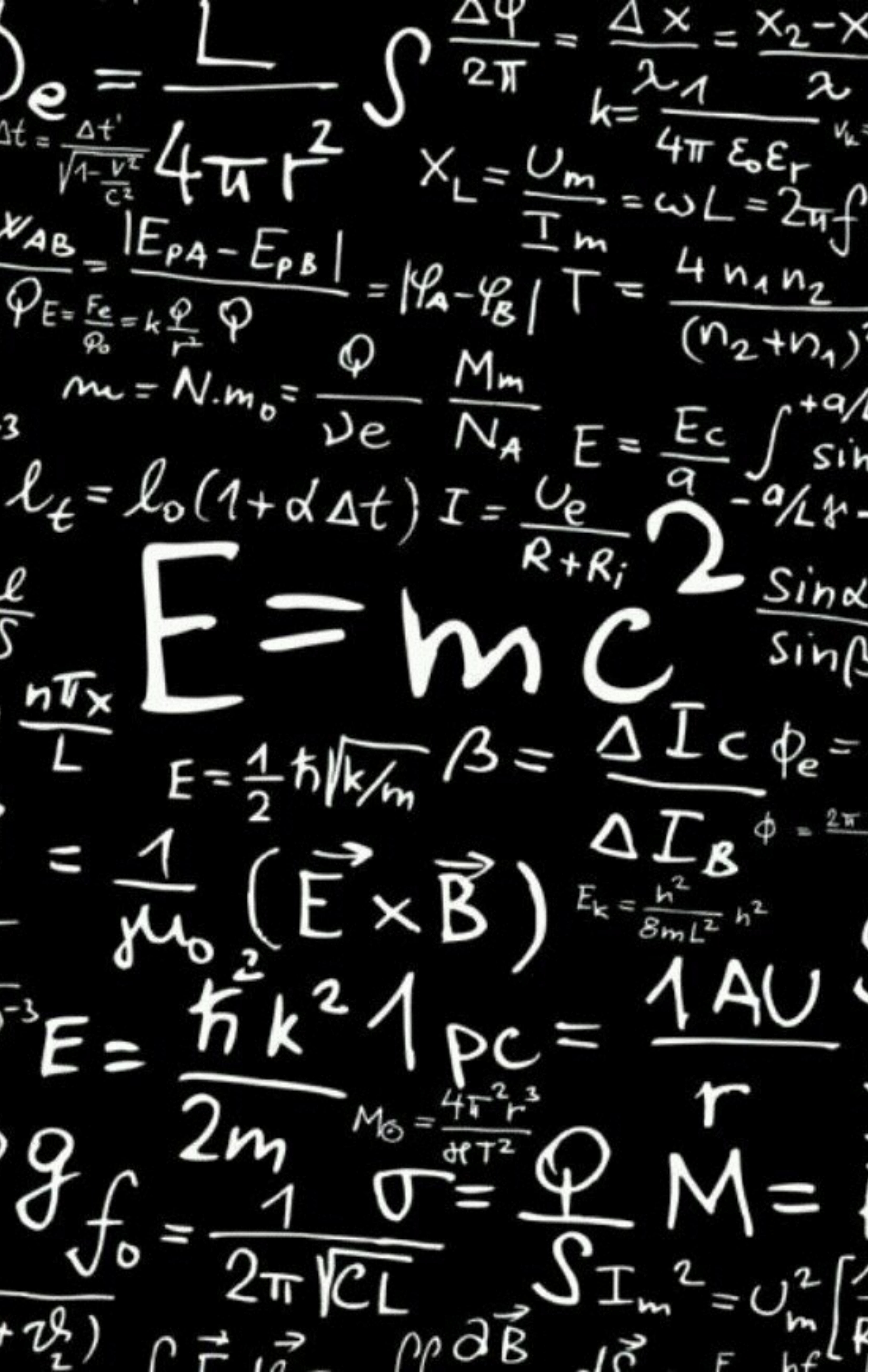


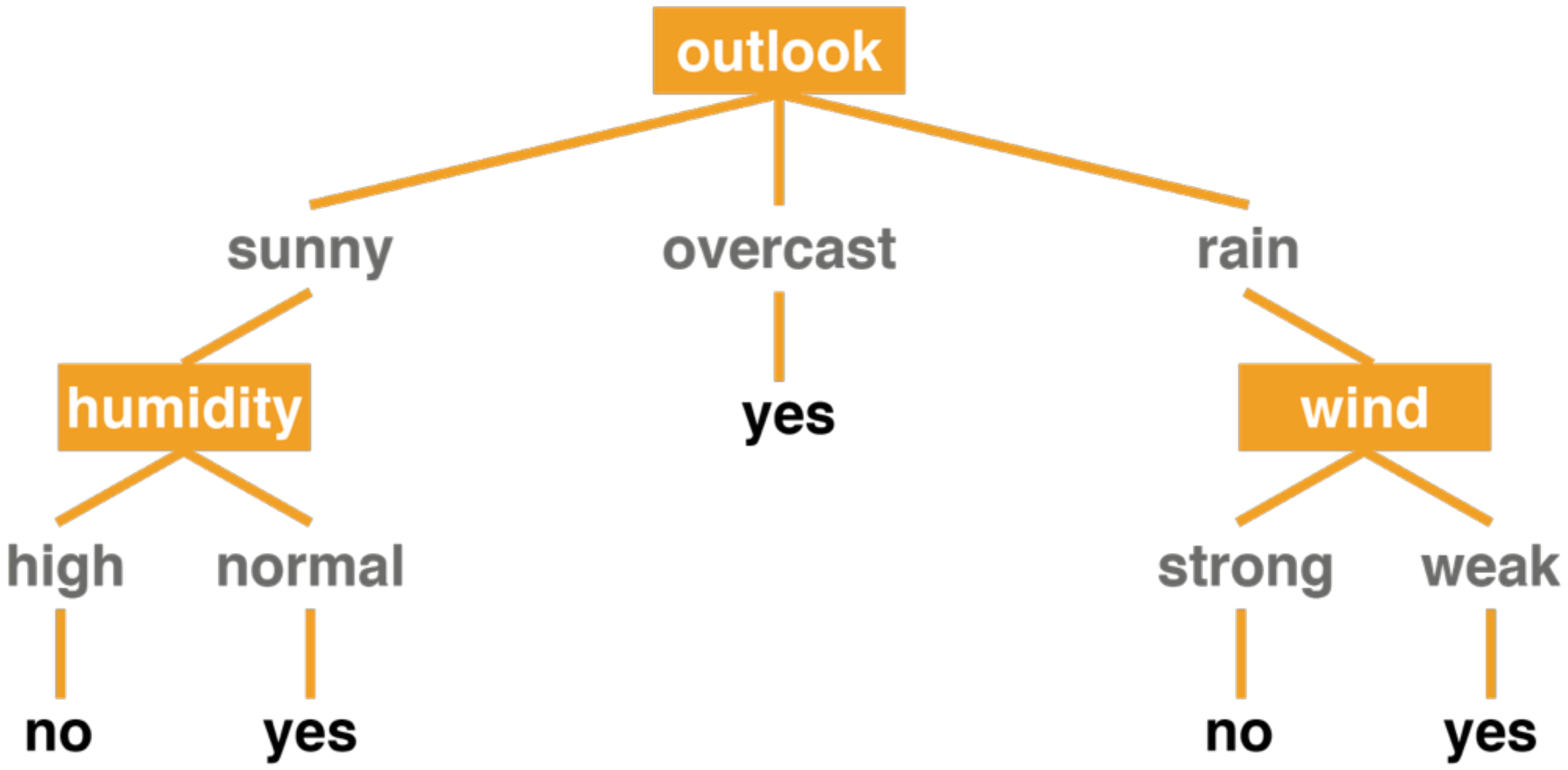
DECISION TREE AND RANDOM FOREST

Decision Tree

- Decision tree is a very old and simple idea. Today, it is perhaps the most popular machine learning algorithm due to the following reasons.
    - 1. Easy to understand
    - 2. Easy to implement (not a lot of parameters to tweak)
    - 3. Can be used for both classification and regression problems

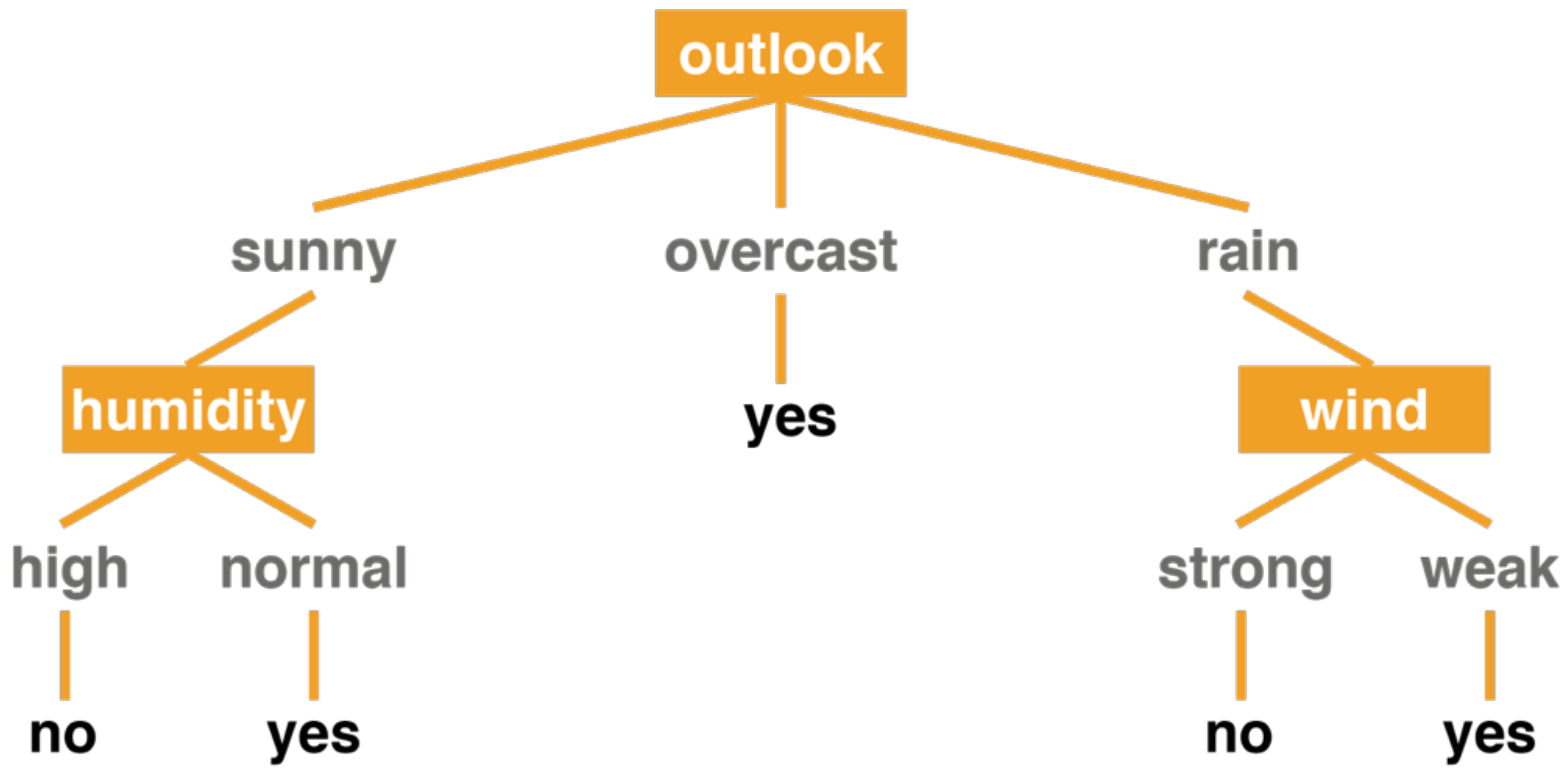
Decision Tree

- Here's an example of a decision tree for classification. Is Josh going to play tennis today?



