CSE 416 Section 4!

~~Tip of the day:~~

Prepend # to every line of your code increases the efficiency of your code and optimizes your code to have O(0) runtime!

July 14, 2020

HONGJUN JACK WU 😆

Me: *uses machine learning*

Machine: *learns*

Me:

THANKS
FOR FILLING OUT THE EVALUATION!

Notebook wise, what we heard:

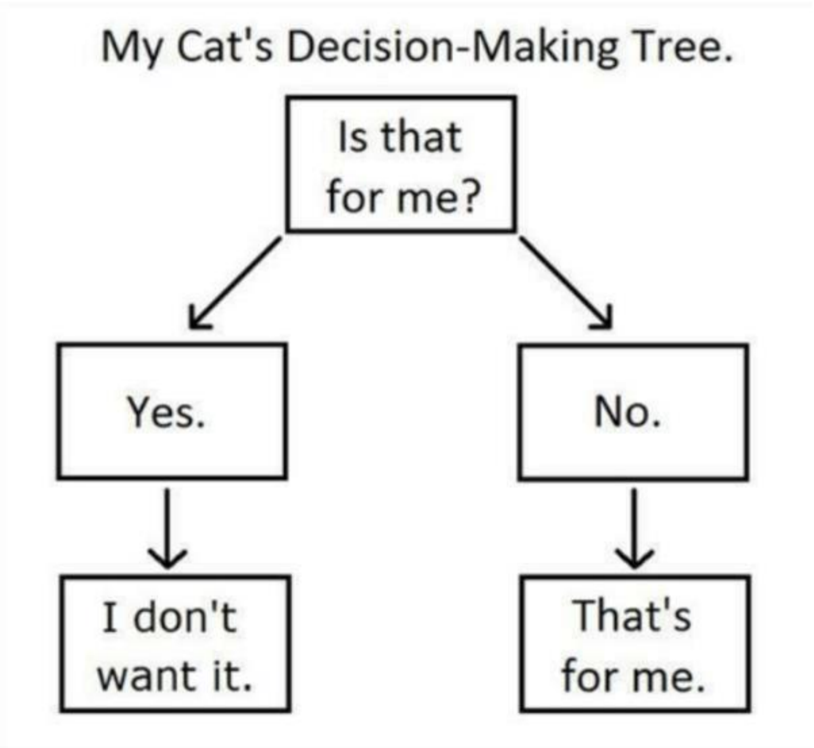
Notebook demos are helpful!!!

So I got two for you this week:

One gives you example of a decision tree and random forest.

Another uses Kaggle Titanic data to predict whether a passenger from the Titanic is alive or not, based on the existing data.

As usual, I’ll explain the concepts as clear as I can in section, and the notebook is there for you to play around with, no pressure!



My Cat's Decision-Making Tree.

Is that for me?

Yes.

No.

I don't want it.

That's for me.

Decision Tree

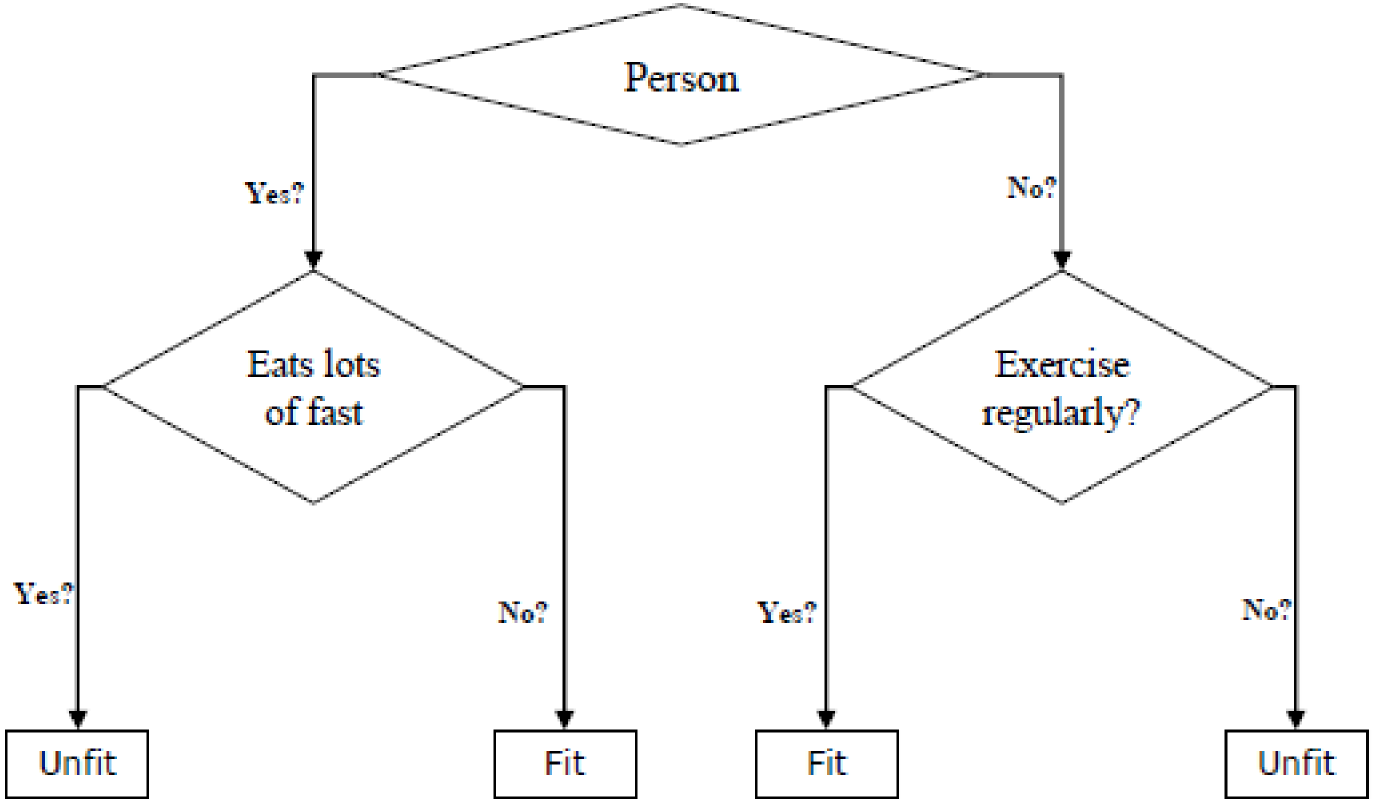
AND

RANDOM FORREST (IN A BIT)

DECISION TREE!

