Predictive Analytics

Week 10: Classification II

Semester 2, 2018

Discipline of Business Analytics, The University of Sydney Business School

Week 10: Classification II

1. Decision Tree Intuition
2. Classification Trees
3. Regression Trees
4. Random Forest

Readings: Chapters 8.1 and 8.2.2
Exercice questions: Chapter 8.4 of ISL, Q1, Q3 and Q4.

Decision Tree Intuition

Decision trees intuition

- Non-parametric (any other nonparametric method we learnt before?)

- Supervised learning method that can be used for both classification and regression.

- Through incorporating a set of if-then-else rules, decision tree can be employed to predict target variable given data features

Decision trees intuition

- Try to discover the **pattern** under which the customer will purchase the product

- Divide data set into subsets (branches of a tree)

- Check whether the **stopping criteria** is met

  If yes

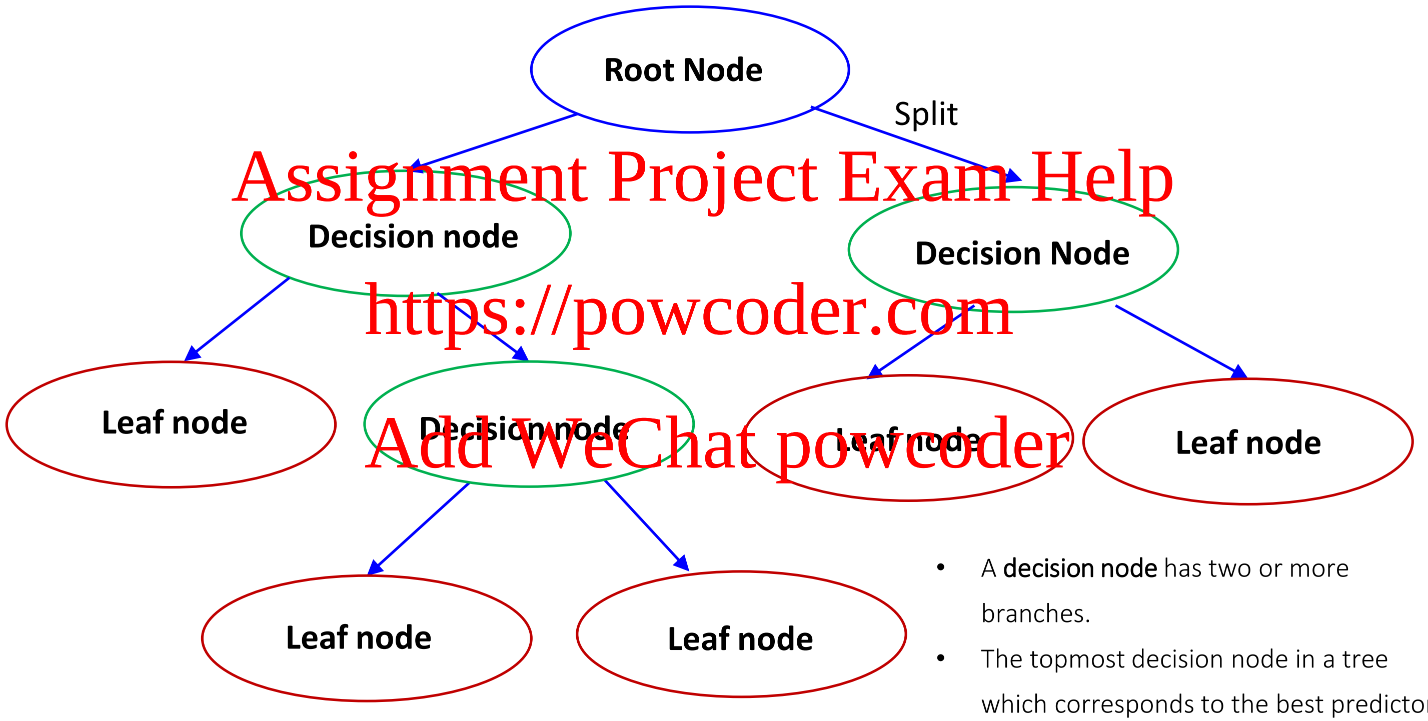
  stop dividing

  Else

  keep dividing

- For a new customer, based on the features, we can see which subset the customer will fall into

Decision Trees example



- A **decision node** has two or more branches.
- The topmost decision node in a tree which corresponds to the best predictor called **root node**.
- **Leaf node** represents a decision.

Types of decision trees

Decision trees used in machine learning are of two main types:

- ❑ Classification tree analysis is when the predicted outcome is the class to which the data belongs. Target variable is categorical.

- ❑ Regression tree analysis is 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). Target variable is continuous.

Classification And Regression Tree (CART), Breiman et al., (1984). An umbrella term used to refer to both of the above techniques.

Classification Trees

- The task of growing a classification tree is quite similar to the task of growing a regression tree

- Categorical response variable, e.g. yes/no, 1/0

- For a classification tree, we predict that each observation belongs to the most commonly occurring class (mode) of training observations in the region to which it belongs

- In interpreting the results of a classification tree, we are often interested not only in the class prediction corresponding to a particular leaf node region, but also in the class **proportions** among the training observations that fall into that region

Classification tree

Customer	Income	Education	Marital Status	Purchase
1	Medium	University	Single	Yes
2	High	University	Single	No
3	High	University	Married	No
4	Low	University	Single	Yes
5	Low	High school	Single	Yes
6	Low	High school	Married	No
7	Medium	High school	Married	Yes
8	High	University	Single	No
9	High	High school	Single	Yes
10	Low	High school	Single	Yes
11	High	High school	Married	Yes
12	Low	University	Married	No
13	High	University	Single	No
14	Medium	University	Married	Yes
15	Medium	High school	Single	Yes

We can have duplicated records.