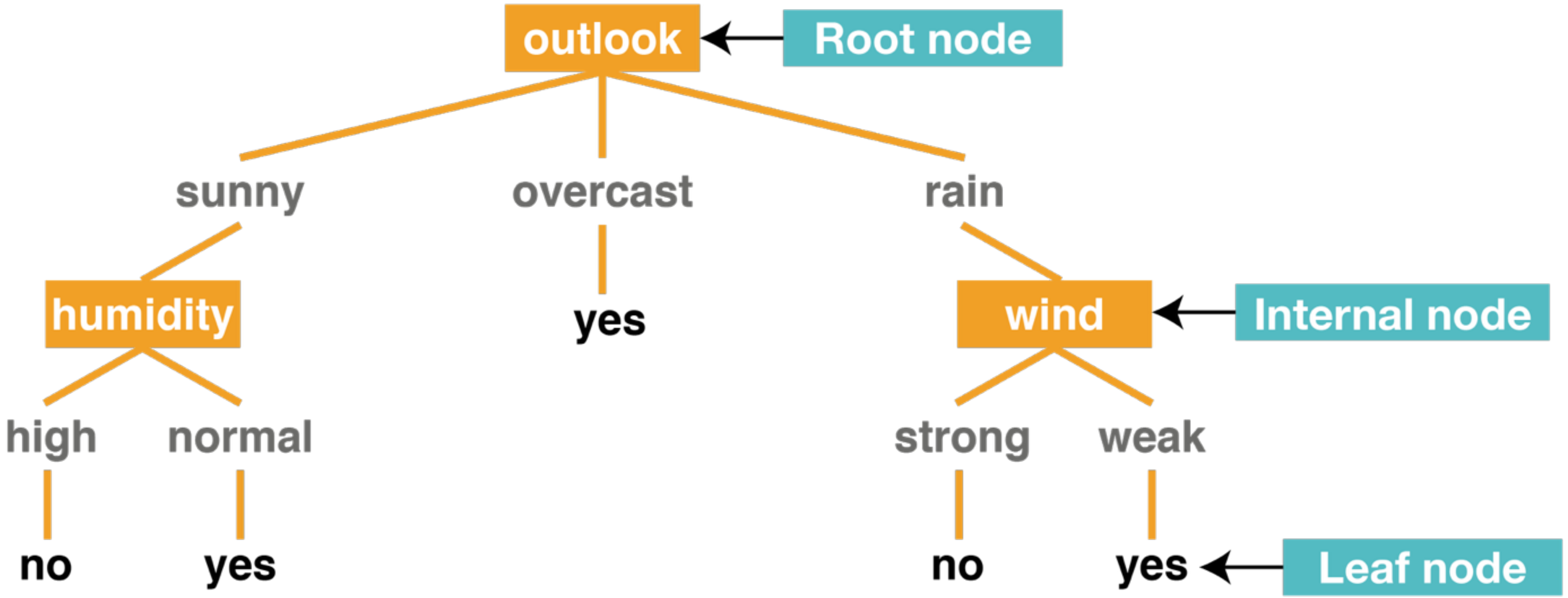
Decision Tree

- The y variable is {yes, no} There are 4 features:
  - x1 = Outlook, {sunny, overcast, rain}
  - x2 = Temperature, {hot, warm, cold}
  - x3 = Humidity, {high, normal}
  - x4 = Wind, {strong, weak}



Decision Tree

Root Node, Internal Node, Leaf Node

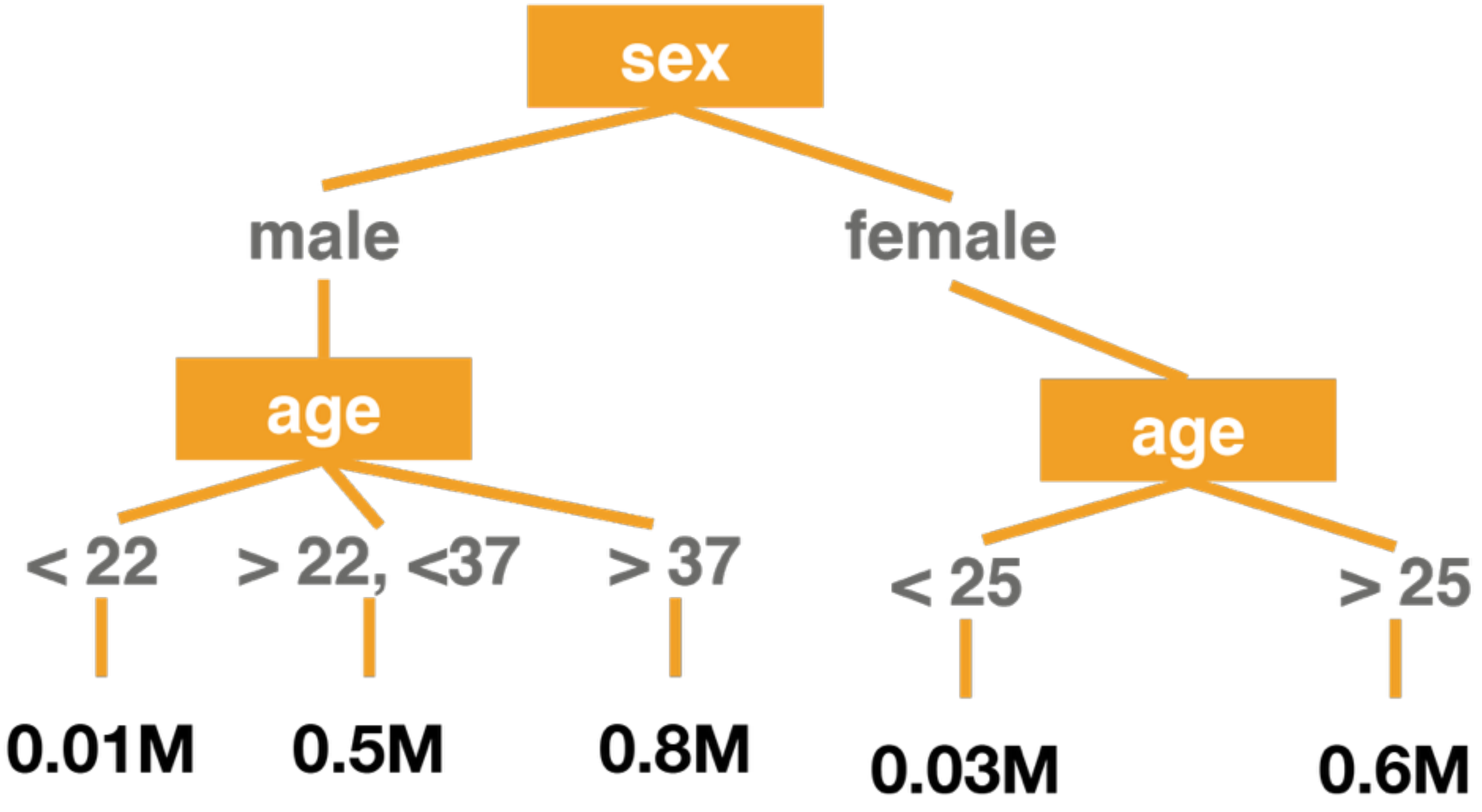


Decision Tree & Continuous Variable

Both x (features) and y (predicted value) can be continuous variables, for example...

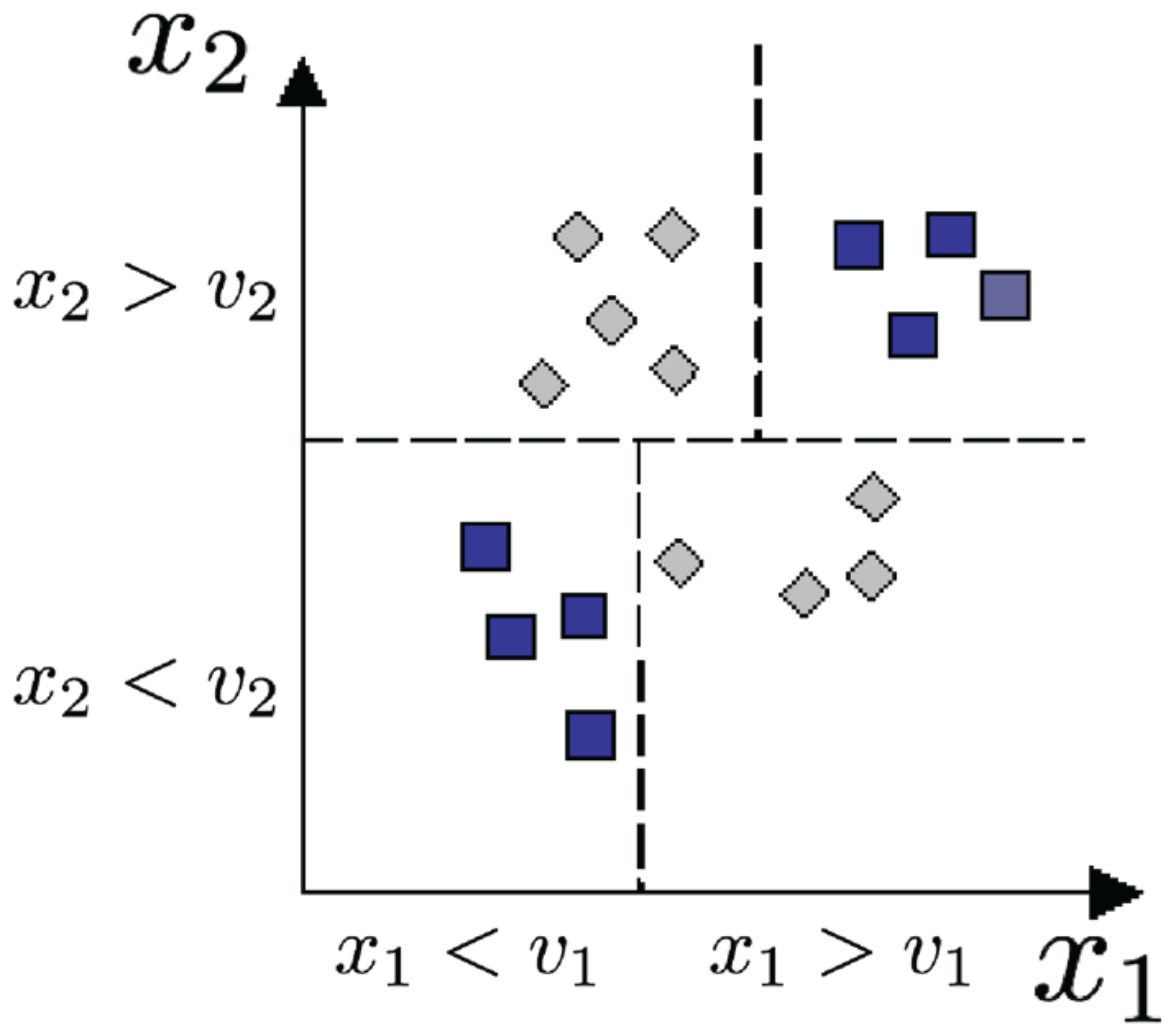
Predict amount of money in saving account of single people (Y) from the following attributes:

- x1 = age (15-65 years old)
- x2 = gender (male or female)



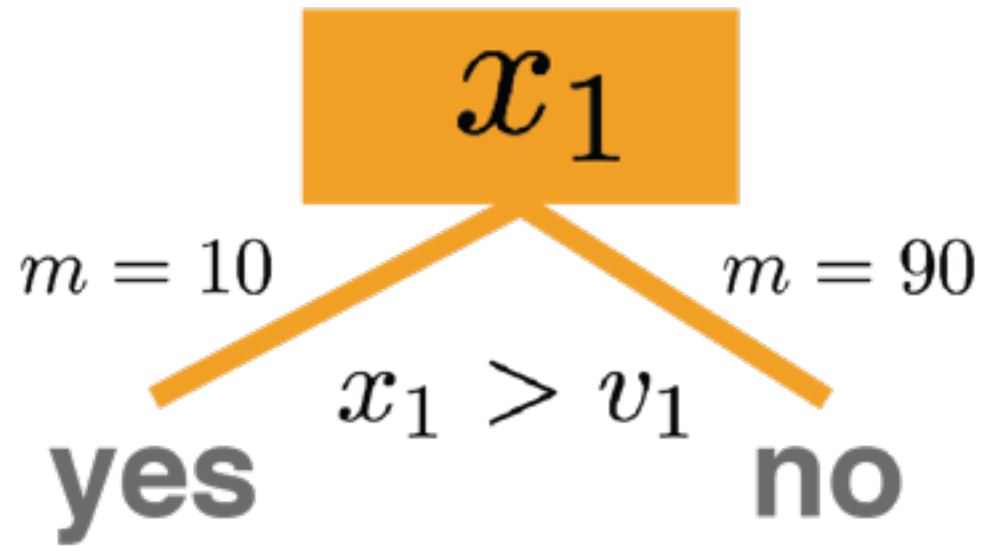
Decision Boundary

Decision trees divide the space into axis-parallel rectangles and label each rectangle with class membership.