WHAT IS IT?

Decision tree algorithm creates a model that predicts the value of a target variable based on several input variables.



DECISION TREE!

WHAT TYPE OF PROBLEMS CAN IT SOLVE?

Classification And Regression Tree (CART)

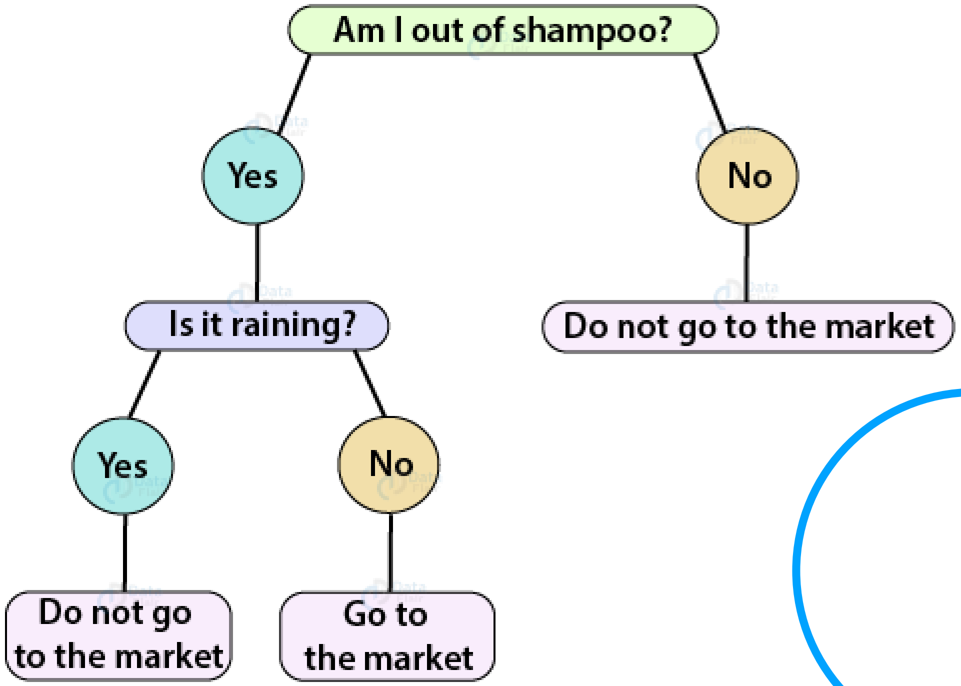
Classification tree analysis:

- When the predicted outcome is the class (discrete) to which the data belongs.

Regression tree analysis:

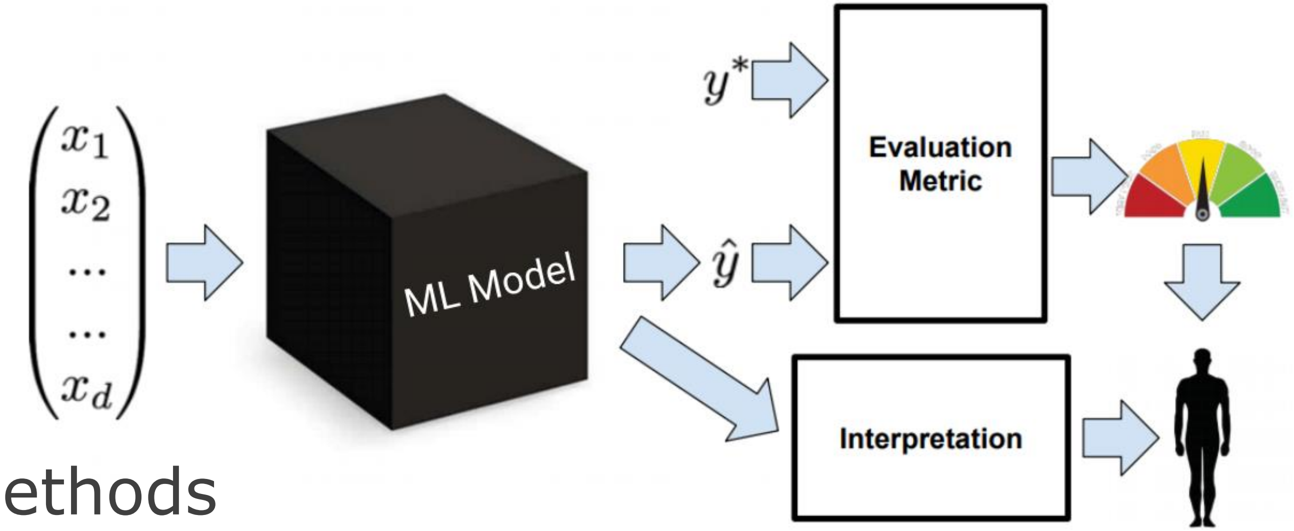
- When the predicted outcome can be considered a real number (e.g. the price of a house, or a patient's length of stay in a hospital).

Decision Trees Example



DECISION TREE!

WHEN TO USE IT?



Black Box vs. Transparent Methods

- **Transparent**
  - If we're using Machine Learning to actually get insights from the data, "black box" models are almost useless and it's best to stick with simpler, transparent techniques.
  - **Decision Tree!!!**
- **Black Box**
  - If we need to build a model that will be directly used for some task and only show it's end results, then we don't really care about building some kind of "black box" if it's accurate enough (image or speech recognition for example).
  - Deep Learning, ensemble learning, etc. NOT decision tree.