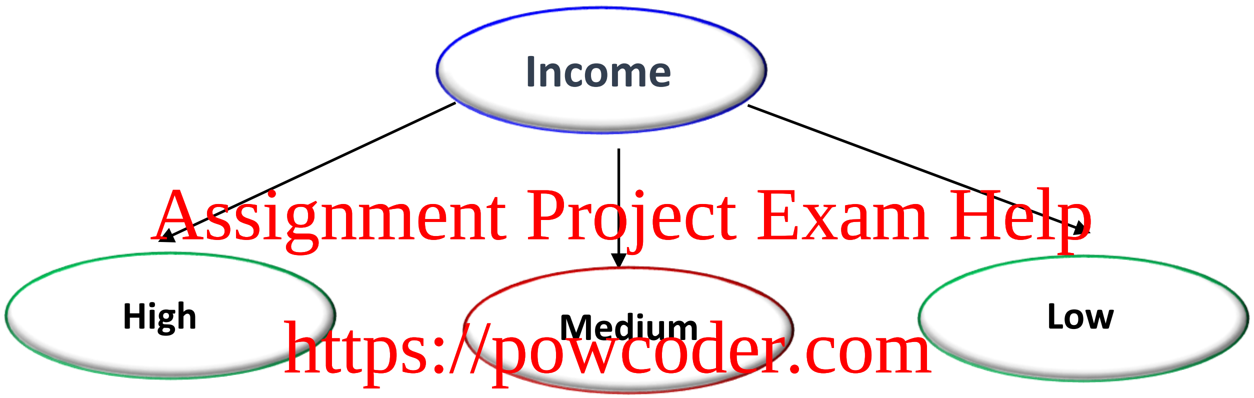
- We need to build the tree from the root node with one feature and then split examples into subsets

- How to select this feature?

- Idea: a good feature splits the examples into subsets that are (ideally) "all positive" or "all negative"

- Purity

Let’s start the decision tree with feature income. Why?



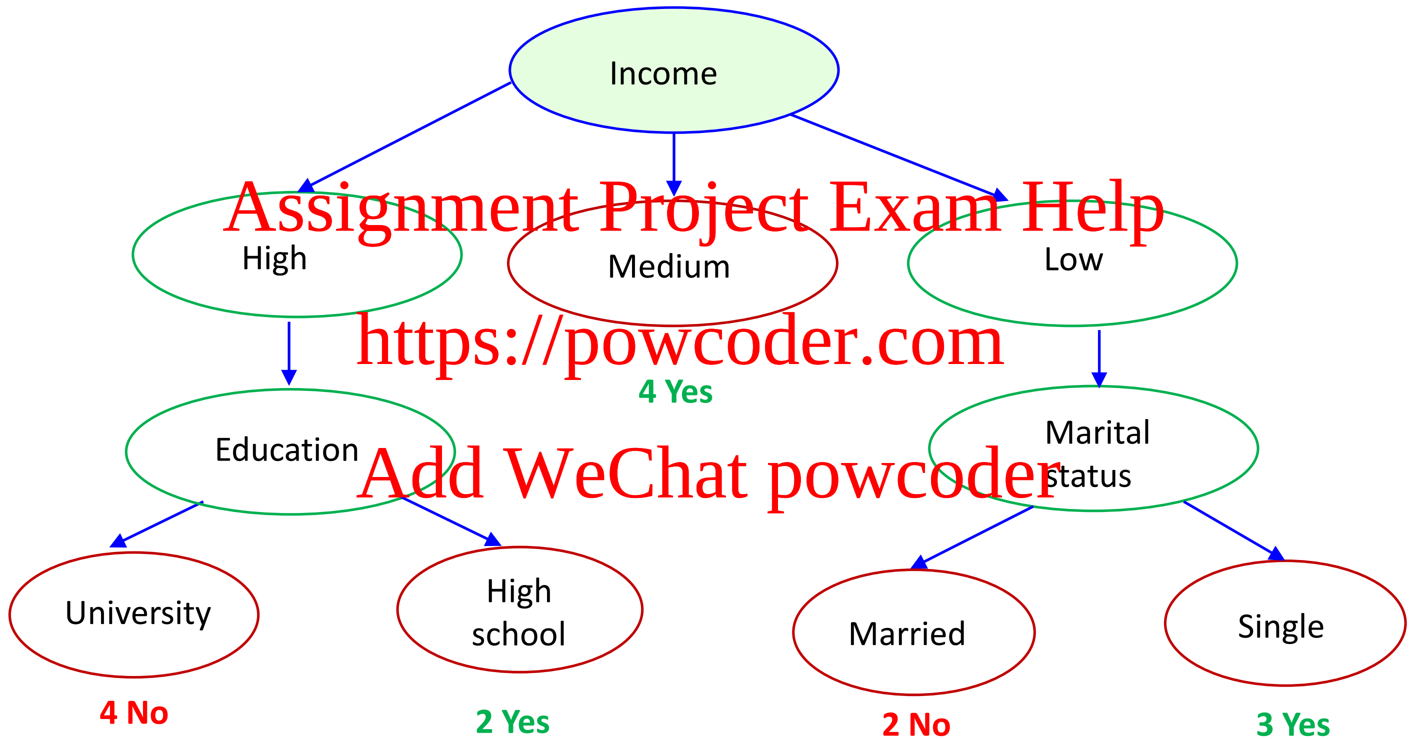
PURE set

Customer	Income	Education	Marital Status	Purchase
3	Medium	University	Single	Yes
7	Medium	High school	Married	Yes
12	Medium	University	Married	Yes
13	Medium	High school	Single	Yes