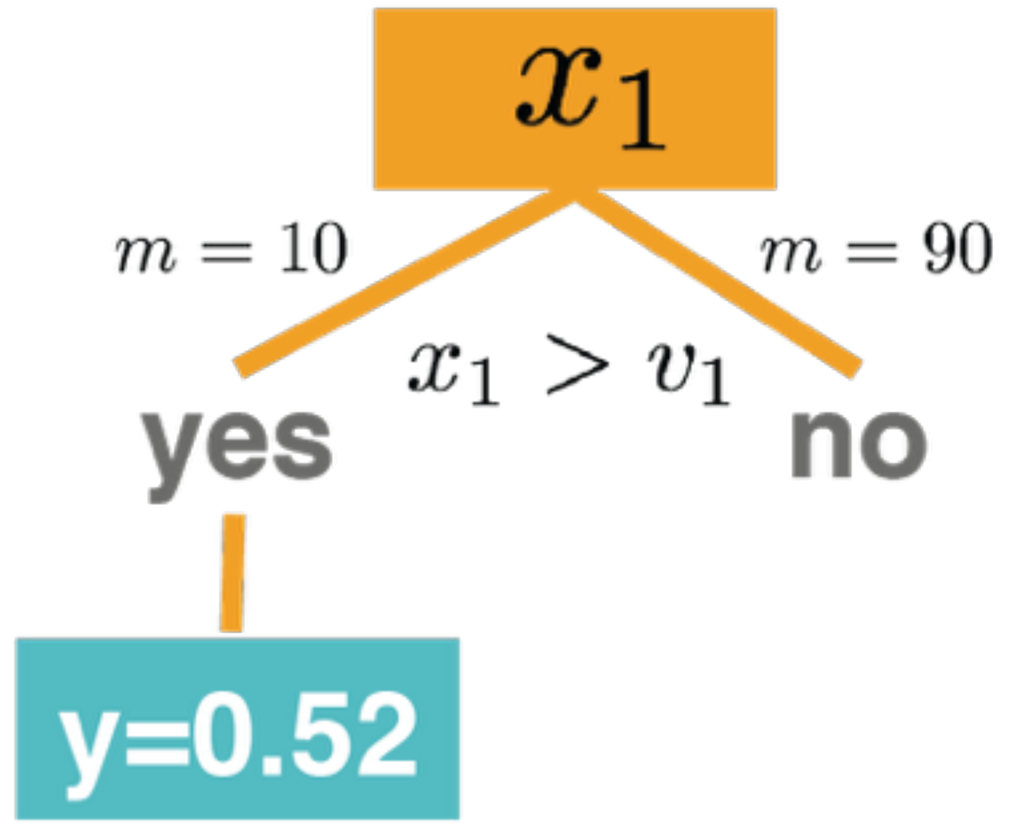


Constructing Decision Tree

One can construct a decision tree by a constructive search. Suppose we have a dataset with m = 100 samples.

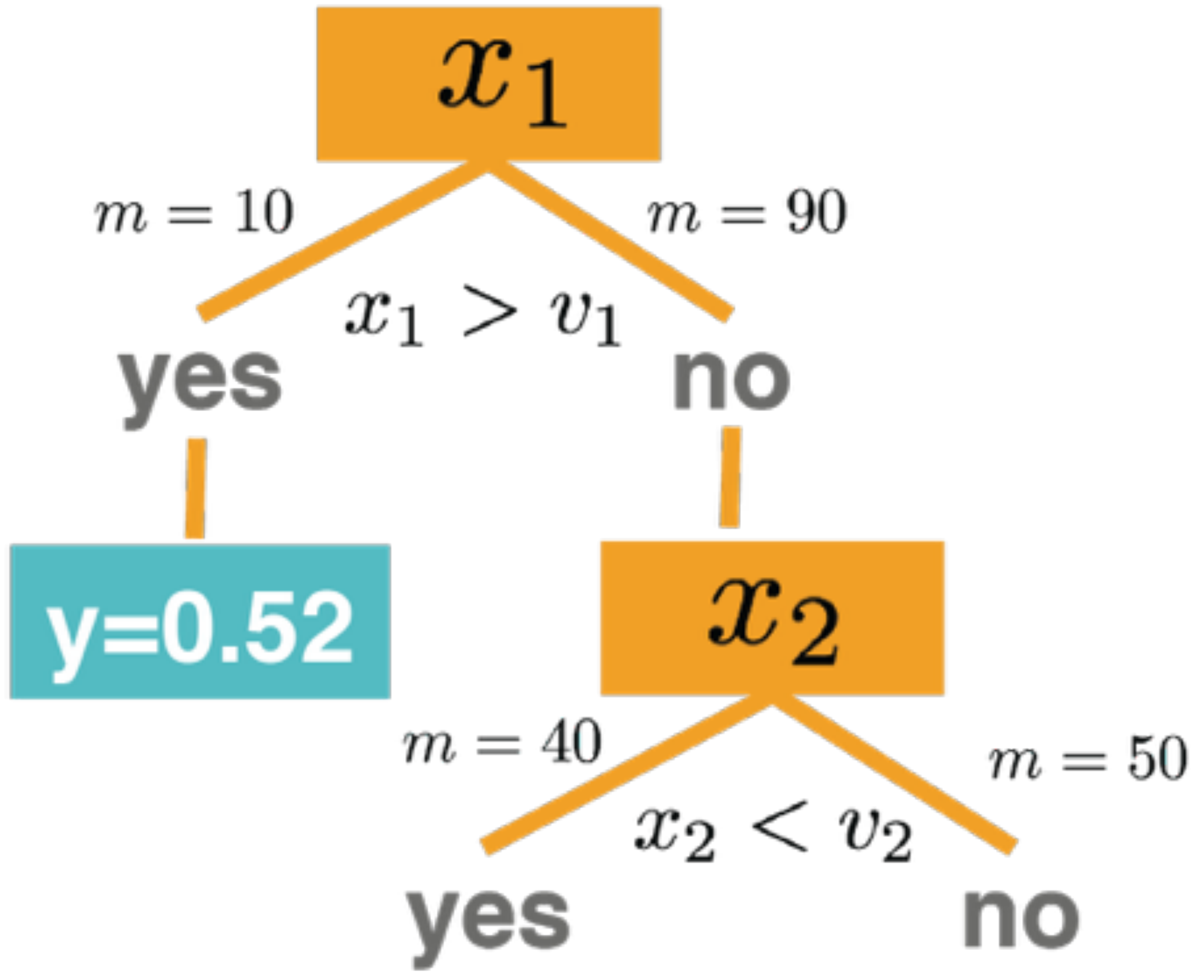


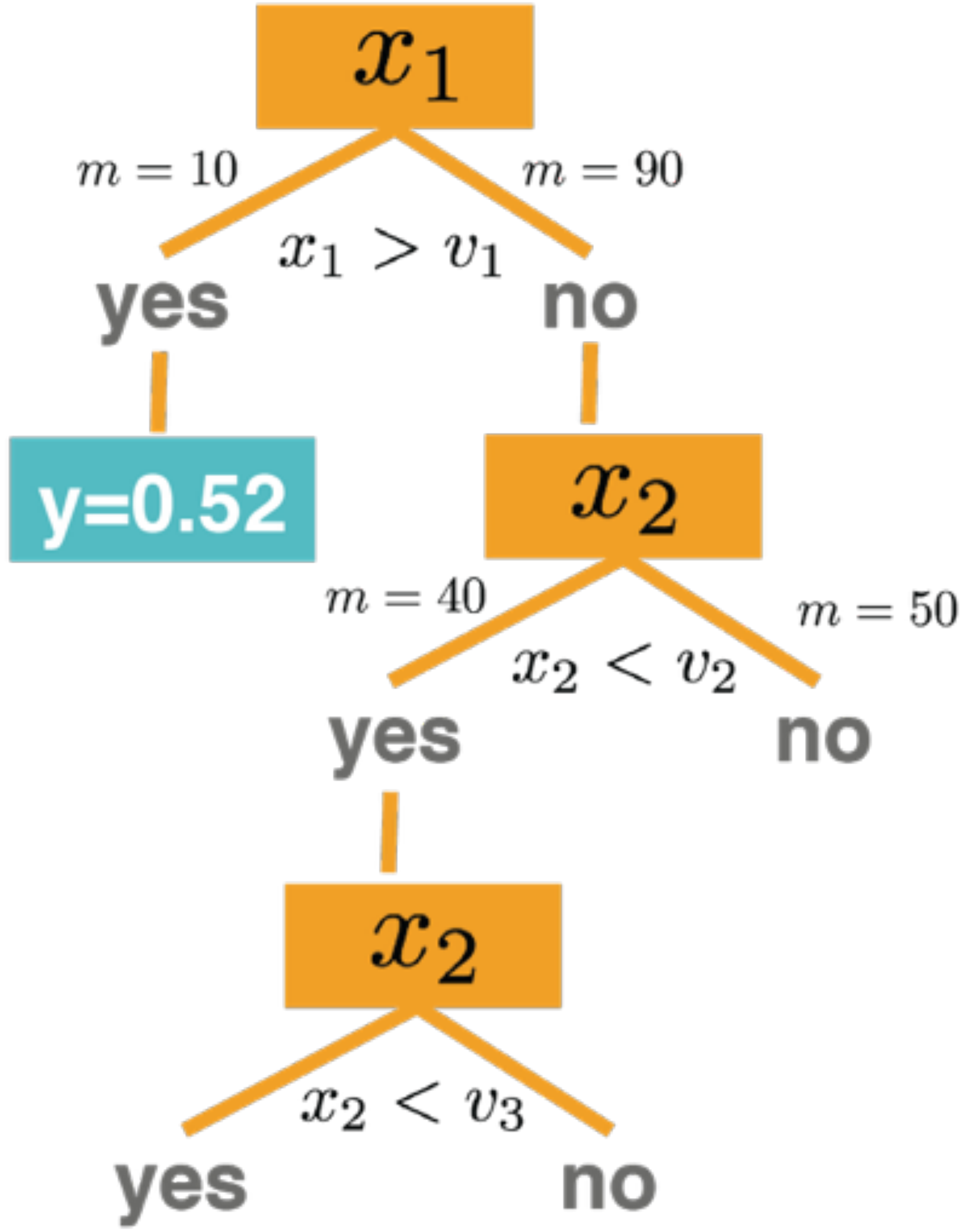
- Decide on the feature at the root node and a boundary to create branches.



- Decide whether to add another internal node. If there's no use in creating another internal node, just add a leaf node and make a prediction.

Constructing Decision Tree





- On the other hand, you may decide to add another internal node which focuses on another feature.
- Repeat the process until we account for all samples.

Pseudo-Algorithm

Function: Build subtree

Require: node n, data at the node D

(n_L, n_R, D_L, D_R) = FindBestSplit(D)

if StoppingCriteria(D_L) **then**
    FindPrediction(D_L)
else
    BuildSubtree(n_L, D_L)
end if

if StoppingCriteria(D_R) **then**
    FindPrediction(D_R)
else
    BuildSubtree(n_R, D_R)
end if