DISCUSSION QUESTION:
Why do we care about interpretability

DECISION TREE!

SOUNDS MAGICAL. ANY DOWNSIDES?

Of course! ~~Stop dreaming there is no all perfect algorithm out there~~

The main downsides of Decision Trees are their tendency to over-fit.

- If your tree goes too big (deep), you will overfit pretty fast!

They are also unable to grasp relationships between features.

- Cuz it’s a tree like structure so you will lose some relationships between features.

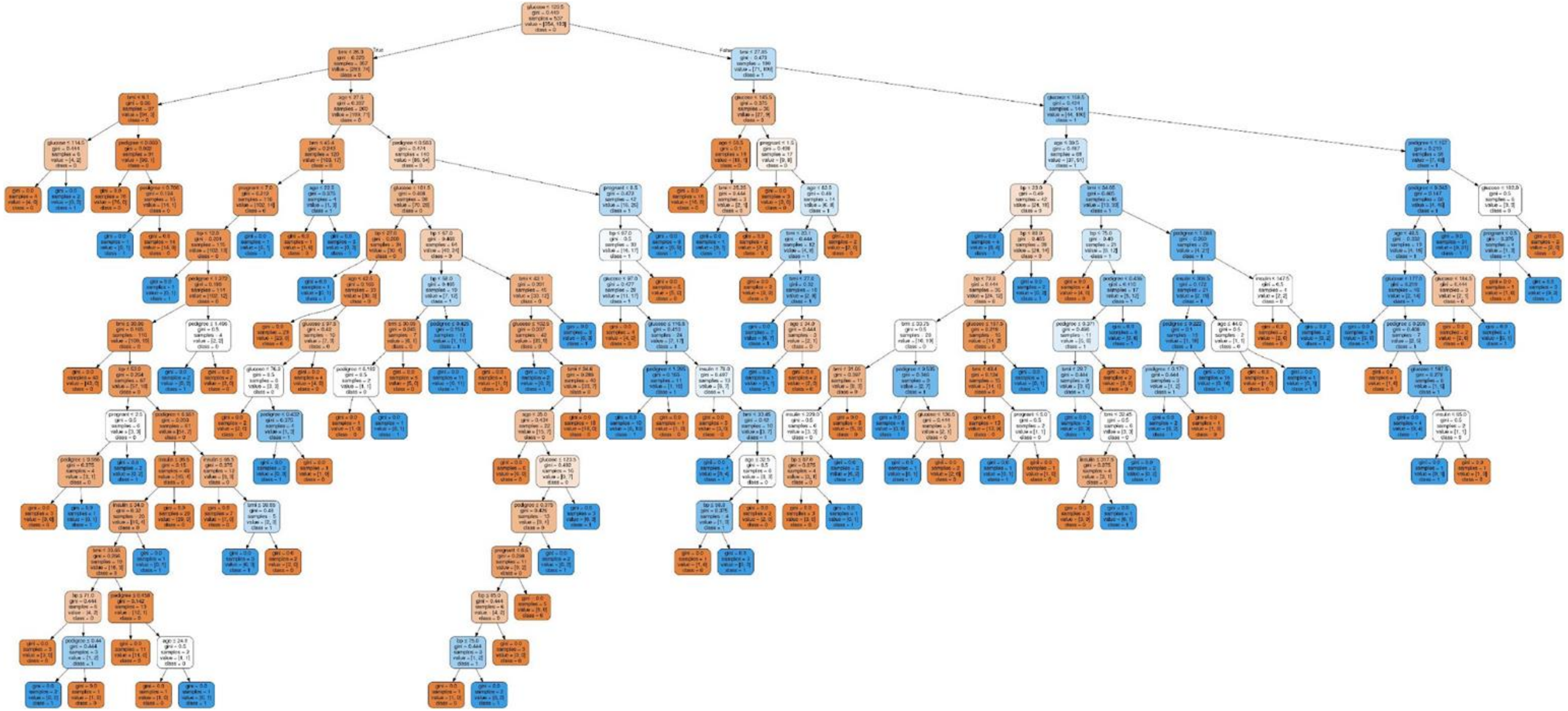
They use greedy learning algorithms, because we usually use greedy traversal when dealing with a tree.

- This is a downside because it is not guaranteed to find the global optimal model.

DECISION TREE!

SOUNDS MAGICAL. ANY DOWNSIDES?

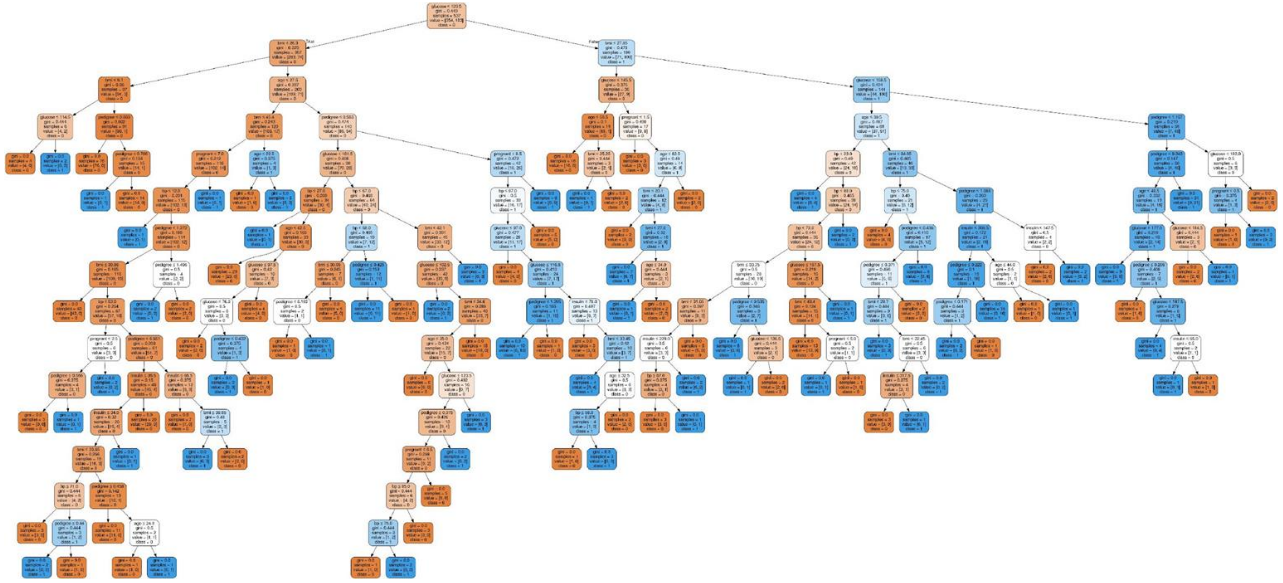
One example of a very complex tree:



DECISION TREE!

ANY WAY TO IMPROVE THE ABOVE DOWNSIDES?

Sure! Using them in a Random Forest (We will talk about that in a bit) helps mitigate some of this issues.



DECISION TREE!

THE ALGORITHM? IF YOU ARE CURIOUS.

Select the best attribute using Attribute Selection Measures(ASM) to split the records.

Make that attribute a decision node and breaks the dataset into smaller subsets.

Starts tree building by repeating this process recursively for each child until one of the condition will match:

- All the tuples belong to the same attribute value.
- There are no more remaining attributes.
- There are no more instances.

DECISION TREE!

ATTRIBUTE SELECTION MEASURES(ASM)!

What? - Attribute selection measure is a heuristic for selecting the splitting criterion that partition data into the best possible manner.

- It is also known as splitting rules because it helps us to determine breakpoints for tuples on a given node.

How? - ASM provides a rank to each feature (or attribute) by explaining the given dataset.

- In class, we talked about using classification error as a splitting rile, but there are other metrics, the main purpose is the same.
- Best score attribute will be selected as a splitting attribute.
- For example, **Gini Impurity.**

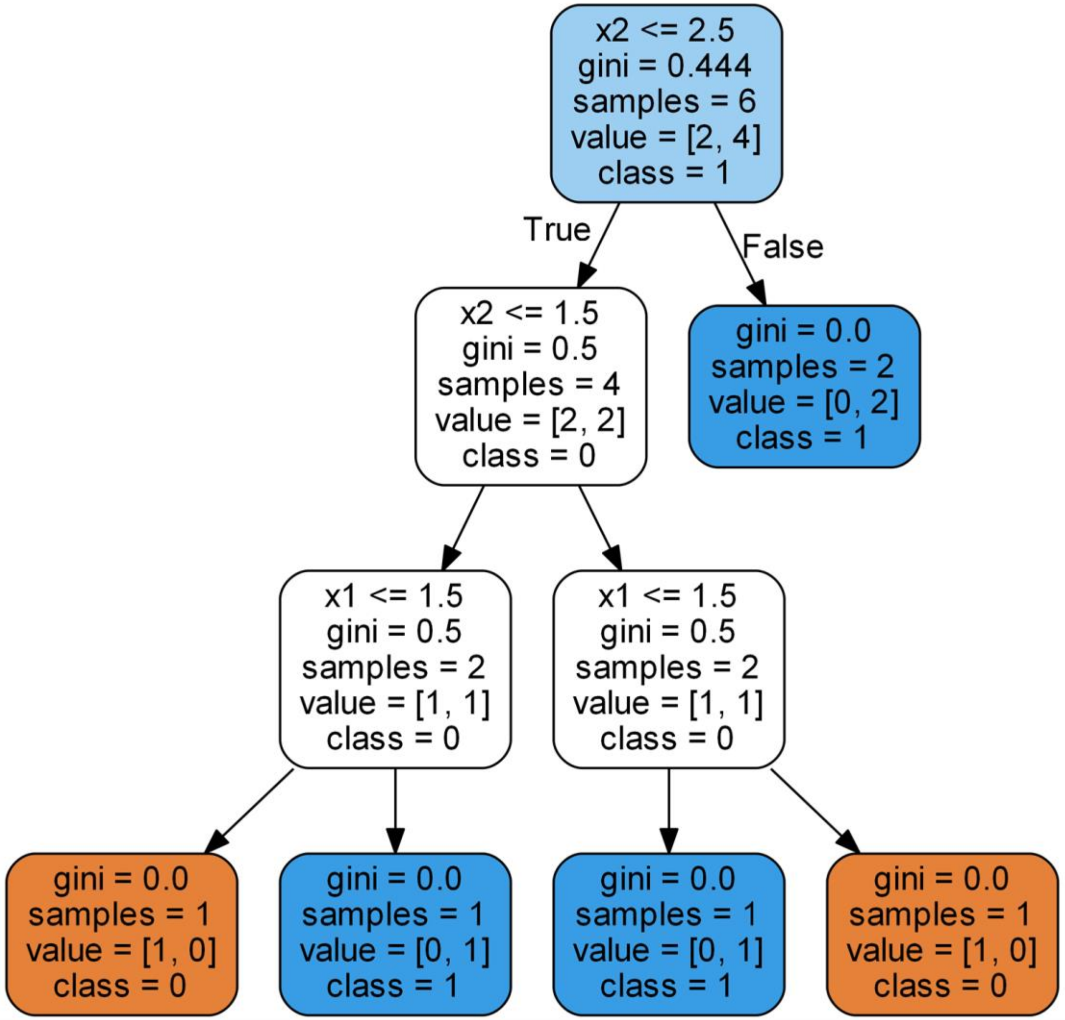
DECISION TREE!

ATTRIBUTE SELECTION MEASURES(ASM)!

Gini Impurity

The Gini Impurity of a node is the probability that a randomly chosen sample in a node would be incorrectly labeled if it was labeled by the distribution of samples in the node.

For example, in the top (root) node, there is a 44.4% chance of incorrectly classifying a data point chosen at random based on the sample labels in the node.