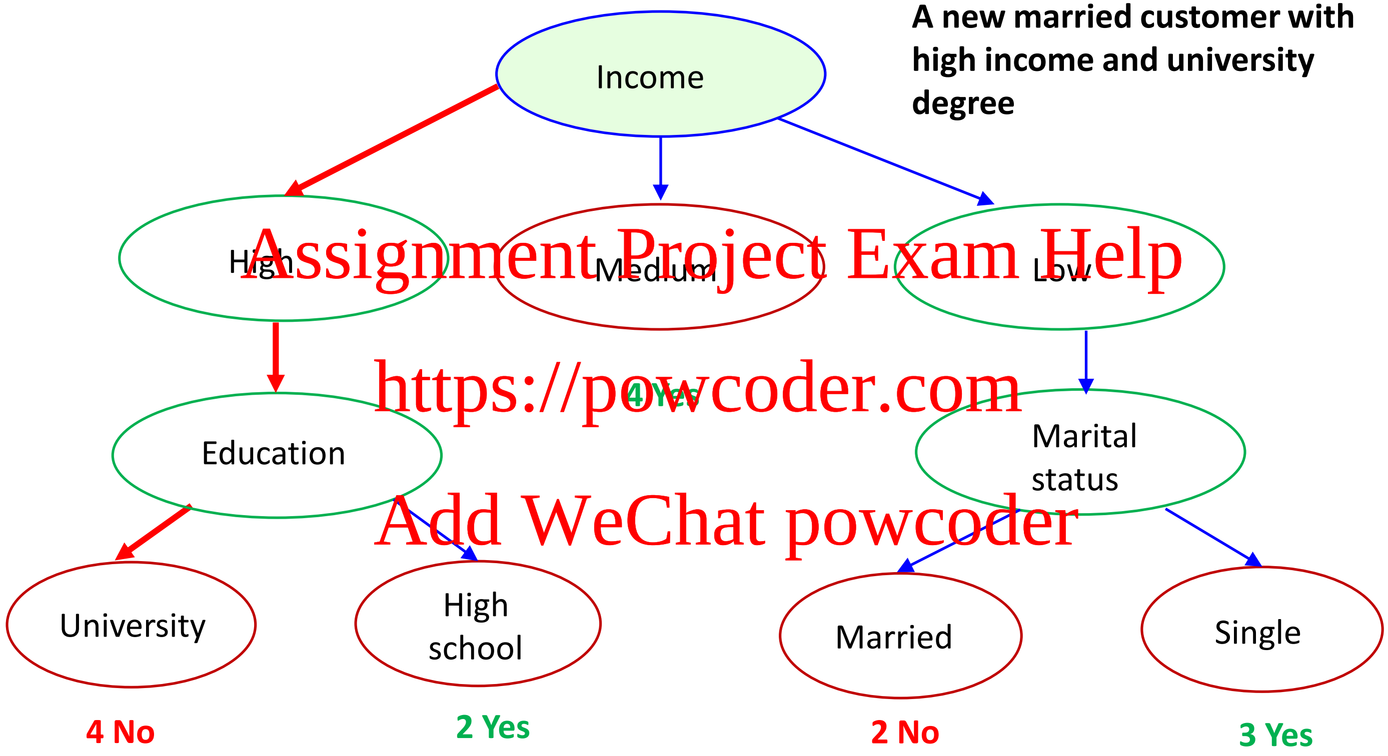
Customer	Income	Education	Marital Status	Purchase
1	High	University	Single	No
2	High	University	Married	No
8	High	University	Single	No
9	High	High school	Single	Yes
11	High	High school	Married	Yes
15	High	University	Single	No

Customer	Income	Education	Marital Status	Purchase
4	Low	University	Single	Yes
5	Low	High school	Single	Yes
6	Low	High school	Married	No
10	Low	High school	Single	Yes
14	Low	University	Married	No

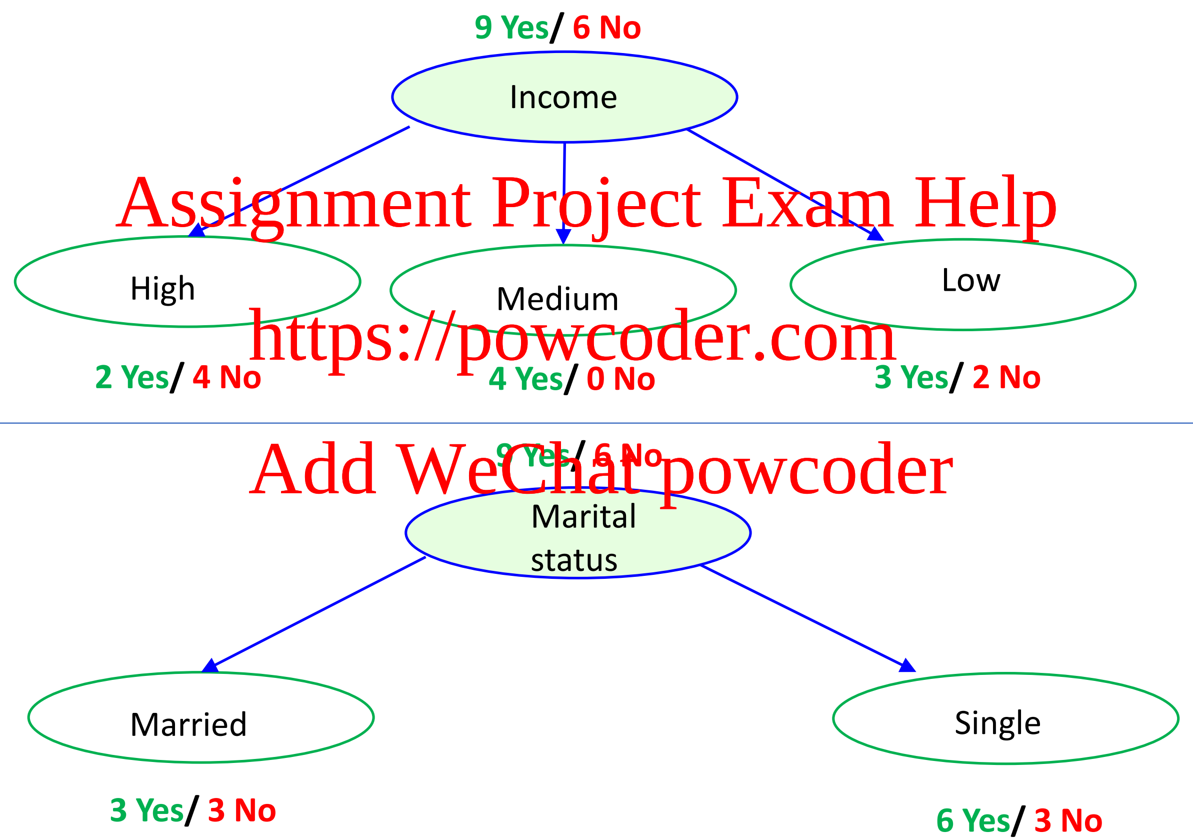
Let's start the decision tree with feature income. Why?



A new married customer with high income and university degree



Best feature of splitting

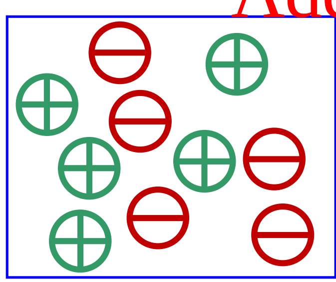


Which split is better?

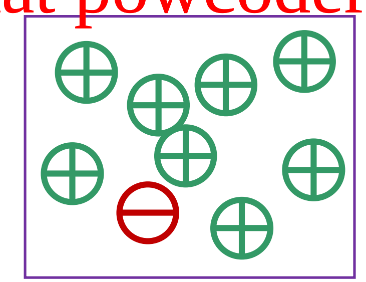
Entropy intuition

- Entropy is a concept originally from physics and measures the disorder in a data set
- In decision trees, we use entropy $\boldsymbol{H}(\boldsymbol{S})$ to measure of the amount of uncertainty in the data set $\boldsymbol{S}$.
- The entropy will be a small value if the dataset is pure.