Find the best split

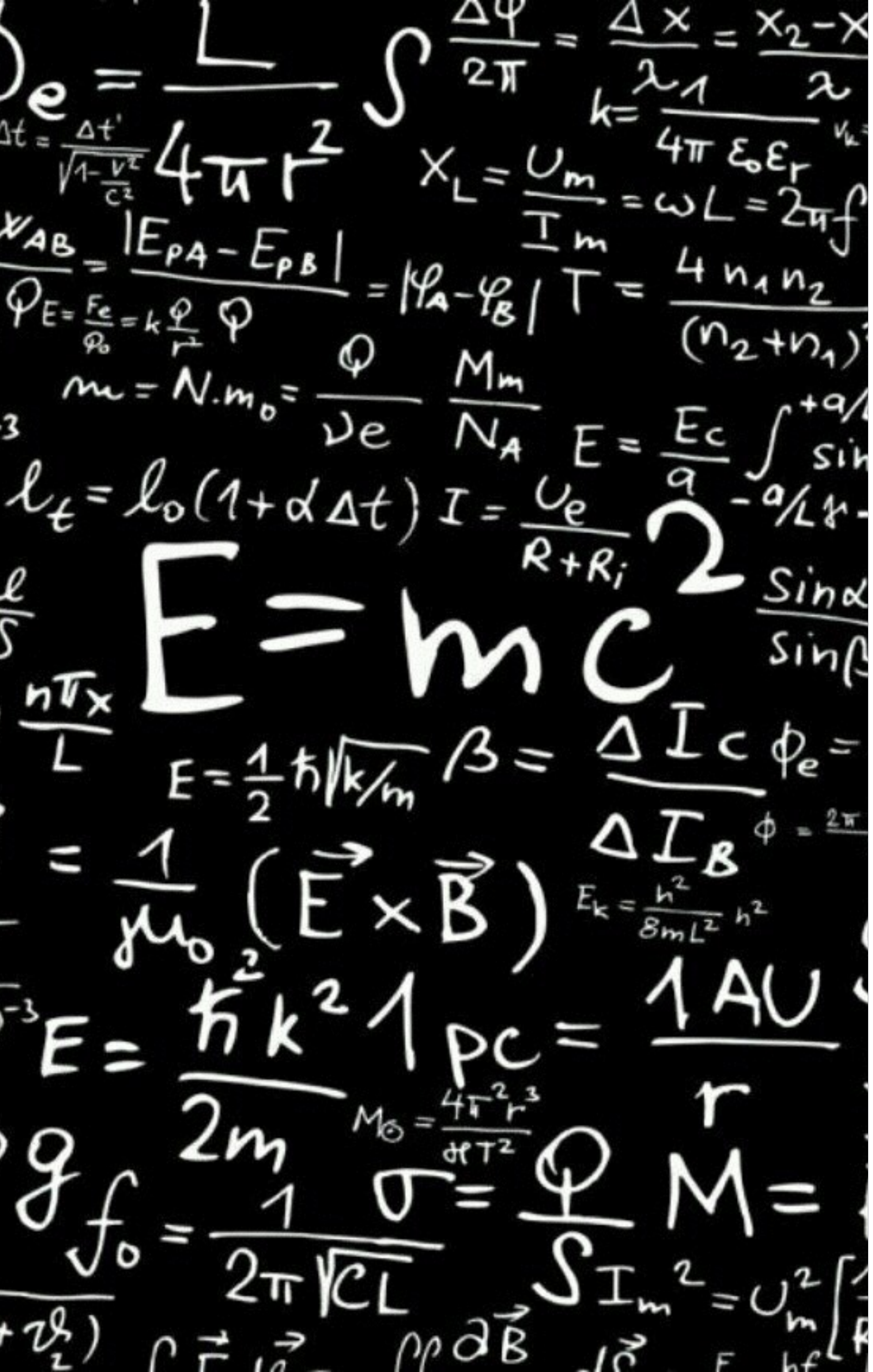
- Intuitively, the best split will send all the 'yes' samples to one side and 'no' samples to the other side.
  - **Choose feature:** first determine which feature gives us the best split. We consider how much information about a given feature x tells us about the value of y (information gain).
  - **Choose threshold:** if that feature is a continuous feature, determine the threshold to split. We consider the threshold that will maximize variance decrease.

Stopping Criteria

- When to decide not to make a branch?
  1. When the leaf is pure, i.e. the variance of Y is small
  2. When the number of samples in the leaf is too small

Making Prediction

- For a classification problem:
  - Predict most common y of the examples in the leaf.
- For a regression problem:
  - Predict the average y of the examples in the leaf or build a linear regression model on the examples in the leaf.



INFORMATION GAIN

Information Theory

- Suppose we have a variable Y which could only be 0 or 1
- Y has the following probability

$$P(Y = 0) = 0.2$$
$$P(Y = 1) = 0.8$$

- We may define 'surprise' variable S, where

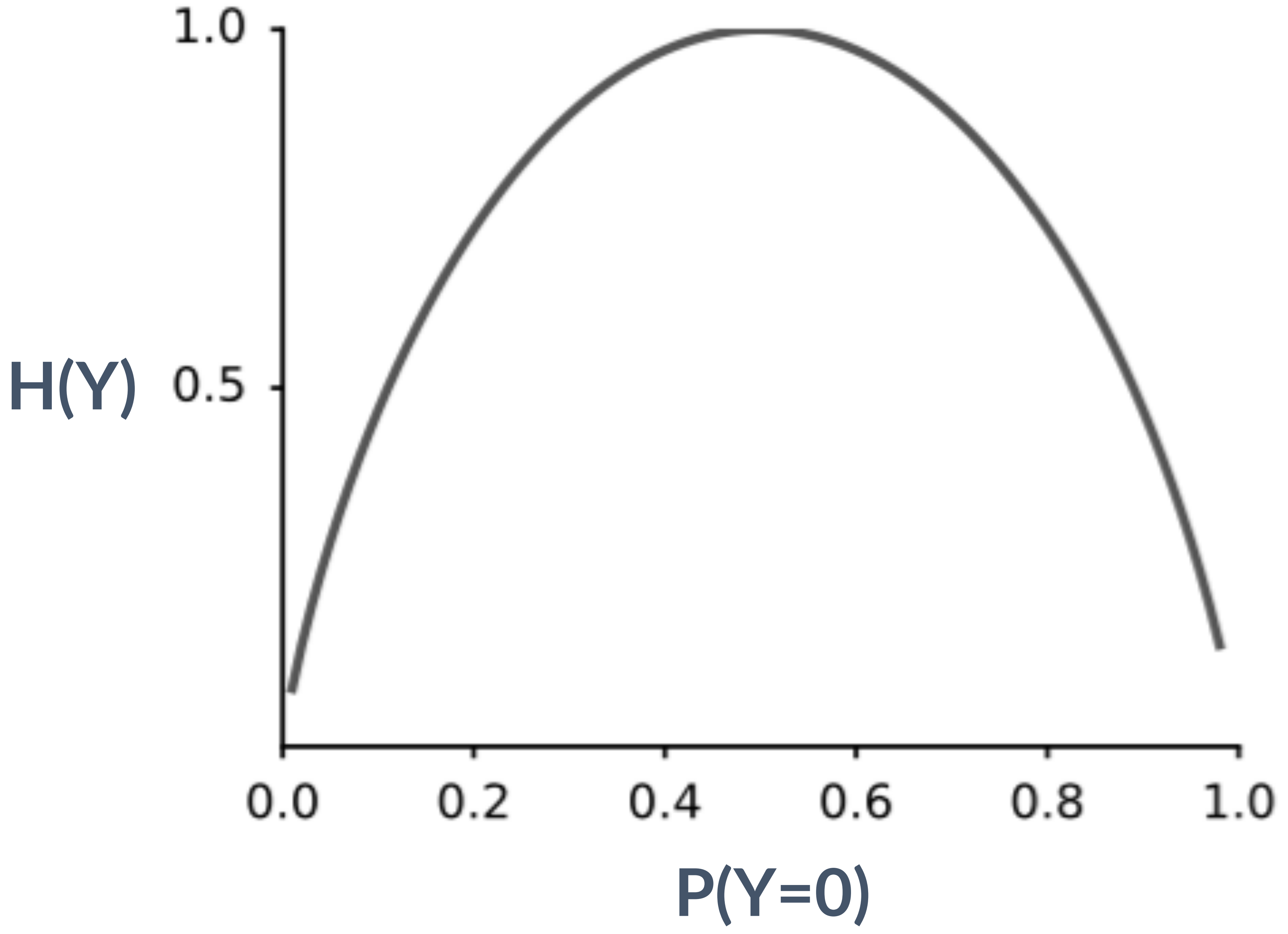
$$S(Y = c) = -log(P(Y = c))$$

- If Y=0, S=2.32 - we are very surprised
- If Y=1, S=0.32 - we are not so surprised.

Entropy

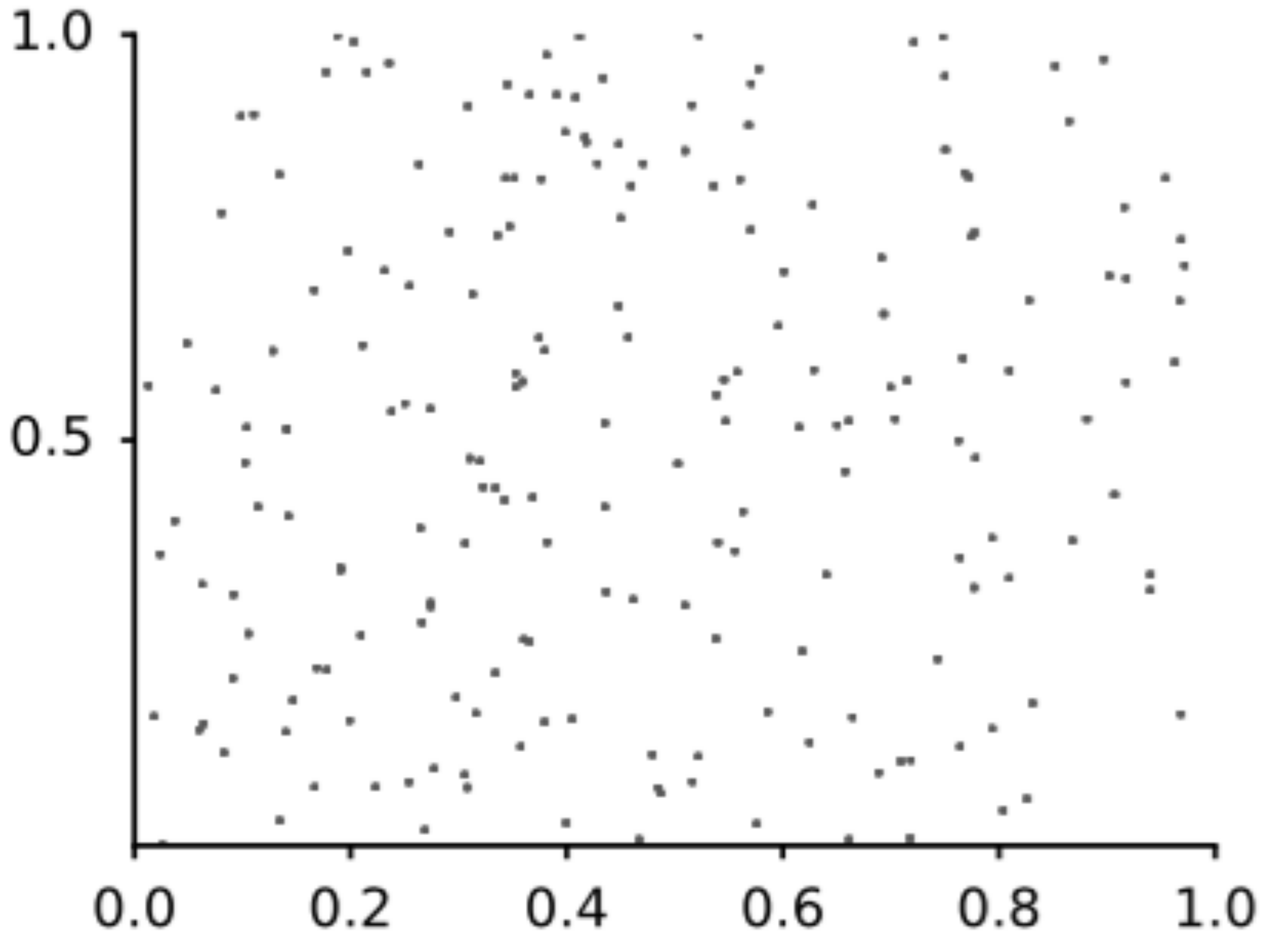
$$H(Y) = \sum_{c=0,1} [-P(Y = c) * log(P(Y = c))]$$

