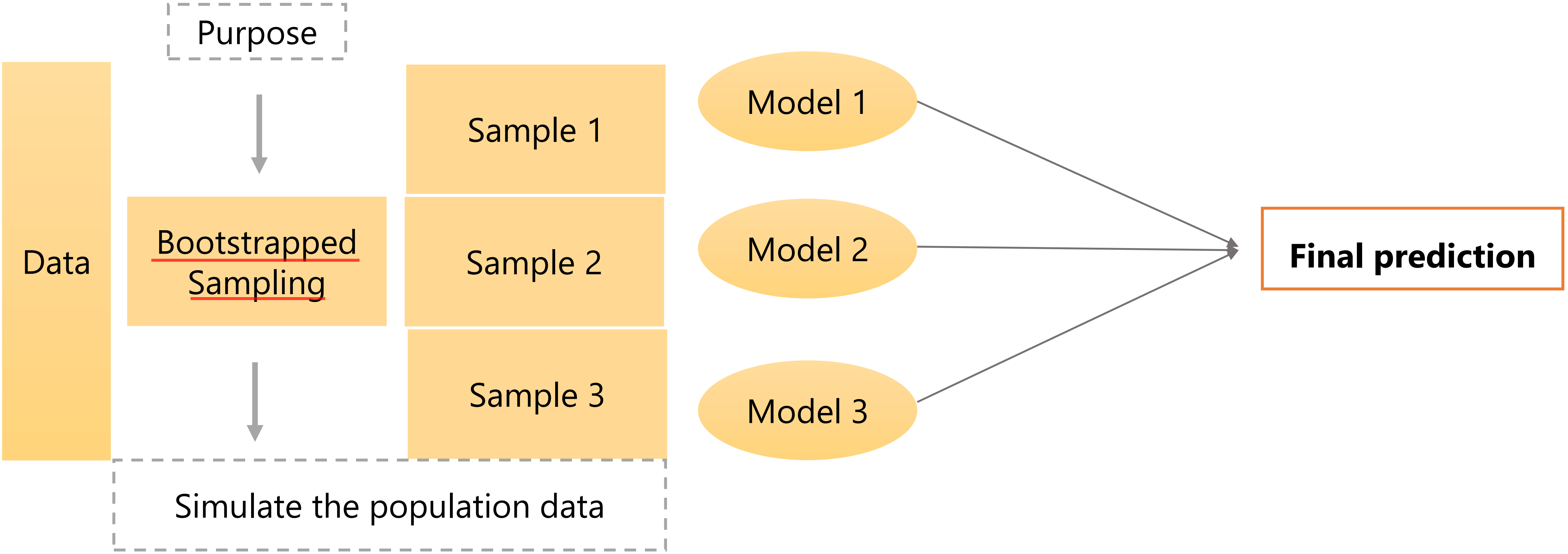
The test data can be very different from training data

Since tree model is being overfitted on training data, the **out-of-sample error will be high**

Hypothetically, if unpruned tree model is being fitted on **total population data** the error will be very low

Low error due to all the variation and variety in data has been already seen by the model as the population data has been used to train the model

Bootstrapped Sampling



Realistic view of the data generation process

Synthetically generate variation and variety in the training data itself

Bagged Trees

Characteristics of Bagged Model

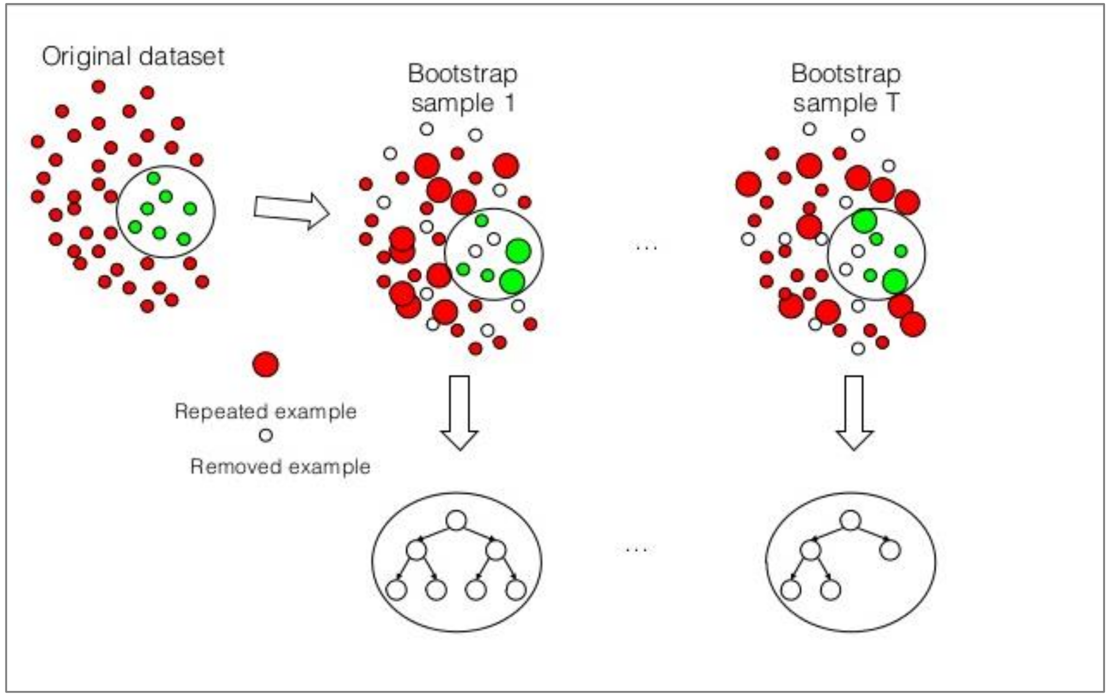
Using **unpruned decision trees** as base learners