- **Smaller entropy, less disorder, higher PUTIRY (CERTAINTY)**
- **Larger entropy, more disorder, higher IMPUTIRY (UNCERTAINTY)**



$\boldsymbol{H}(\boldsymbol{S}) = \mathbf{1}$



$\boldsymbol{H}(\boldsymbol{S}) = \mathbf{0.469}$

A glass of water and ice cubes, which one is purer?

Best feature of splitting

Measure the **PURITY** of the split:

Aim to be more certain about Yes/No after the spit

- Pure set: (4 yes/0 no)=> 100% certain
- Impure set: 3 yes/3 no => 50% certain and 50% uncertain

- Impure set: 1 yes/3 no => 25% certainty and 75% uncertain
  Should be as **PURE** as
- Impure set: 3 yes/1 no => 75% certainty and 25% uncertain

Entropy calculation

Entropy $\boldsymbol{H}(\mathbf{S})$ is a measure of the amount of uncertainty in the data set $\mathbf{S}$. The entropy will be a **small** value if the dataset is **pure**.

$$H(\mathbf{S}) = \sum_{k=1}^{K} p_k(\mathbf{S}) \log_2\left(\frac{1}{p_k(\mathbf{S})}\right) = -\sum_{k=1}^{K} p_k(\mathbf{S}) \log_2\left(p_k(\mathbf{S})\right)$$

- $\mathbf{S}$: The current (data) set for which entropy is being calculated (changes every iteration of the ID3 algorithm)
- $p_k(\mathbf{S})$: The proportion of the number of elements in class k to the number of elements in set $\mathbf{S}$. K classes in total in $\mathbf{S}$.
- $\sum_{k=1}^{K} p_k(\mathbf{S}) = 1$
- $p_k(\mathbf{S})\log_2\left(p_k(\mathbf{S})\right)$ equals zero when $p_k(\mathbf{S}) = 0$.

Entropy- two classes

More specifically, for a training set with $\boldsymbol{p}$ **positive examples** and $\boldsymbol{n}$ **negative examples**:

$$H(\mathbf{S}) = -\boxed{\frac{p}{p+n}} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$

% of positive examples in $\boldsymbol{S}$

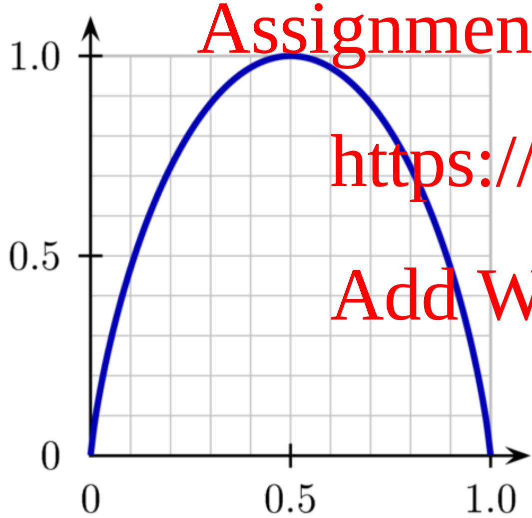
Equivalently:

$$H(\mathbf{S}) = -\boxed{p_+(\mathbf{S})} \log_2 p_+(\mathbf{S}) - p_-(\mathbf{S}) \log_2 p_-(\mathbf{S})$$

Interpretation: assume an item belongs to **S**, how many **bits** of information are required to tell whether **x** is positive or negative. The smaller it is, the higher certainty.

Entropy- two classes

A two class problem



When $\boldsymbol{H}(\mathbf{S}) = \mathbf{0}$, the set $\mathbf{S}$ is perfectly classified, e.g. all elements in $\boldsymbol{S}$ are of the same class

$p_-(\mathbf{S}) = 0.5, p_+(\mathbf{S}) = 0.5, H(\mathbf{S}) = 1$

$p_-(\mathbf{S}) = 0, p_+(\mathbf{S}) = 1, H(\mathbf{S}) = 0$

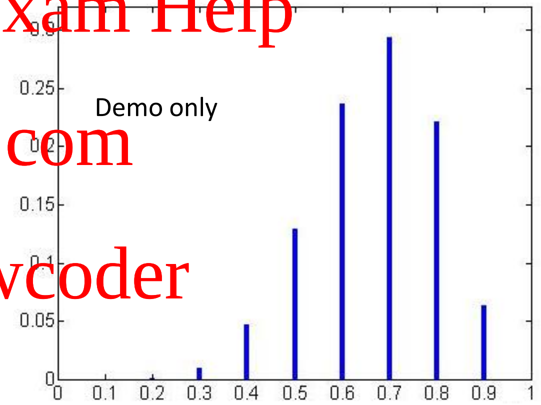
Symmetric

$p_+(\mathbf{S}) = 0, p_-(\mathbf{S}) = 1, H(\mathbf{S}) = 0$

Entropy- multiple classes

If there are more than two classes: $1, 2, \ldots, K$:

$$
\begin{aligned}
H(\mathbf{S}) = &-p_1(\mathbf{S})\log_2 p_1(\mathbf{S}) \\
&-p_2(\mathbf{S})\log_2 p_2(\mathbf{S}) \\
&-p_3(\mathbf{S})\log_2 p_3(\mathbf{S}) \\
&\ldots\ \ldots \\
&-p_K(\mathbf{S})\log_2 p_K(\mathbf{S})
\end{aligned}
$$

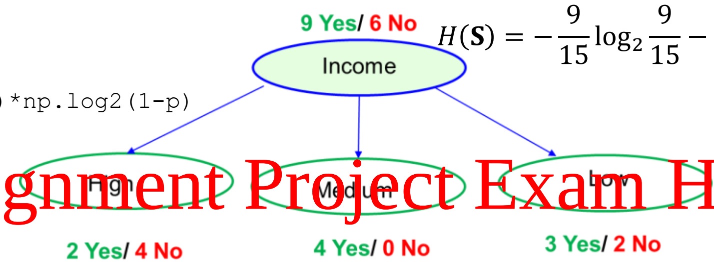


K **classes in total in S**

$$\sum_{k=1}^{K} p_i(\mathbf{S}) = 1$$

Entropy example

```
# entropy calculator
p= 3.0/5
H= - p*np.log2(p)-(1-p)*np.log2(1-p)
```