DECISION TREE!

ATTRIBUTE SELECTION MEASURES(ASM)!

Gini Impurity (Math)

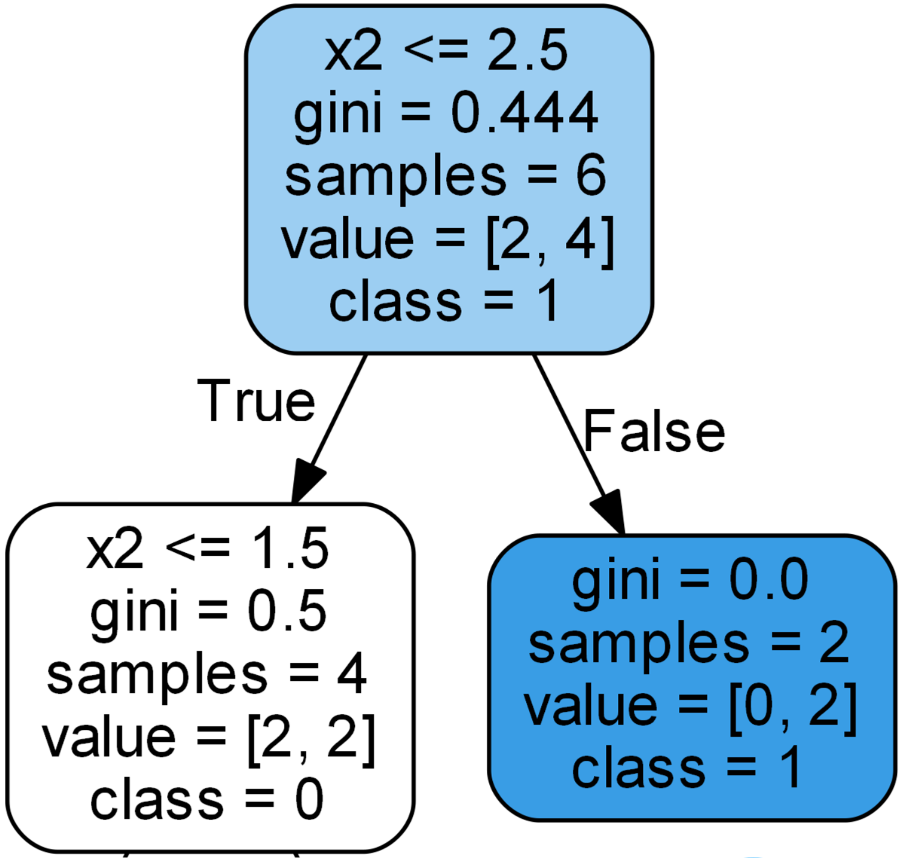
Just here for those of you who are curious…

$$I_G(n) = 1 - \sum_{i=1}^{J} (p_i)^2$$

The Gini Impurity of a node *n* is 1 minus the sum over all the classes J (for a binary classification task this is 2) of the fraction of examples in each class p_i squared.

For example, the root Gini Impurity is calculated:

$$I_{root} = 1 - ((\tfrac{2}{6})^2 + (\tfrac{4}{6})^2) = 1 - \tfrac{5}{9} = 0.444$$



DECISION TREE!

ATTRIBUTE SELECTION MEASURES(ASM)!

Gini Impurity

At each node, the decision tree searches through the features for the value to split on that results in the *greatest reduction* in Gini Impurity.

As the algorithm recursively go down the tree branches, the weighted total Gini Impurity at each level of tree must decrease.

Eventually, the weighted total Gini Impurity of the last layer goes to 0.

DECISION TREE!

ATTRIBUTE SELECTION MEASURES(ASM)!

Gini Impurity

Eventually, the weighted total Gini Impurity of the last layer goes to 0.

Each node is completely pure and there is no chance that a point randomly selected from that node would be misclassified.

While this may seem like a positive, it means that the model may potentially be overfitting because the nodes are constructed only using *training data.*

DECISION TREE!

WHY SHOULD YOU CARE?

The flowchart-like structure helps you in decision making.

It's visualization like a flowchart diagram which easily mimics the human level thinking.

That is why decision trees are easy to understand and interpret.

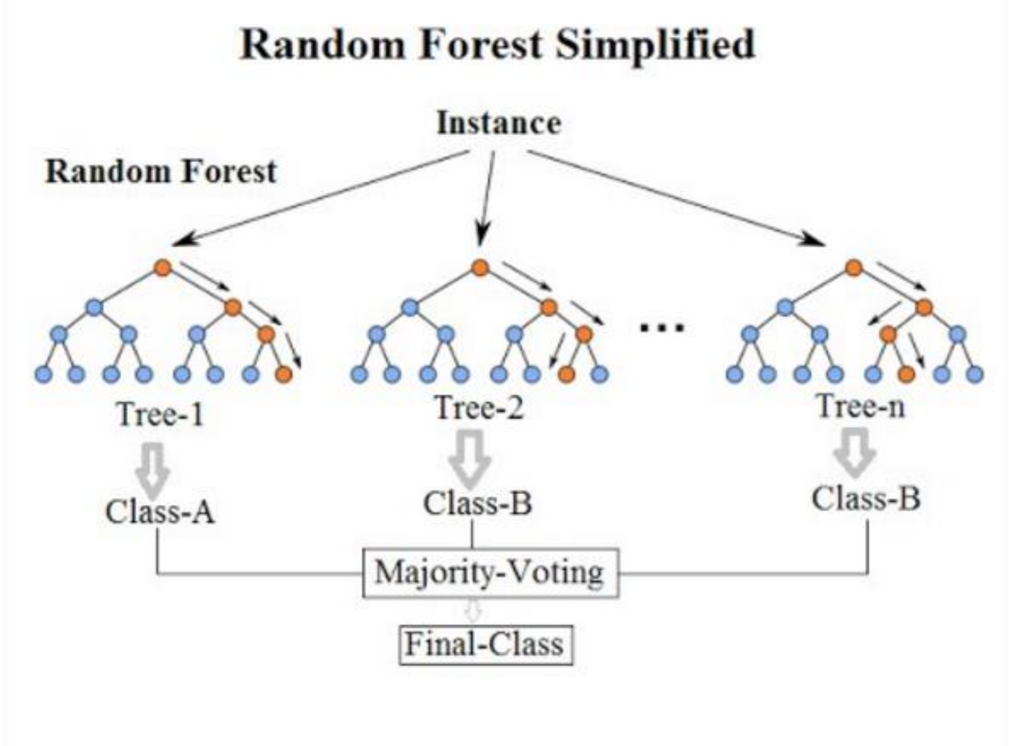
DECISION TREE!