Using **Bootstrapped Sampling** to create samples that are fed to each of the base learners

Peculiarities of Bagged Trees

Bagged tree ensemble is comprised of multiple decision trees

Not interpretable as a linear model or simple decision tree



Qualitative statement on ensemble models

Identify the important predictors by looking at **Variable Importance**

Variable Importance

Variable Importance - Averaging or summing the improvement in **Gini or Entropy for a classification model** and **RSS for a regression model** for all the variables

Bagged tree ensemble model contains many tree models

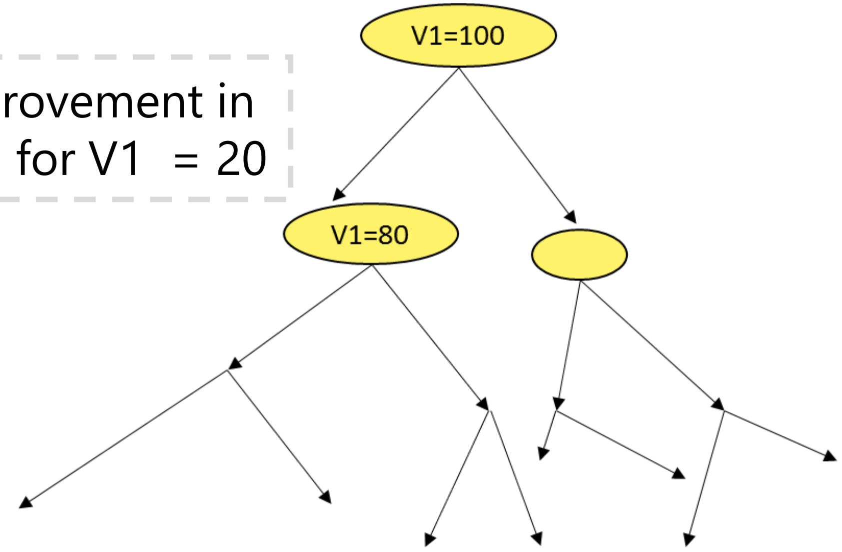
Feature importance of each variable in each of the constituent trees

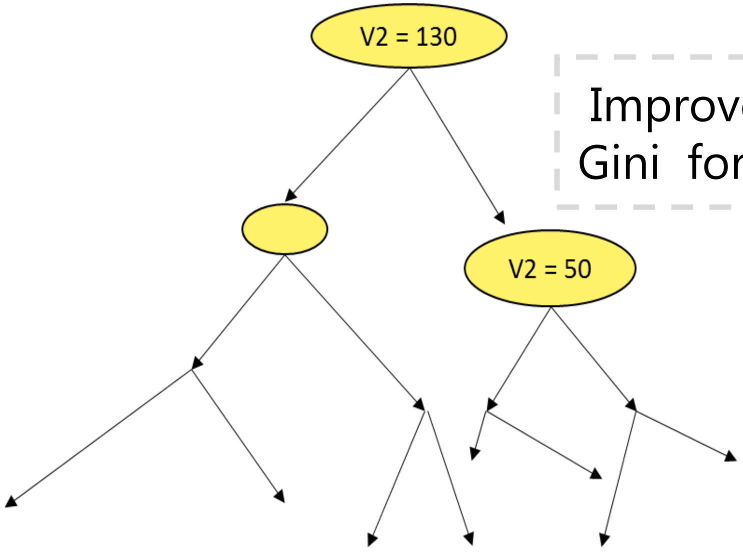
Tracking the decrease in Gini metric and weighing this decrease appropriately