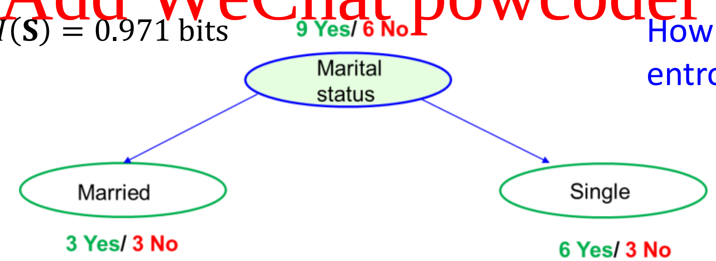
$$H(\mathbf{S}) = -\frac{9}{15}\log_2\frac{9}{15} - \frac{6}{15}\log_2\frac{6}{15} = 0.971 \text{ bits}$$

$$H(\mathbf{S}_1) = -\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6} = 0.918 \text{ bits}$$

$$H(\mathbf{S}_2) = 0 \text{ bits}$$

$$H(\mathbf{S}_3) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.971 \text{ bits}$$

$$H(\mathbf{S}) = 0.971 \text{ bits}$$



How to merge these entropy together?

$$H(\mathbf{S}_1) = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1 \text{ bit}$$

$$H(\mathbf{S}_2) = -\frac{6}{9}\log_2\frac{6}{9} - \frac{3}{9}\log_2\frac{3}{9} = 0.918 \text{ bits}$$

Other Measurements

- Entropy is not the only measurement of selecting the best feature to split
- Other measurements include:
  - Gini index

$$H(\mathbf{S}) = \sum_{k=1}^{K} p_i(\mathbf{S})(1 - p_i(\mathbf{S}))$$

  - The Gini index and the entropy are similar numerically
  - Misclassification rate: not sufficiently sensitive for tree-growing. James et al., (2014).

Information Gain

- How much information do we gain if we disclose/split the value of some features?
- Answer: uncertainty before minus uncertainty after
- **Information Gain (IG)** or reduction in entropy from the feature test
- Information Gain is a measure of the disorder/uncertainty decrease achieved by splitting the data set S
- Choose the feature split with the **largest** IG

$$\boxed{\textbf{Information Gain}} = \textbf{Entropy before} - \boxed{\textbf{Entropy after}}$$

We want this term to be large

Weighted sum of Entropy.
We want this term to be small.

Information Gain

Information gain IG(S,A) is the measure of the difference in entropy from before to after the data set $\mathbf{S}$ is split on an feature A.

In other words, how much **uncertainty** in $\mathbf{S}$ was **reduced** after splitting set $\mathbf{S}$ on feature A.

$$IG(\mathbf{S}, A) = H(\mathbf{S}) - EH(A)$$

$H(\mathbf{S})$ – Entropy of set $\mathbf{S}$

$EH(A)$ – Expected entropy with split by feature A

Expected Entropy

A selected feature A with J distinct values, e.g. feature "income" has $J = 3$ possible values "high", "medium" and "low", partitions the training set $\mathbf{S}$ into J subsets/branches $\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_J$

The **expected entropy** with split by feature A is:

Weights based on size of the subset

$$EH(A) = \sum_{j=1}^{J} \boxed{\frac{|\mathbf{S}_j|}{|\mathbf{S}|}} \boxed{H(\mathbf{S}_j)}$$

Note this is the entropy of the subset, calculated according to the target categories

$\mathbf{S}$: the current (data) set for which entropy is being calculated
$\mathbf{S}_j$: subset j
Expected entropy is a measurement of subsets impurity.

Information Gain example

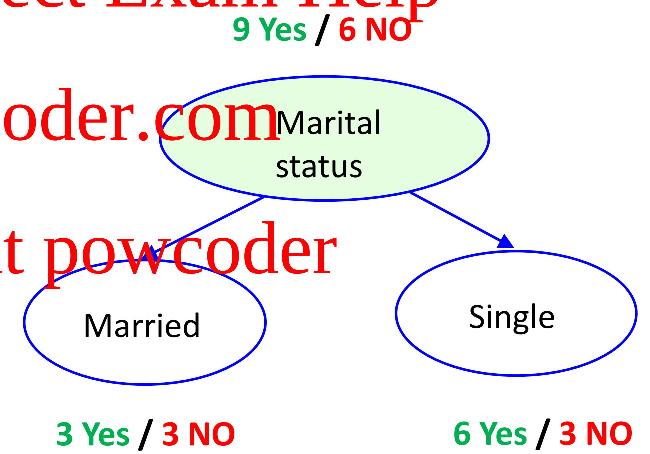
Entropy before split. High impurity.

$$H(\mathbf{S}) = -\frac{9}{15}\log_2\frac{9}{15} - \frac{6}{15}\log_2\frac{6}{15} = 0.971 \text{ bits}$$

Weights based on size of the subsets to S

$$IG(S,A) = H(S) - \boxed{\frac{6}{15}} H(S_{\text{Married}}) - \boxed{\frac{9}{15}} H(S_{\text{Single}})$$

$$= 0.97 - \frac{6}{15}\times 1 - \frac{9}{15}\times 0.91 = 0.0239$$



If split on “marital status”, we would **GAIN** 0.0239 bits on certainty.
Or we are 0.0239 bits more certain.

Entropy after split

$$H(S_{\text{Married}}) = 1 \qquad H(S_{\text{Single}}) = 0.918$$

Information Gain drawback

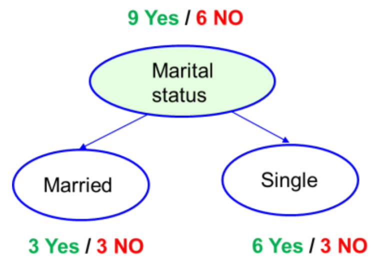
- IG favours split on an feature with many values (many leaf nodes): causing bias
- If 1 feature splits in many more classes than another, it has an (unfair) advantage if we use information gain
- The Gain-Ratio is designed to compensate for this problem

$$\text{GainRatio} = \frac{\text{Information Gain}}{\boxed{\text{Split Entropy}}}$$