WHY SHOULD YOUR BOSS CARE?

If your boss is a marketing manager, he wants a set of customers who are most likely to purchase your product.

- This is how he can save his marketing budget by finding your audience.

If your boss is a loan manager, he needs you to help identify risky loan applications to achieve a lower loan default rate.



Random Forest Simplified

RANDOM FORREST

WHY A FOREST IS BETTER THAN ONE TREE

You might be tempted to ask why not just use one decision tree?

It seems like the perfect classifier since it did not make any mistakes!

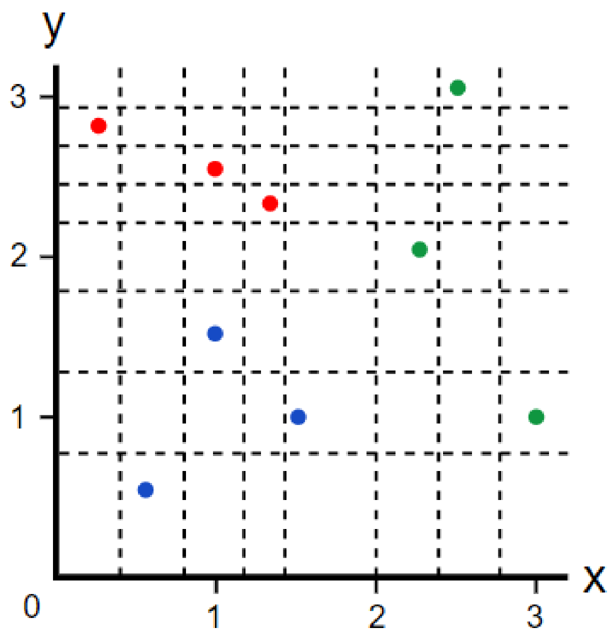
OVERFITTING:
WHY A FOREST IS BETTER THAN ONE TREE

A critical point to remember is that the tree made no mistakes **on the training data**. We expect this to be the case since we gave the tree the answers and didn’t limit the max depth (number of levels).

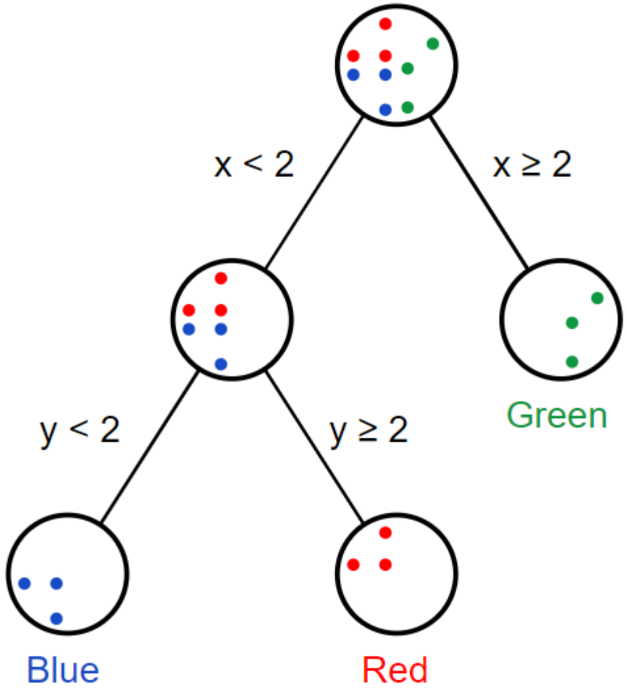
The objective of a machine learning model is to generalize well to **new data** *it has never seen before.*

PART I: BAGGING

I'll give an easy example here:
Say I am training a tree on classifying colors of n points.
If I train the tree it will look something like this:



All Thresholds



PART I: BAGGING

In order to prevent overfitting, what we do is we train many smaller (less deep) trees, and bag them together.

Consider the following algorithm to train a bundle of decision trees given a dataset of *n* points:

Sample, **with replacement**, *n* training examples from the dataset.

Train a decision tree on the *n* samples.