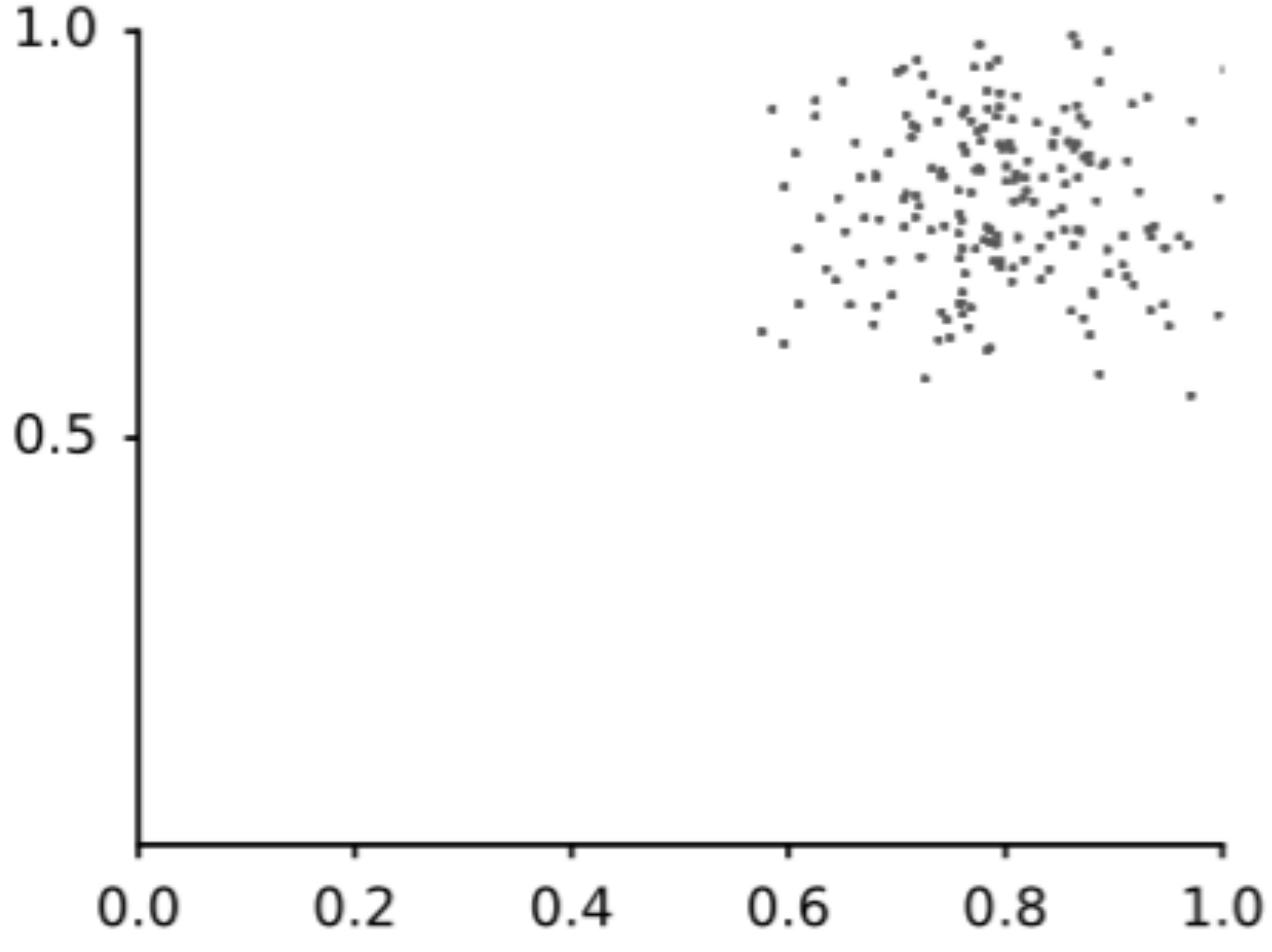
- Entropy is the sum of the product of the surprise and the probability of the outcome.
- This is the average surprise yielded by a single occurrence of Y or the uncertainty of Y.



Entropy

- For continuous variable,
  - 'high entropy': the variable has uniform (boring) distribution. It is hard to guess what the variable is.
  - 'low entropy': the distribution has peaks and valleys, making it easy to guess what the variable is.





Conditional Entropy

- Suppose we want to predict Y from X:

  X = college major

  Y = likes 'Titanic'

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Conditional Entropy

$$P(Y = Yes) = 0.5$$

$$H(Y) = -0.5 \cdot log_2(0.5) - 0.5 \cdot log_2(0.5) = 1$$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Conditional Entropy

Define conditional entropy at $X = v$

$H(Y|X = v)$: the entropy of Y among all records where $X = v$

Example:

$$H(Y|X = Math) = 1$$

$$H(Y|X = History) = 0$$

$$H(Y|X = CS) = 0$$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Conditional Entropy

Conditional entropy $H(Y|X)$ is the average conditional entropy at all values of X:

$$H(Y|X) = \sum_v P(X=v) \cdot H(Y|x=v)$$

Example:

v	$P(X=v)$	$H(Y\|X=v)$
Math	0.5	1
History	0.25	0
CS	0.25	0

$$H(Y|X) = 0.5 * 1 + 0.25 * 0 + 0.25 * 0 = 0.5$$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Information Gain

Information gain is defined as:

$$IG(Y|X) = H(Y) - H(Y|X)$$

The difference between the uncertainty of Y when we don't know X and when we know X.

Example:

$$H(Y) = 1$$

$$H(Y|X) = 0.5$$

$$IG(X|Y) = 0.5$$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Find the best split

- Going through all x, find IG(Y |x), pick the feature with the highest IG to make a split!
- For example, to predict who will live the longest, we compute the following IG
  - IG(LongLife | HairColor) = 0.0001
  - IG(LongLife | Gender) = 0.25
  - IG(LongLife | Smoke) = 0.1
  - IG(LongLife | NSiblings) = 0.005
- Then you'll choose Gender as the feature you'd like to use for the split.

Find the best split

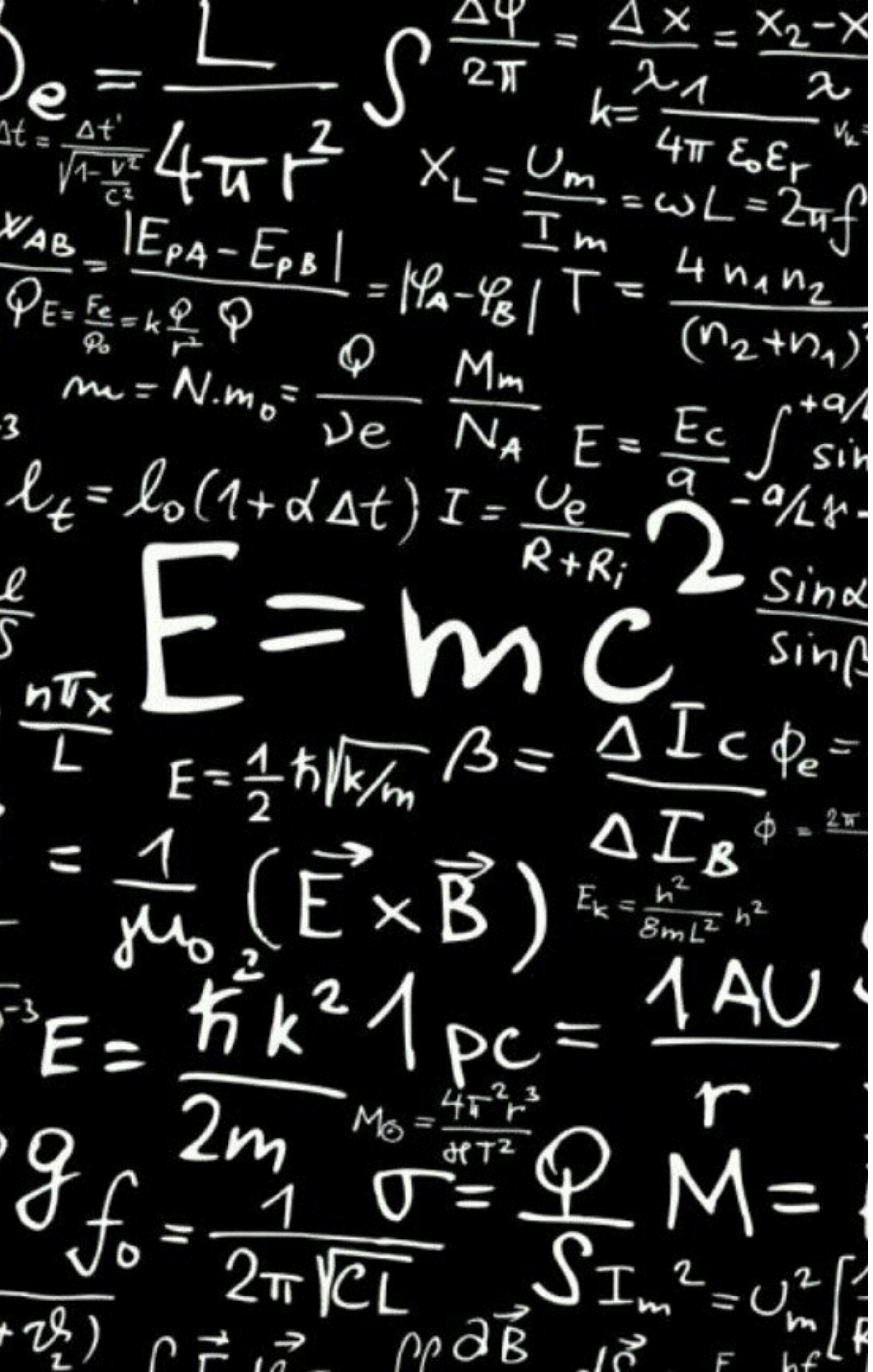
- Choose a threshold: If X is a continuous variable, we must pick the best threshold to split. This is often done by maximizing the following function:

$$m \cdot Var(D) - (m_L \cdot Var(D_L) + m_R \cdot Var(D_R))$$

- This means the variance should decrease most substantially when we split.

Problems with Decision Tree

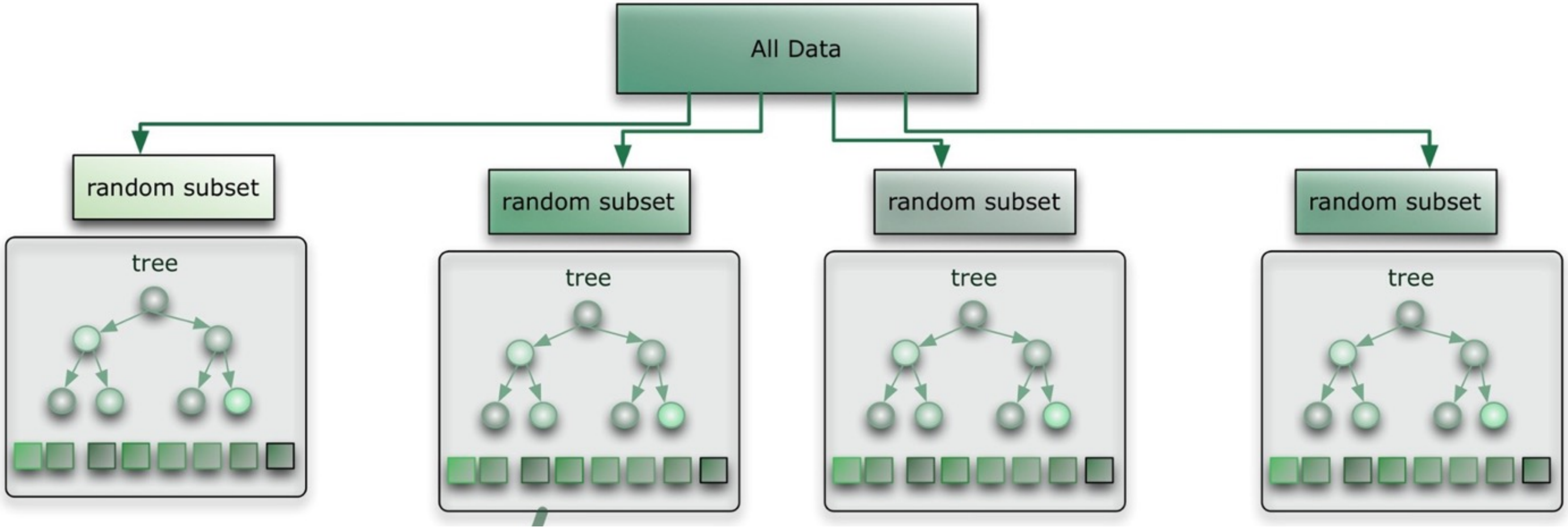
- Decision trees are prone to overfitting.
    - Use early stop criteria
    - Use ensemble method (i.e. random forest)