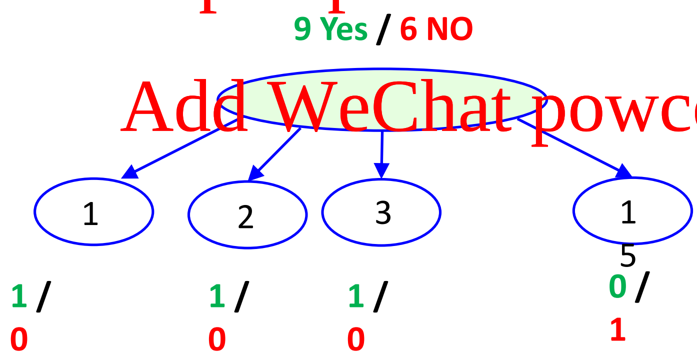
Penalize split with too many small subsets

$$\text{Split\_Entropy}(\boldsymbol{S}, A) = -\sum_{j=1}^{J} \frac{|\mathbf{S}_j|}{|\mathbf{S}|} \log_2 \frac{|\mathbf{S}_j|}{|\mathbf{S}|}$$

Split Entropy Example



$$\text{Split Entropy} = -\frac{6}{15}log_2\left(\frac{6}{15}\right) - \frac{9}{15}log_2\left(\frac{9}{15}\right) = 0.971$$



Penalize split with too many small subsets, although the IG for such split is high.

$$\text{Split Entropy} = -15\left[\frac{1}{15}log_2\left(\frac{1}{15}\right)\right] = 3.907$$

Split over numeric features

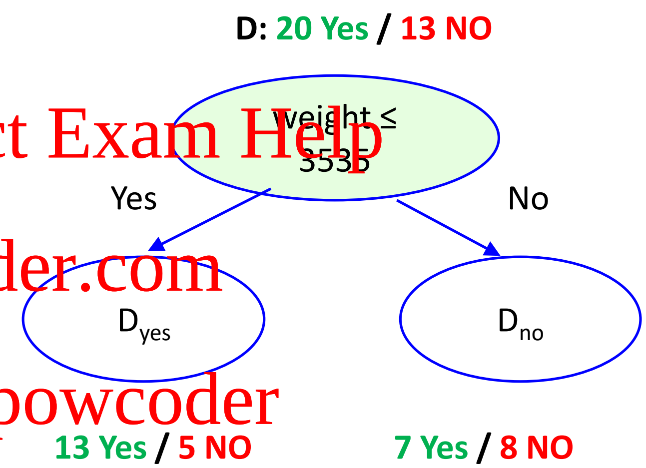
- What should we do if some of the features are numeric/continuous?
- We use the form of $x < \theta$ where θ is called a splitting value or cutting point.

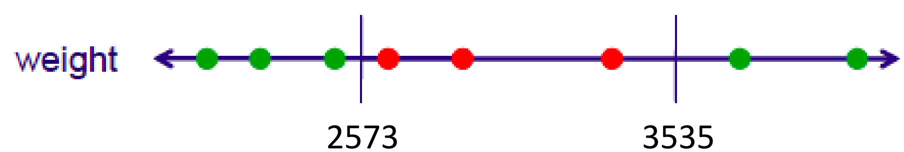
Infinite number of possible split values!!!

mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	maker
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	america
bad	4	121	110	2600	12.8	77	europe
bad	8	350	175	4100	13	73	america
bad	6	198	95	3102	16.5	74	america
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad	8	302	139	3570	12.8	78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
good	4	120	79	2625	18.6	82	america
bad	8	455	225	4425	10	70	america
good	4	107	86	2464	15.5	76	europe
bad	5	131	103	2830	15.9	78	europe

Split over Numeric Features

- Splits on numeric features use a threshold
- How to decide a threshold like 3535 in the diagram
- Consider a feature A. Sort the values of A in data
- Evaluate split thresholds in intervals between instances of different classes





ID3 algorithm summary

Ross Quinlan, 1986

The ID3 algorithm begins with the original set **S** as the root node.

For each iteration of the algorithm:

- Loop through every unused feature of the set S and calculates the information gain $IG(S)$ of that feature.

- Select the feature which has the largest information gain value, **best feature of splitting**

- S is then split by the **selected feature**, e.g. income, to produce subsets of the data.
- The algorithm continues to loop on each subset, **excluding** features used before.

Stopping Criteria

- All elements in the subset belong to the same class (Yes or No, 1 or 0, + or -), then the node is turned into a leaf node and labelled with the class of the examples
- No more features to be selected, while the examples still do not belong to the same class (some are 1 and some are 0), then the node is turned into a leaf node and labelled with the most common class of the examples in the subset
- No examples in the subset, for example if there is no example with age >= 100. Then a leaf node is created, and labelled with the most common class of the examples in the parent set.

What to do if...

In some leaf nodes there are no examples:

- Choose yes or no according to the number of yes/no examples at parent

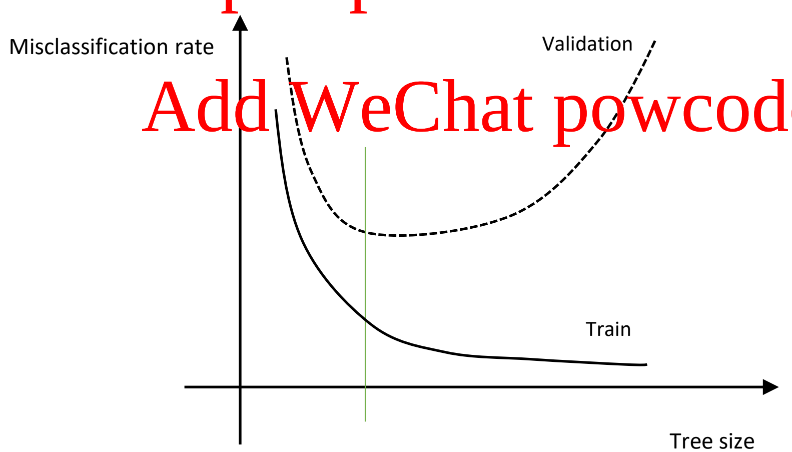
Some examples have the same features but different label: we have an error/noise

- Stop and use majority vote

In the applications of our unit, we focus more on decision tree with **binary** split. Also, scikit-learn uses an optimised version of the CART algorithm which constructs binary trees.

Overfitting in decision trees

- If we keep growing the tree until perfect classification for the training set we might over-fit
- For example, we can keep splitting the tree until each node contains 1 example
- This will fit perfectly on the training data, while NOT work on the new test data



Tree Pruning

Prepruning:

Stop growing when data split is not statistically significant. For example: stop tree construction when node size is smaller than a given limit, or impurity of a node is below a given limit. (faster)

Postpruning:

Grow the whole tree, then prune subtrees which overfit on the validation set. (more accurate)

How to Avoid Overfitting?