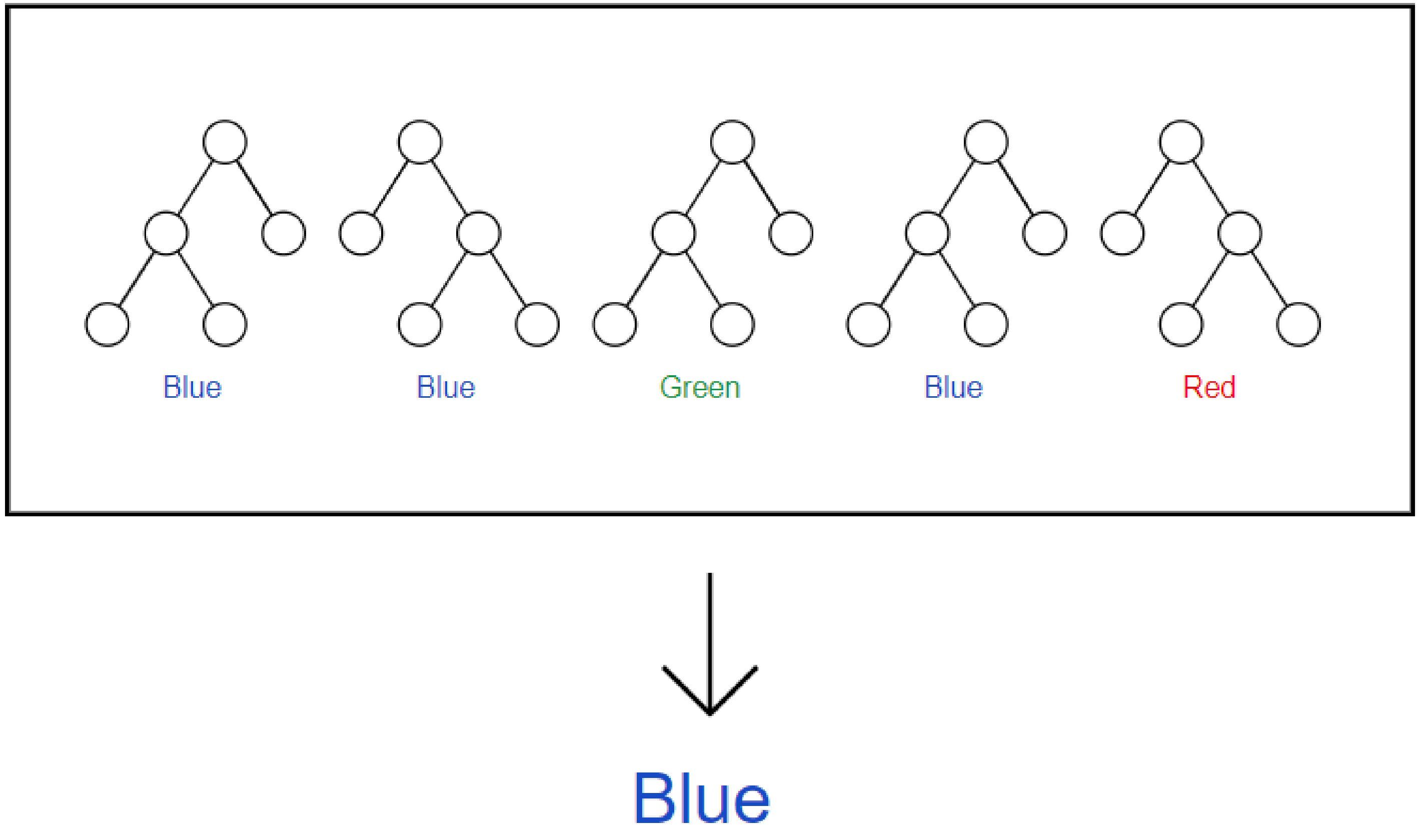
Repeat *t* times, for some *t*.

PART I: BAGGING

To make a prediction using this model with *t* trees, we aggregate the predictions from the individual decision trees and either:

Take the **majority vote** if our trees produce class labels (like colors).

Take the **average** if our trees produce numerical values (e.g. when predicting temperature, price, etc).