RANDOM FOREST

Random Forest

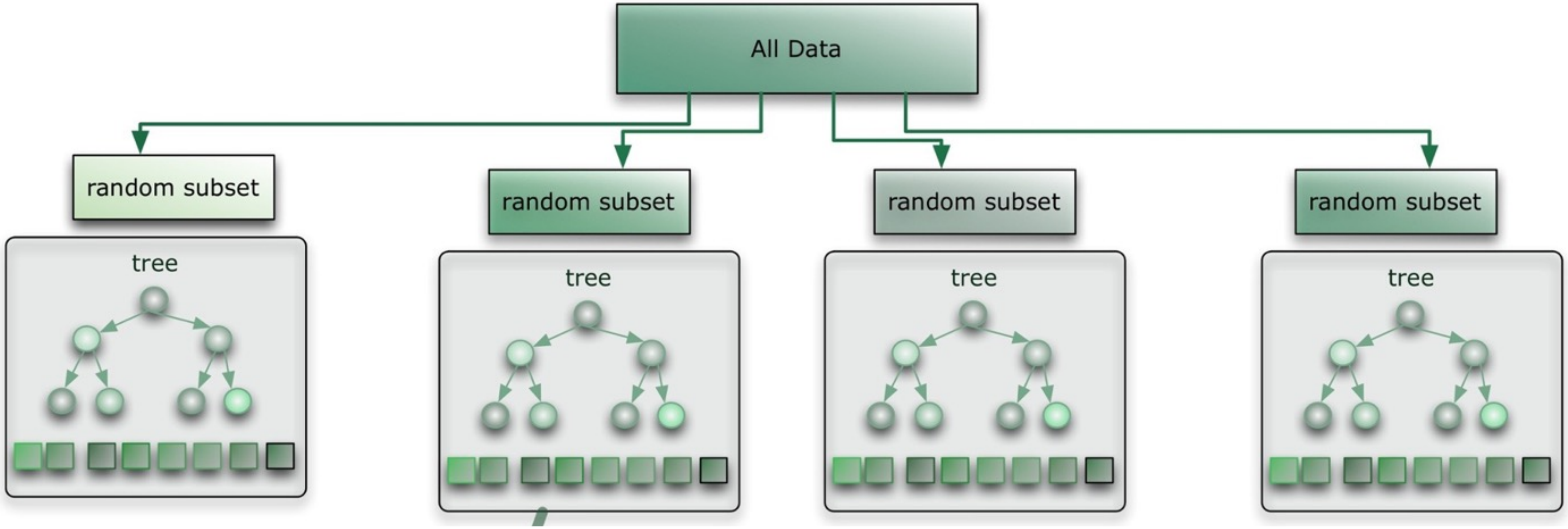
- Random forest is an ensemble. The trees are weak learners and the random forest combines all weak leaners to build a strong learner.
- Number of trees could be 10, 50, 500. The more trees, the less overfitted.



credit: citizennet.com

Random Forest

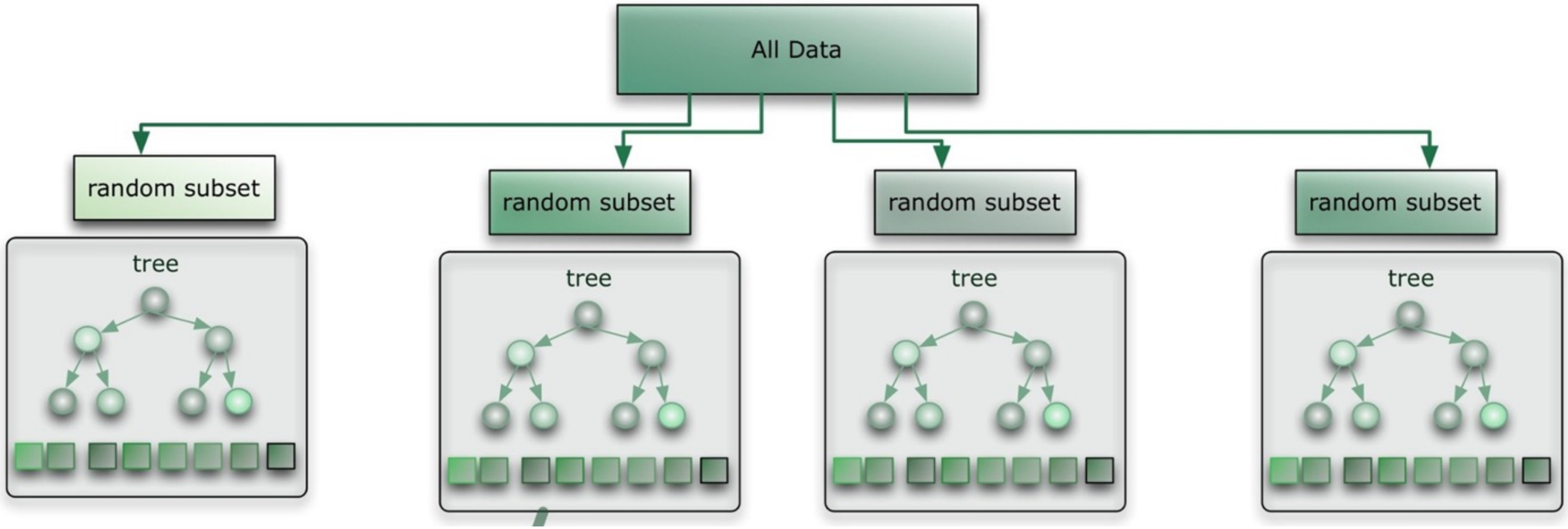
- Each tree will receive subset of features and samples from all data. So that each tree comes up with uncorrelated model (each is equally right in setting up their rules).



credit: citizennet.com

Random Forest

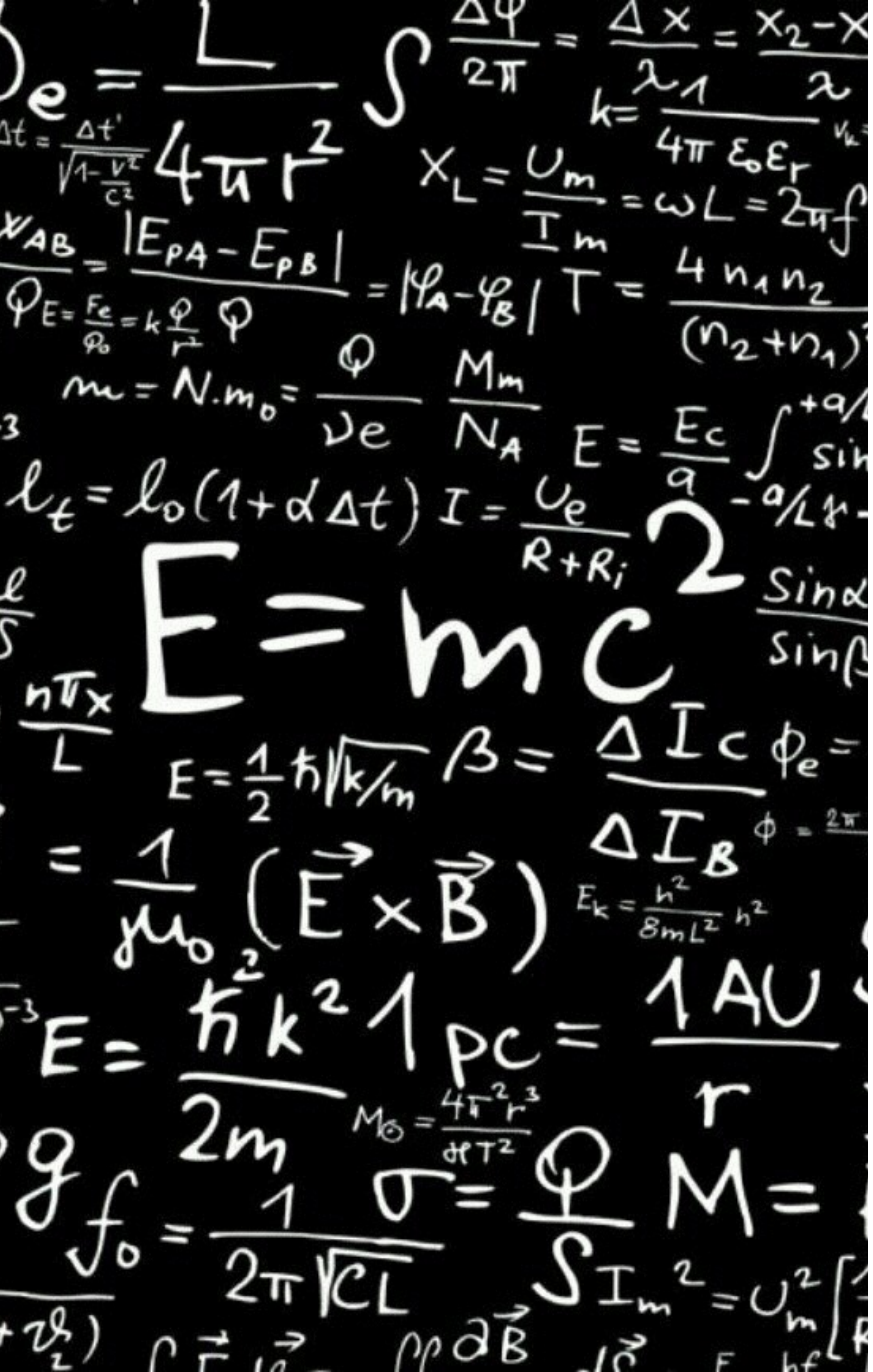
- The answer from all trees will be pooled to create the final answer. We can simply count votes for each categorical option.



credit: citizennet.com

Random Forest

- Random Forest is simple and fast to train. It has no requirement in terms of feature scaling.
- Works well with most tabular data.
- Other variants include Gradient Boosted Tree, Extra Tree, which have additional features that make RF fit data more accurately without overfitting.

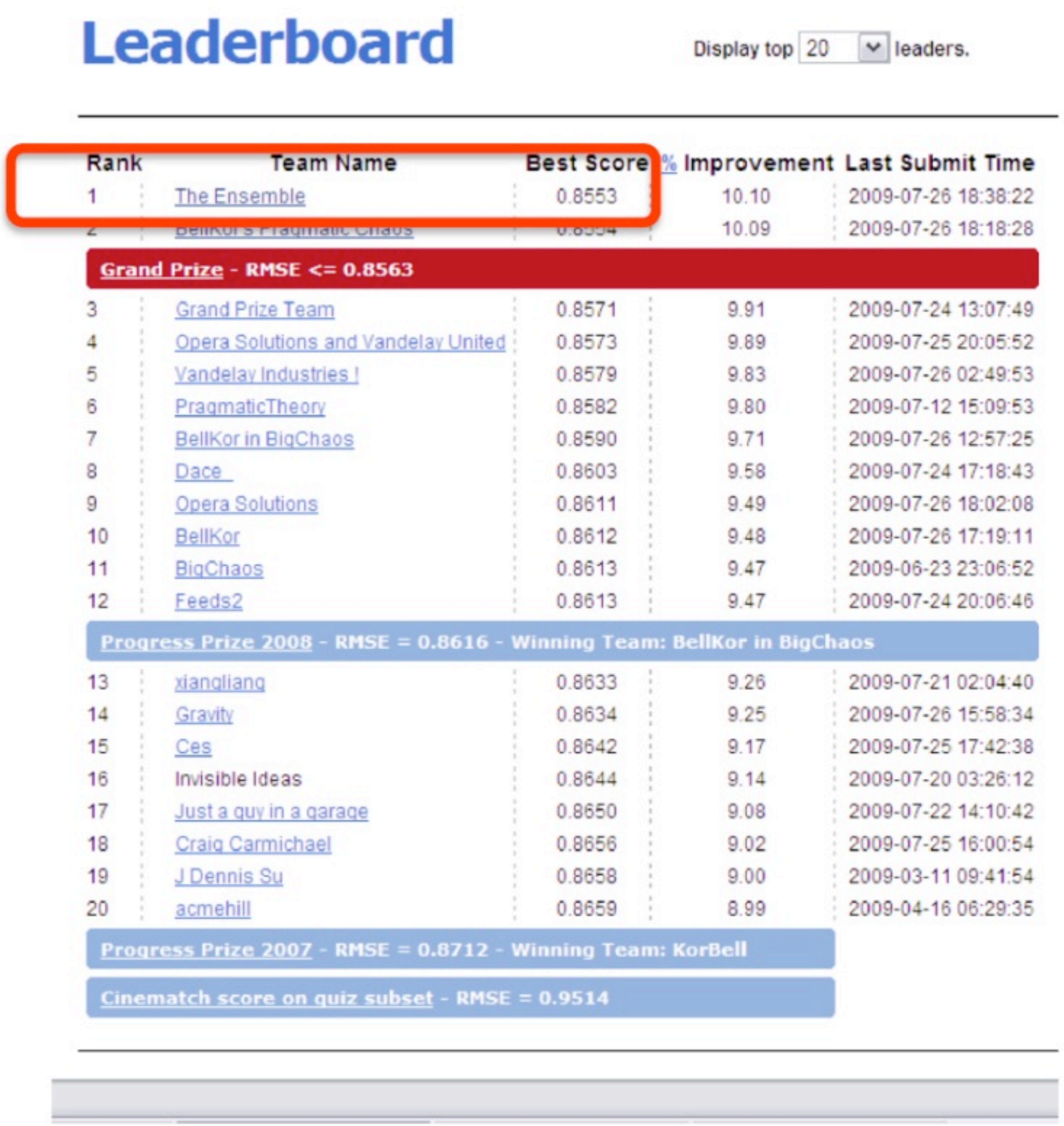


ENSEMBLE LEARNING

Ensemble Learning

- **Ensemble learning**
  - The process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem.
- **Ensemble classification**
  - Aggregation of predictions of multiple classifiers with the goal of improving accuracy.