Variable Importance

Example

Improvement in Gini for V1 = 20





Improvement in Gini for V2 = 80

Tree 1
Gini Measure for each split

Tree 2
Gini Measure for each split

Computing variable importance for all the variables used in the split

Variable Importance

Ensemble has N trees

Improvement in Gini/RSS Across Splits

Variable	Tree 1	Tree 2	Tree 3	.....	Tree N
V1	300	30	12	.....	0
V2	600	0	200	.....	150
...	...	...	...	.....	...
V_k	120	450	30	.....	19

Variable	Variable Importance
V1	$\frac{(300 + 30 + 12 + \cdots + 0)}{N}$
V2	$\frac{(600 + 0 + 200 + \cdots + 150)}{N}$
...	....
V_k	$\frac{(120 + 450 + 30 + \cdots + 19)}{N}$

The average values of importance measures per variable will produce a consolidated number

Parameters of Bagged Trees

What could be the user specified parameters while building a bagged tree model?

User specified parameters or **Hyperparameters**

- Number of tree used to build an ensemble
- Depth of the tree
- Number of observations per node of a tree

Parameters of Bagged Trees

User specified parameters have an implication

Different ensemble model depending on different parameters

Which among the three model?

K-Fold CV to get an estimate of out of sample error

Expensive

Out of Bag Error is generally used in most tree based models

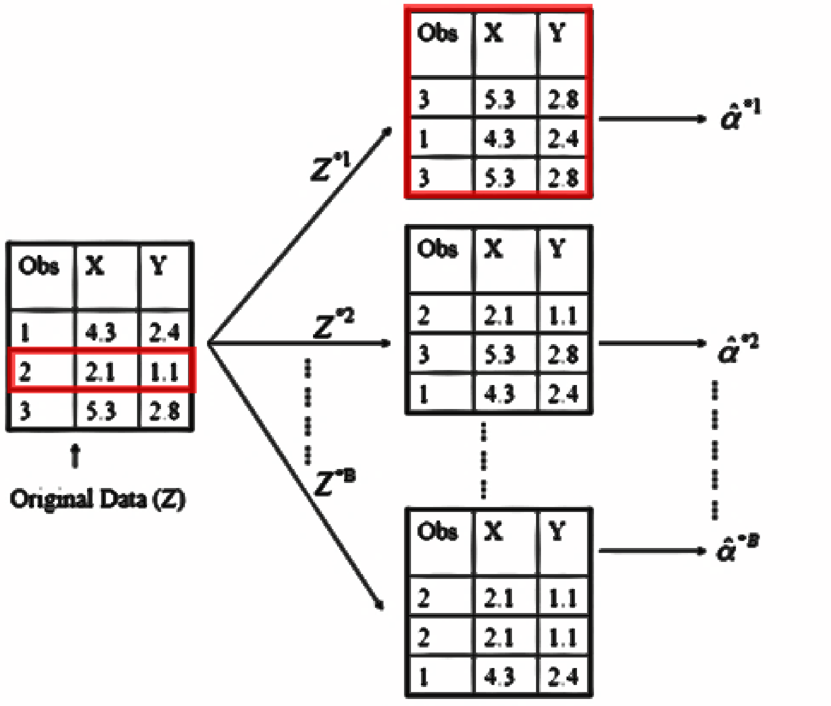
Model 1:
Trees = 100
Depth of Tree = 4

Model 2:
Trees = 150
Depth of Tree = 3

Model 3:
Trees = 500
Depth of Tree = 4

Out Of Bag Error (OOB)