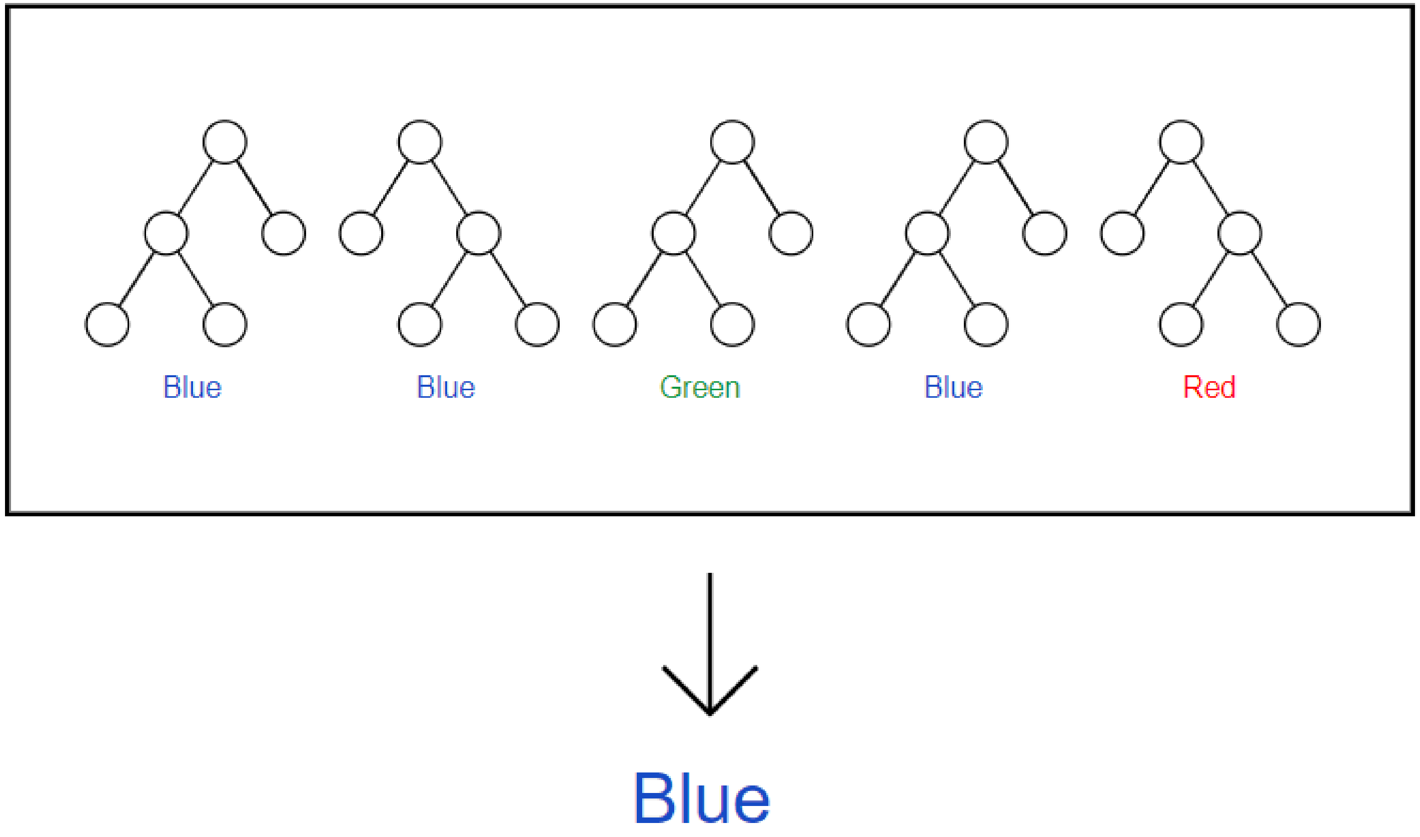


Bagged Decision Trees predicting color

PART II: RANDOM FOREST

Bagged decision trees have only one parameter: *t*, the number of trees.

For example, t for the bag of trees is 5. We use the majority vote to classify the point as blue.

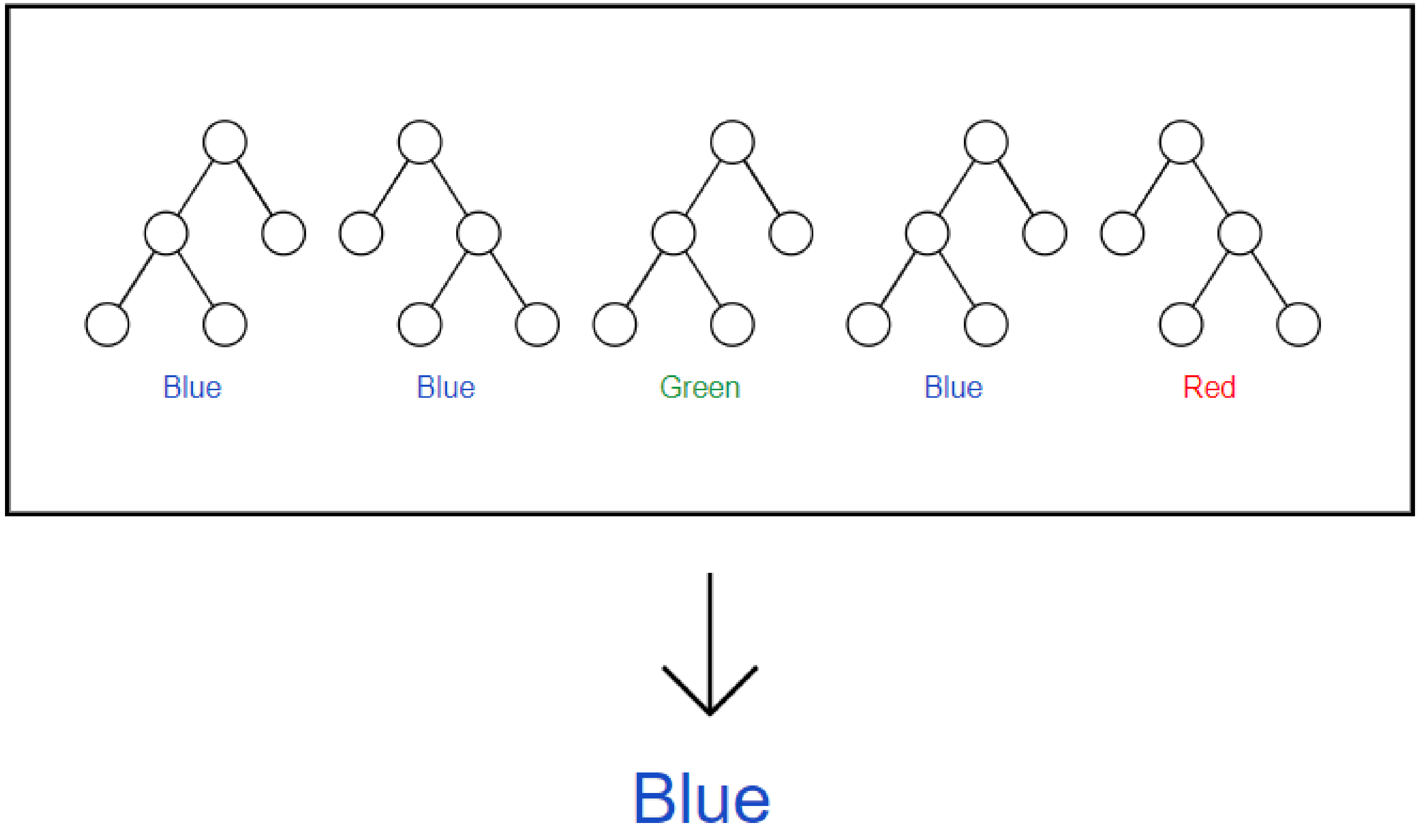


Bagged Decision Trees predicting color

PART II: RANDOM FOREST

Random Forests have a second parameter that controls **how many features to try when finding the best split**.

Our simple dataset for this tutorial only had 2 features (*x* and *y*), but most datasets will have far more (hundreds or thousands).



Bagged Decision Trees predicting color

PART II: RANDOM FOREST

Suppose we had a dataset with pp features. Instead of trying all features every time we make a new decision node, we **only try a subset of the features**, usually of size $\sqrt{p}$ or $\frac{p}{3}$.

We do this primarily to inject randomness that makes individual trees more unique and **reduces correlation between trees**, which improves the forest’s performance overall. This technique is sometimes referred to as **feature bagging**.

License

This material is originally made by Hongjun Wu for the course CSE416: Introduction to Machine Learning in the Spring 2020 quarter taught by Dr. Valentina Staneva, at University of Washington Paul G. Allen School of Computer Science and Engineering.

It was originally made for educational purpose, in a section taught by teaching assistants to help students explore material in more depth.

Any other materials used are cited in the Credits section.

Anyone, especially other educators and students, are welcomed and strongly encouraged to study and use this material.