Machine learning competition with a $1 million prize



Leaderboard

Display top 20 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	The Ensemble	0.8553	10.10	2009-07-26 18:38:22
2	BellKor's Pragmatic Chaos	0.8554	10.09	2009-07-26 18:18:28
Grand Prize - RMSE <= 0.8563				
3	Grand Prize Team	0.8571	9.91	2009-07-24 13:07:49
4	Opera Solutions and Vandelay United	0.8573	9.89	2009-07-25 20:05:52
5	Vandelay Industries !	0.8579	9.83	2009-07-26 02:49:53
6	PragmaticTheory	0.8582	9.80	2009-07-12 15:09:53
7	BellKor in BigChaos	0.8590	9.71	2009-07-26 12:57:25
8	Dace_	0.8603	9.58	2009-07-24 17:18:43
9	Opera Solutions	0.8611	9.49	2009-07-26 18:02:08
10	BellKor	0.8612	9.48	2009-07-26 17:19:11
11	BigChaos	0.8613	9.47	2009-06-23 23:06:52
12	Feeds2	0.8613	9.47	2009-07-24 20:06:46
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8633	9.26	2009-07-21 02:04:40
14	Gravity	0.8634	9.25	2009-07-26 15:58:34
15	Ces	0.8642	9.17	2009-07-25 17:42:38
16	Invisible Ideas	0.8644	9.14	2009-07-20 03:26:12
17	Just a guy in a garage	0.8650	9.08	2009-07-22 14:10:42
18	Craig Carmichael	0.8656	9.02	2009-07-25 16:00:54
19	J Dennis Su	0.8658	9.00	2009-03-11 09:41:54
20	acmehill	0.8659	8.99	2009-04-16 06:29:35
Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell				
Cinematch score on quiz subset - RMSE = 0.9514				

			users		
1		?	3	5	?
?	1				2
	4		4	5	?

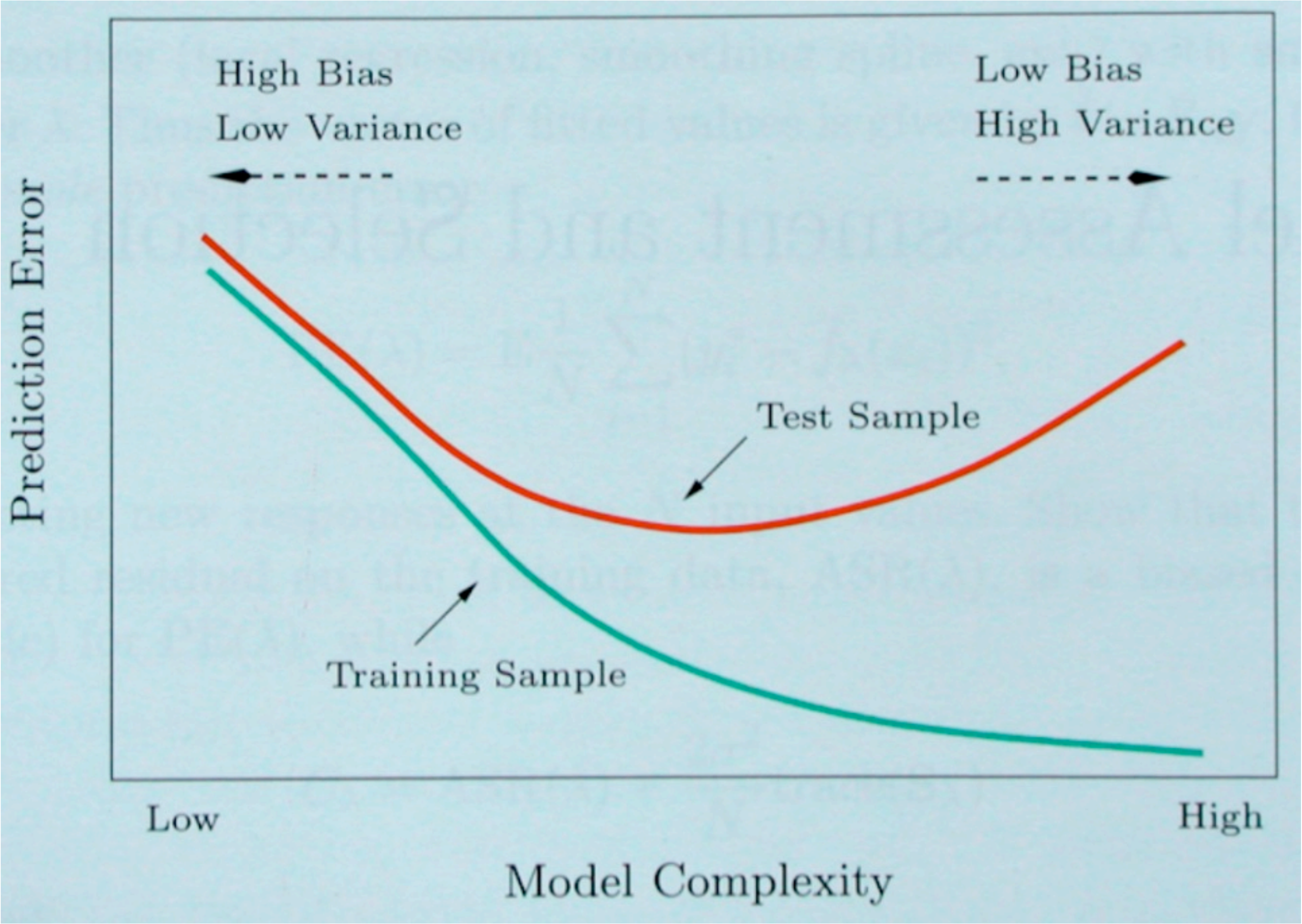
movies

Slide of Dr. Santitham Prom-On

Ensemble Learning

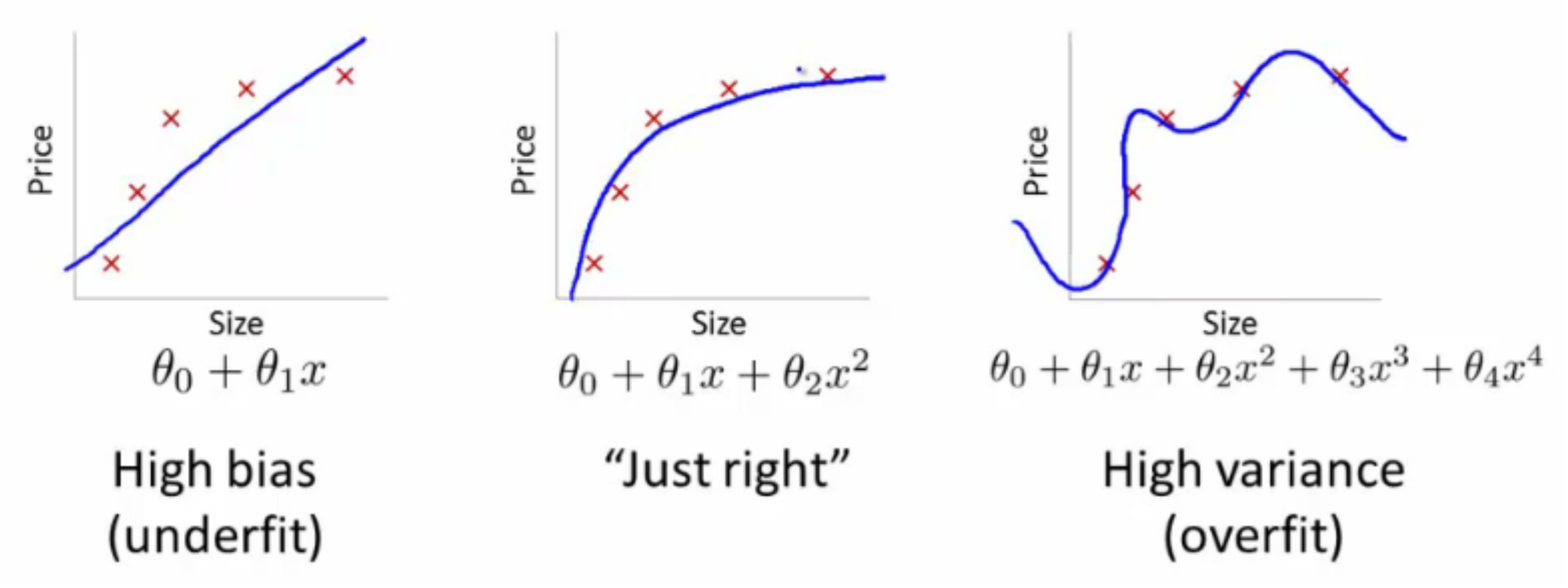
- Utility of combining diverse, independent opinions in human decision-making
- Majority vote
  - Suppose we have 5 completely independent classifiers…
  - If accuracy is 70% for each
    - 10 (.7^3)(.3^2) + 5 (.7^4)(.3) + (.7^3)
    - 83.7% accuracy
  - 101 such classifiers
    - 99.9% accuracy

Bias-Variance Tradeoff



Hastie, Tibshirani, Friedman "Elements of Statistical Learning" 2001

Bias-Variance Tradeoff



Model Complexity

Slide of Dr. Santitham Prom-On

Reducing variance without increasing bias

- Averaging reduces variance:

$$\operatorname{var}\left(\bar{X}\right) = \frac{\operatorname{var}\left(X\right)}{n}$$

  - Average models to reduce model variance
  - One problem
    - Only one training set
    - Where do multiple models come from?

Bagging: bootstrap aggregation

- Take repeated bootstrap samples from training set D.
  - Bootstrap sampling: Given set D containing N training examples, create D′ by drawing N examples at random with replacement from D.

- Bagging:
  - Create k bootstrap samples D1 … Dk
  - Train distinct classifier on each Di
  - Classify new instance by majority vote / average

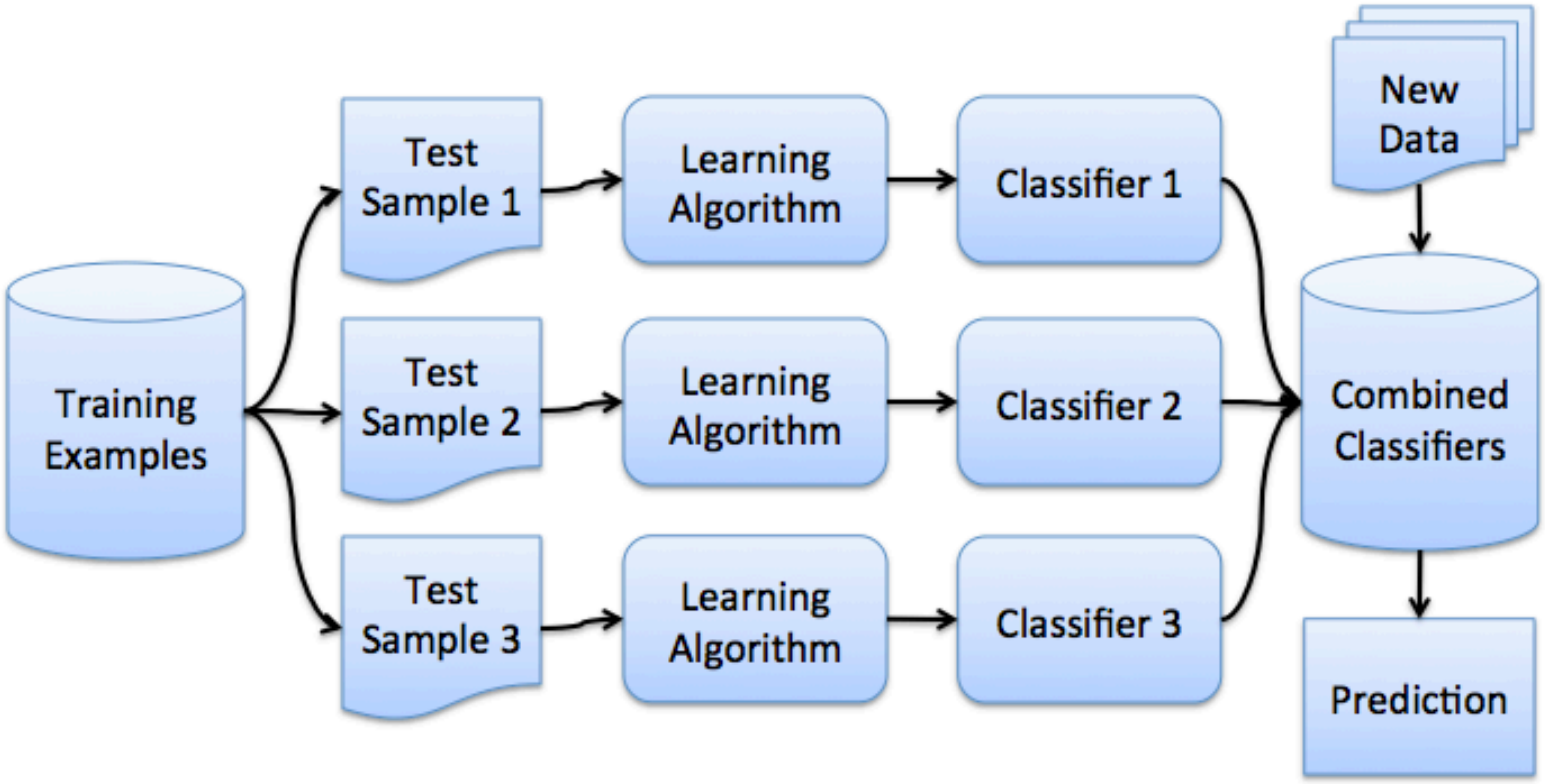
Bagging

- Best case

$$\operatorname{var}\left(bagging\left(L\left(x,D\right)\right)\right) \rightarrow \frac{\operatorname{var}\left(L\left(x,D\right)\right)}{N}$$

- In practice, models are correlated, so the reduction is smaller than 1/N
- Variance of the models trained on fewer training cases usually somewhat larger

Bagging



Training Examples
Test Sample 1
Test Sample 2
Test Sample 3
Learning Algorithm
Learning Algorithm
Learning Algorithm
Classifier 1
Classifier 2
Classifier 3
New Data
Combined Classifiers
Prediction

Reduce bias and decrease variance?