- **Prepruning :** stop splitting when there is no statistically significant:
  - Stop when Info-Gain (Gain-Ratio) is smaller than threshold
  - Stop when there are p, e.g. $p = 5$, examples in each leaf node
- **Postpruning:** grow the tree, then post-prune it based on validation set
- **Regularization:** penalize complex trees by minimizing with “complexity” = “# of leaf nodes”. $|T|$ indicates the number of leaf nodes of the tree T

Note: if tree grows, complexity grows, but entropy shrinks (uncertainty decreases).

$$\sum_{\text{All leaf nodes}} H(S_j) + \lambda * |T|$$

- Compute many trees on subsets of data and test: pick the best, or do prediction vote

- Random Forests are state of the art classifiers!

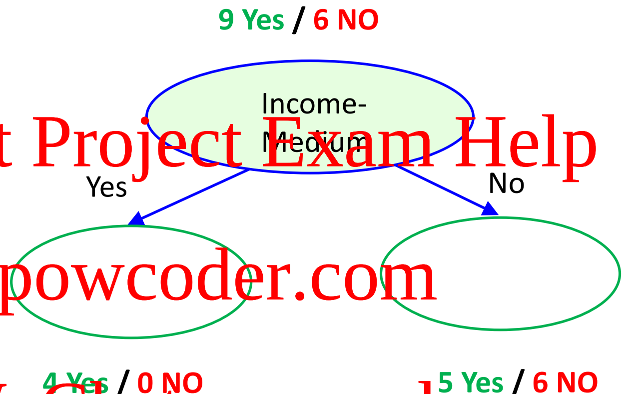
Python Example

In the real implementation, we transform the categorical features into dummy variables.

Index	Income_High	Income_Low	Income_Medium	Education_High school	Education_University	Marital Status_Married	Marital Status_Single	y
0	1	0	0	0	1	0	1	0
1	1	0	0	0	1	1	0	0
2	0	0	1	0	1	0	1	1
3	0	1	0	0	1	0	1	1
4	0	1	0	1	0	0	1	1
5	0	1	0	1	0	1	0	0
6	0	0	1	1	0	1	0	1
7	1	0	0	0	1	0	1	0
8	1	0	0	1	0	0	1	1
9	0	1	0	1	0	0	1	1
10	1	0	0	1	0	1	0	1
11	0	0	1	0	1	1	0	1
12	0	0	1	1	0	0	1	1
13	0	1	0	0	1	1	0	0
14	1	0	0	0	1	0	1	0

Used expected entropy as impurity measurement to select the best feature for 1st split (depth 1).

[0.82584103752696802,
0.97095059445466847,
0.72895548841643465,
0.78514543156506744,
0.78514543156506744,
0.95097750043269369,
0.95097750043269369]

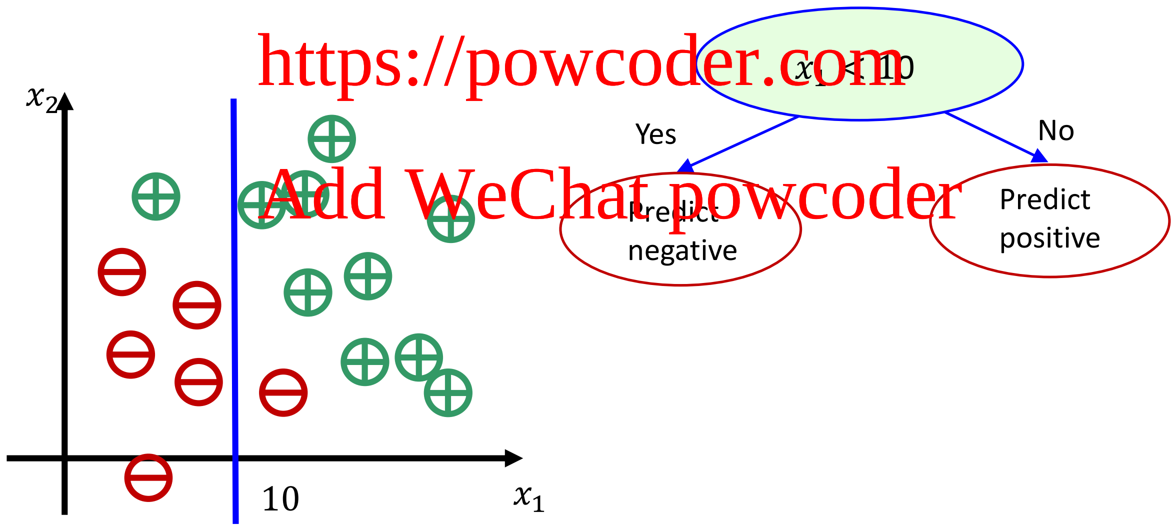


Income-Medium is selected as the best feature to split the root node.

Left node: [0. 4.] => [0 no, 4 yes]
Right node: [6. 5.] => [6 no, 5 yes]

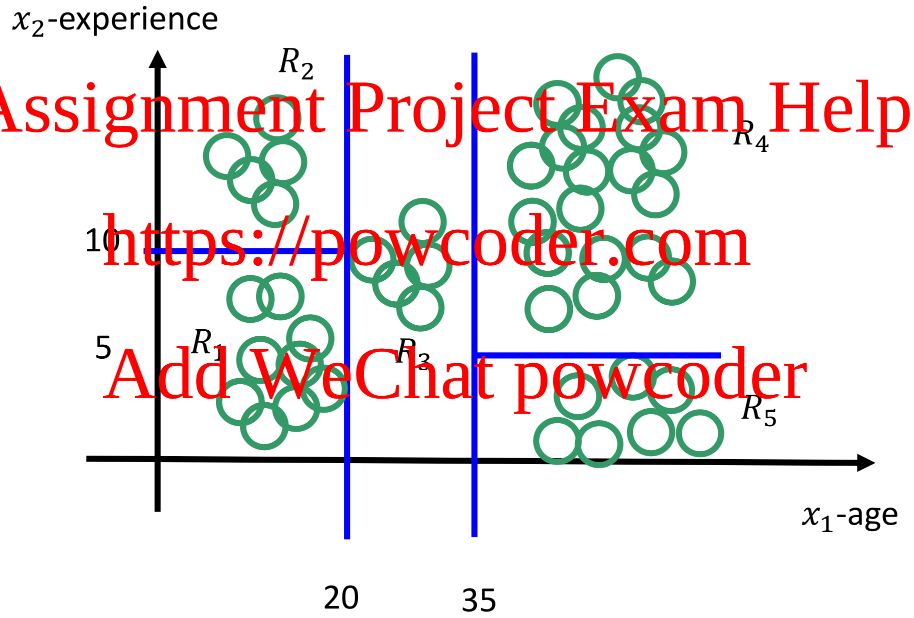
Decision Stump

- A decision stump is a decision tree consisting of only one-level.
- A decision tree with one root node which is immediately connected to the leaf nodes.
- We will use this concept to explain the boosting of the next lecture