In Bootstrap Sampling, some observation gets left out from the original data



Tree 1 — Row 2 is **out of bag**

Tree 2

Tree B — Row 3 is **out of bag**

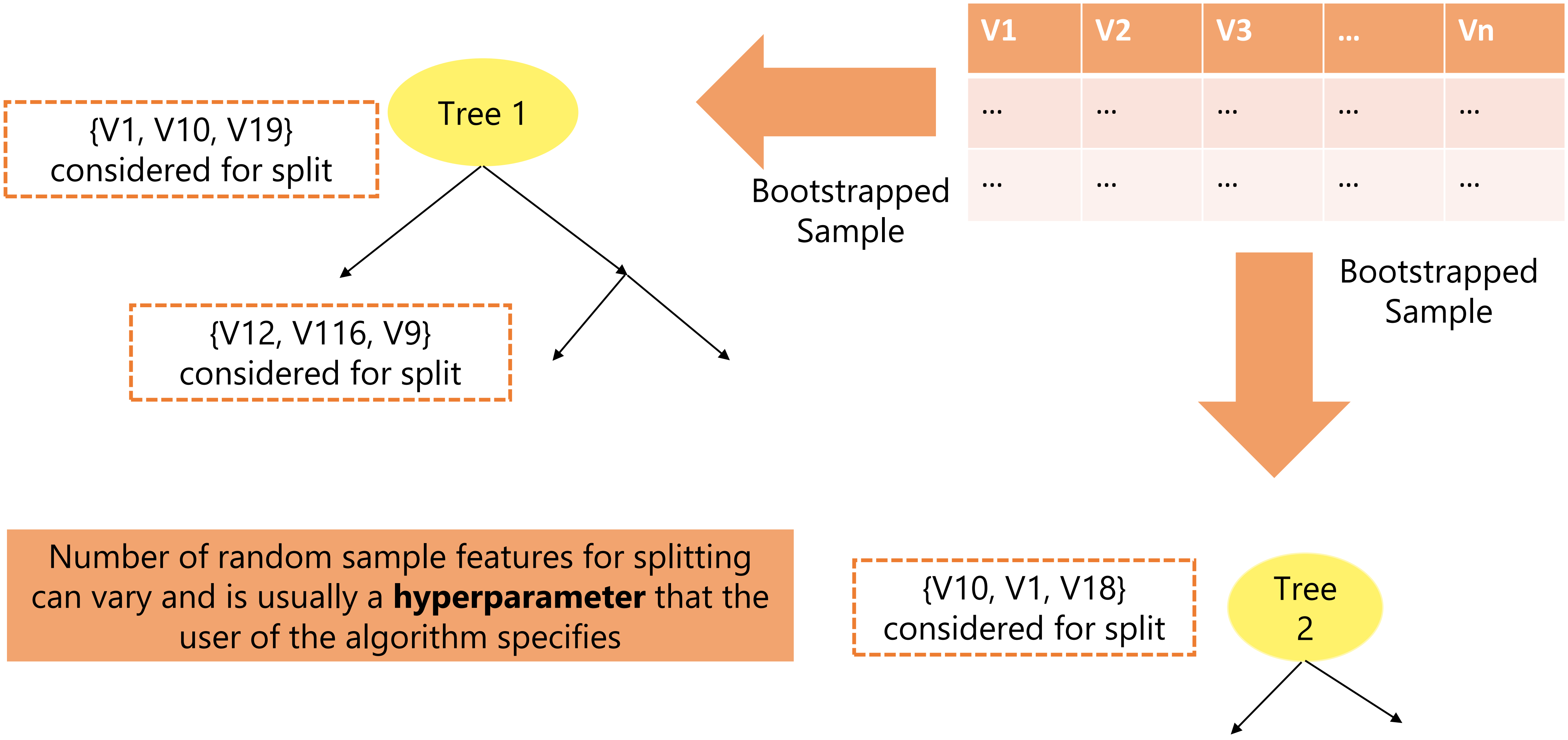
Average Out of Bag observations in Bootstrapped Sampling is around **33%**

Random Forests

Random Forests

- Similar to Bagged Trees
- Difference in resampling data
- For each base leaner (Decision Tree), at each split only a subset of features are considered
- Random bootstrap samples

Random Forests



Random Forests

Random Forests uses tree models as base learner

Extract **Variable Importance** or compute **Out-of-bag Error** to get an estimate of out of sample model performance for parameter tuning

Parameters of Radom Forests

- Number of features considered for each split
- Number of trees
- Number of observations in root node
- Depth of tree models

Recap

- Tree based ensembles overview
- Tree based ensembles models – Bagged Tree and Random Forests
- Bootstrapped sampling
- Variable importance
- Out of bag error (OOB error)
- Random Forests
- Code Demo

MACHINE LEARNING

Algorithms

Class

Tree Based Models

Topic

Tree Based Ensembles: Adaboost and Gradient Boosting

Adaboost

Boosting – Another ensemble technique built using decision trees as base learners

Boosted trees works differently than Bagged trees and Random forests

	Boosted Trees	Bagged Trees	Random Forests
Used to build an ensemble	Data re-weighing strategy	Bootstrapped samples	
Depth of tree	Not large (2 to 3 levels)	As many required	

Adaboost is a popular technique

Data Re-weighing Strategy

Classification task using a data set

Wherever the model makes a mistake, that row is given more importance

X	Y
…	1
…	0
…	1

Tree Model T1

Data re-weighing

X	Y	Y'
…	1	0
…	0	1
…	1	1

Row 1 and 2 are given more weight

Tree Model T2

Data re-weighing

X	Y	Y'
…	1	1
…	0	1
…	1	1

Row 2 is given more weight

Tree Model Tn

Tree models will be shallow

Final model is a combination of these trees (T1, T2,… Tn)