- Bagging reduces variance by averaging
- In practice, bagging has little effect on bias
- Can we average and reduce bias?
- Yes: Boosting

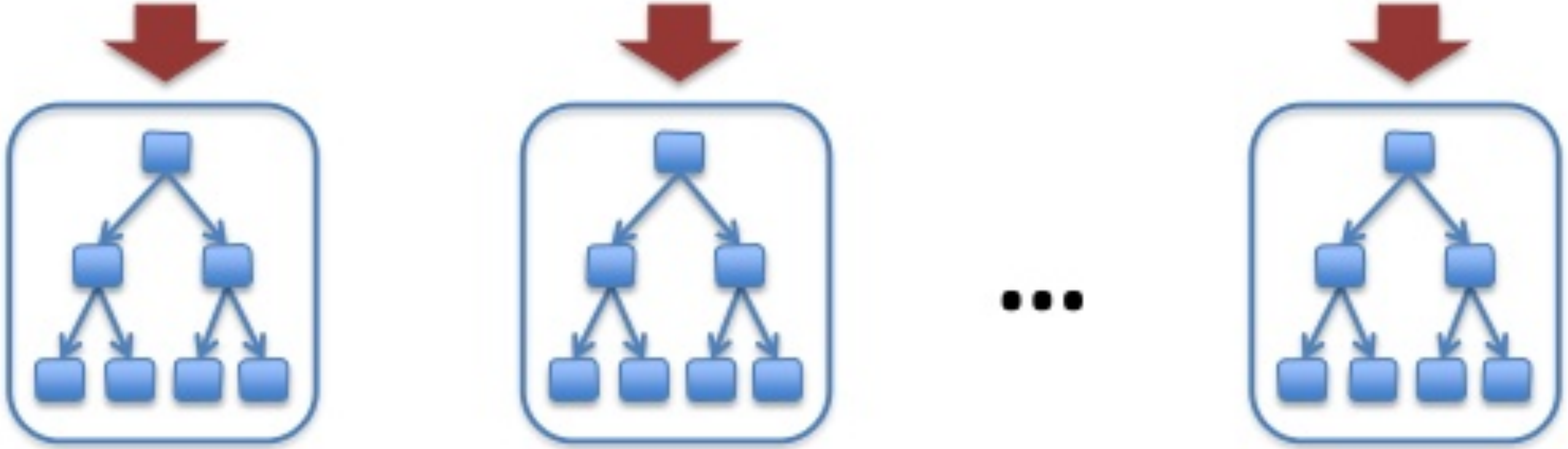
Boosting

- Boosting aims to reduce bias.
- Can a set of weak learners create a single strong learner?
- A weak learner is defined to be a classifier which is only slightly correlated with the true classification (it can label examples better than random guessing).
- In contrast, a strong learner is a classifier that is arbitrarily well-correlated with the true classification.
- It make examples currently misclassified more important (or less, in some cases)

Boosting (AdaBoost)

$$h(x) = a_1h_1(x) + a_2h_2(x) + \ldots + a_3h_n(x)$$

$S' = \{(x,y,u_1)\}$ $S' = \{(x,y,u_2)\}$ $S' = \{(x,y,u_3))\}$



$h_1(x)$ $h_2(x)$ $h_n(x)$

u – weighting on data points
a – weight of linear combination

https://www.cs.princeton.edu/~schapire/papers/explaining-adaboost.pdf

Stop when validation performance plateaus (will discuss later)

Boosting

- Create a sequence of classifiers, giving higher influence to more accurate classifiers
- At each iteration, make examples currently misclassified more important (get larger weight in the construction of the next classifier)
- Then, combine classifiers by weighted vote (weight given by classifier accuracy)

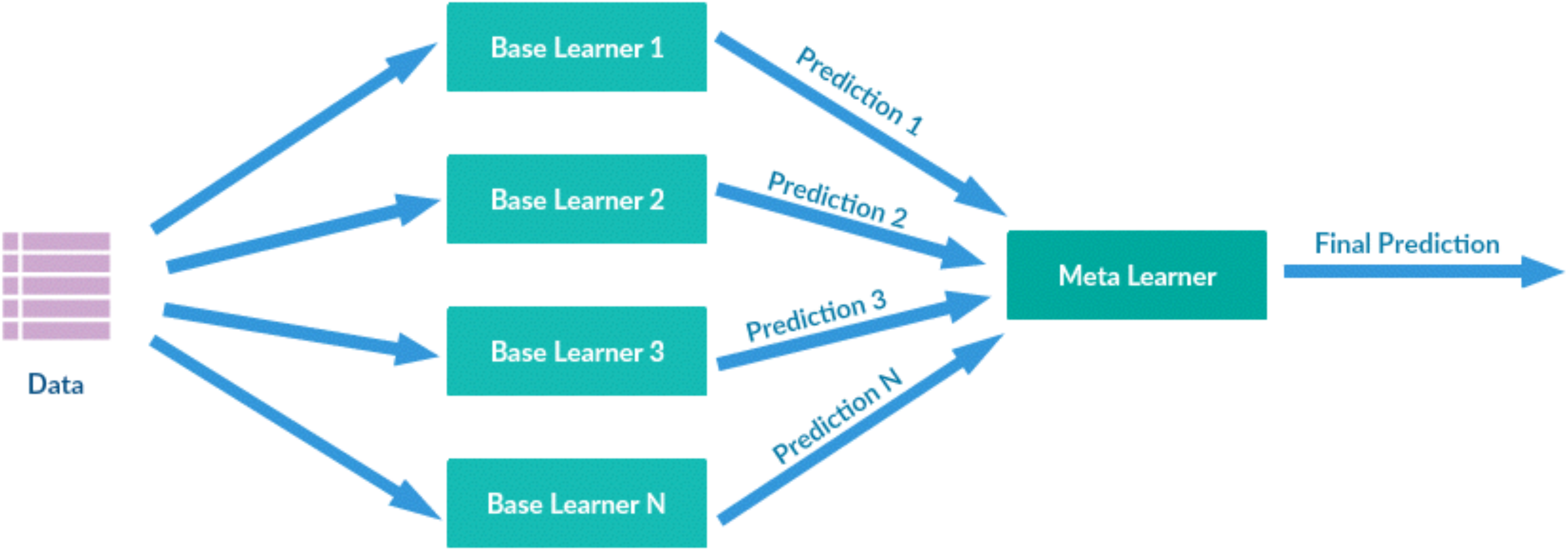
AdaBoost

- Advantages
  - Very little code
  - Reduce variance
- Disadvantages
  - Sensitive to noise and outliers

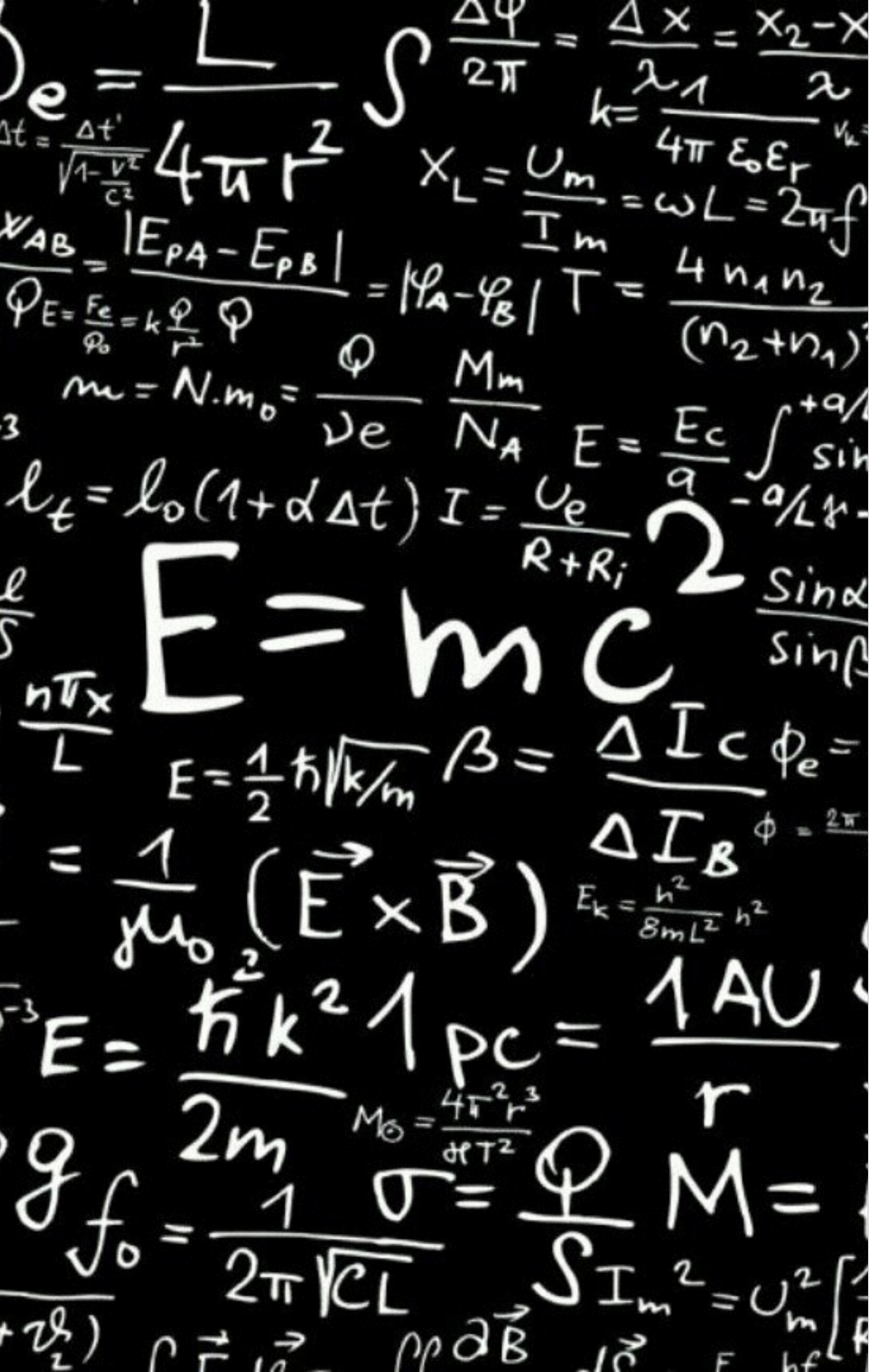
Stacking

- Stacking (sometimes called stacked generalization) involves training a learning algorithm to combine the predictions of several other learning algorithms
- First, all of the other algorithms are trained using the available data, then a combiner algorithm is trained to make a final prediction using all the predictions of the other algorithms as additional inputs

Stacking



credit:supunsetunga.blogspot.com



RANDOM FOREST LAB