Data Re-weighing Strategy

Classification task using a data set

Wherever the model makes a mistake, that row is given more importance

X	Y
…	1
…	0
…	1

Tree Model T1

Data re-weighing

X	Y	Y′
…	1	0
…	0	1
…	1	1

Row 1 and 2 are given more weight

Tree Model T2

Data re-weighing

X	Y	Y′
…	1	1
…	0	1
…	1	1

Row 2 is given more weight

Tree Model Tn

Re-weighing strategy - each successive tree pays more attention to the parts of the data that preceding trees have failed to correctly predict

Successive trees try to improve the error rate

Gradient Boosting

Gradient Boosting is another popular boosting technique

Gradient boosting is an iterative algorithm

Regression task using a dataset

Yet, the mechanics of the discussions are valid in a classification task

Gradient Boosting

Step 1

Data set with 1 predictor and 1 target variable

X	Y
30	32
48	62
19	23
22	25

T1

Simple tree model

Predictions

X	Y	Y’	Residual
30	32	31	1
48	62	60	2
19	23	24	-1
22	25	26	-1

Error

T2

Error – difference between the actual variable and predicted variable

Step 2

X	Y
30	32
48	62
19	23
22	25

T1+T2

Combination of 2 tree models

X	Y	Y’	Residual
30	32	92.5	0.9
48	62	61	1
19	23	23.5	-0.5
22	25	25.5	-0.5

T3

Gradient Boosting

Keep on repeating this process quite a few times and eventually end up with an ensemble of trees

Gradient boosting is a general ensemble framework

Boosting trees can use other base learner other than decision tree

Partial Dependence Plot

Not as interpretable as simple models like decision trees or linear models

Ensembles provide a list of important predictors by computing variable importance measures

Able to calculate the variables that are important predictors

What is the direction of impact a given predictor has on the dependent variable?

Is the given predictor positively or negatively impacting the dependant variable?

Partial Dependence Plot

Partial Dependence Plot – helps in understanding relationships between a dependent variable and an independent variable

Help in establishing the direction of impact of a predictor on target variable

Depending on which machine learning framework is being used, partial dependence plots for the ensembles may or may not be supported

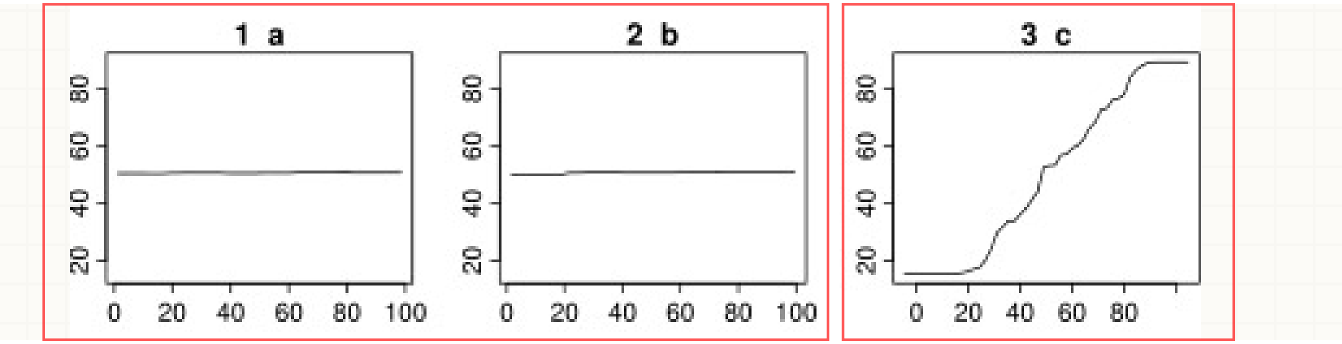
Drawbacks

- Only bivariate relationships can be understood but unearthing interaction effects can be difficult
- Creating partial dependence plot is computationally expensive

Partial Dependence Plot

Partial dependence plots helps identifying the relationship between value of target and the value of a predictor variable after considering the effect of all the other variables

Y axis = values of target variable



3 partial dependence plots

X axis = values of a predictor

Plot 1 and 2 - Value of target variable doesn't change

Plot 3 - Positive relationship between the target variable and the predictor variable

Code Demo

Recap

- Adaboost
- Data re-weighing strategy
- Gradient boosting
- Partial dependence plot
- Code demo