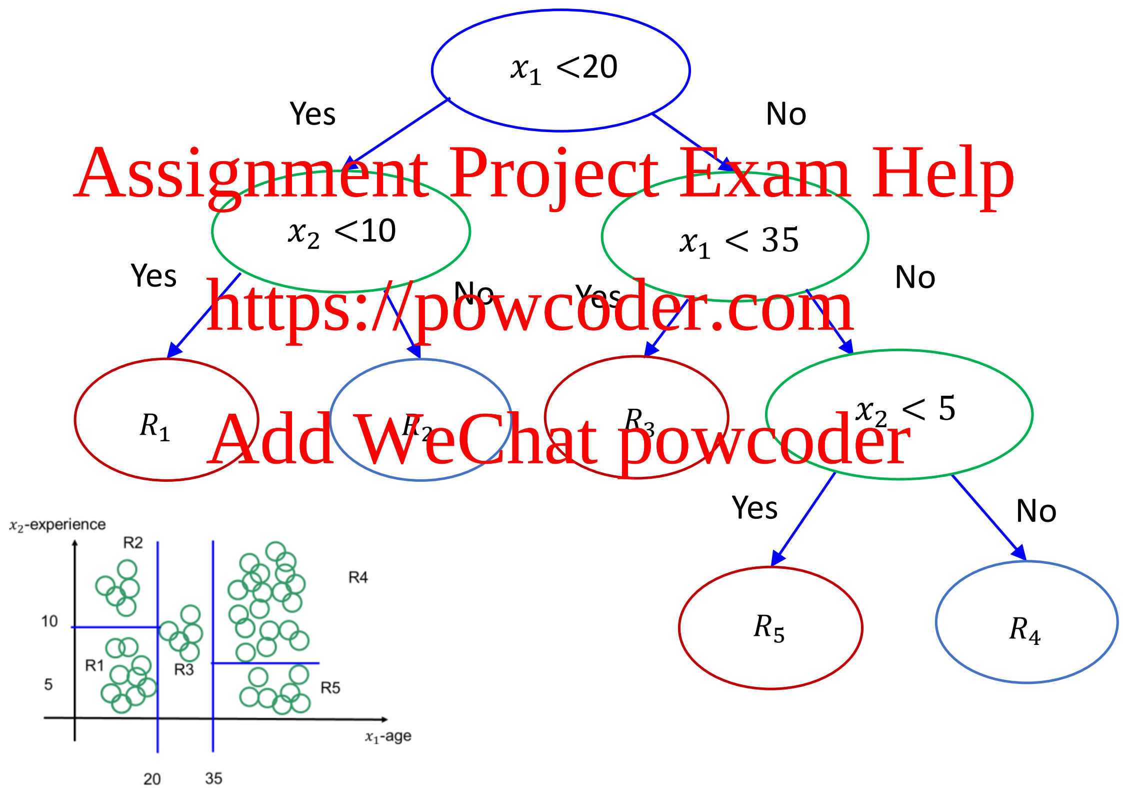


Regression Tree

Regression tree intuition



Regression tree Intuition



Building Regression Tree

Two steps of building a regression tree:

1. Partition the feature space: the set of possible values for $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$ into J distinct and non-overlapping regions for R_1, R_2, …, R_J

2. For a new observation that falls into the region R_j, we make the same prediction, which is simply the **mean** of the response values for the training examples in R_j

Random Forest

Random Forest introduction

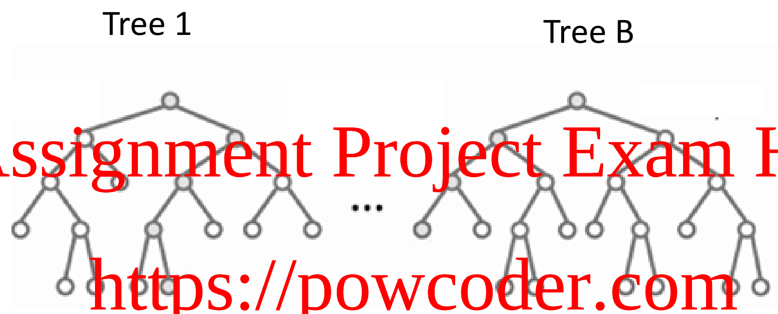
- **Random forest** (or **random forests**) is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees.

- The term came from random decision forests that was first proposed by Tin Kam Ho of Bell Labs in 1995.

- The method combines Breiman's "**bagging**" idea and the **random selection of features.**

- Random forests provide an improvement over bagged trees by way of a random small tweak that **decorrelates** the trees.

Random Forest introduction



- Random forests (RF) are a combination of tree predictors
- Each tree depends on the values of a random set sampled in dependently
- The generalization error depends on the strength of the individual trees and the correlation between them

Random Forest introduction

- Random forests provide an improvement over bagged trees by way of a random small tweak that decorrelates the trees
- In bagging, we build a number forest of decision trees on bootstrapped training samples
- Each time a split in a tree is considered, a random sample of p features is chosen as split candidates from the full set of d features
- **In RF, the number of features considered at each split is approximately equal to the square root of the total number of features**
- To avoid the situation that in bagging there is a quite strong feature, resulting most or all of the trees will use this strong predictor in the top split and produce very similar trees
- Random forests overcome this problem by forcing each split to consider only a subset of the features

RF Algorithm

1. For tree $b = 1$ to B:
   (a) Choose a bootstrap sample of size N from training data
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each leaf node of the tree, until the minimum node size is achieved:
      i. Select p variables at random from the d variables ($p \leq d$). **Why doing this?**
      ii. Pick the best variable/split-point among the p.
      iii. Split the node into two decision nodes.

2. Output the ensemble of trees $\{T_b\}_1^B$.

- Randomly select N observations (**with replacement**) from the original data set in order to produce a bootstrap data set

Friedman et al., (2001)

RF Prediction

To make a prediction at a new point $\mathbf{x_0}$:

For regression:

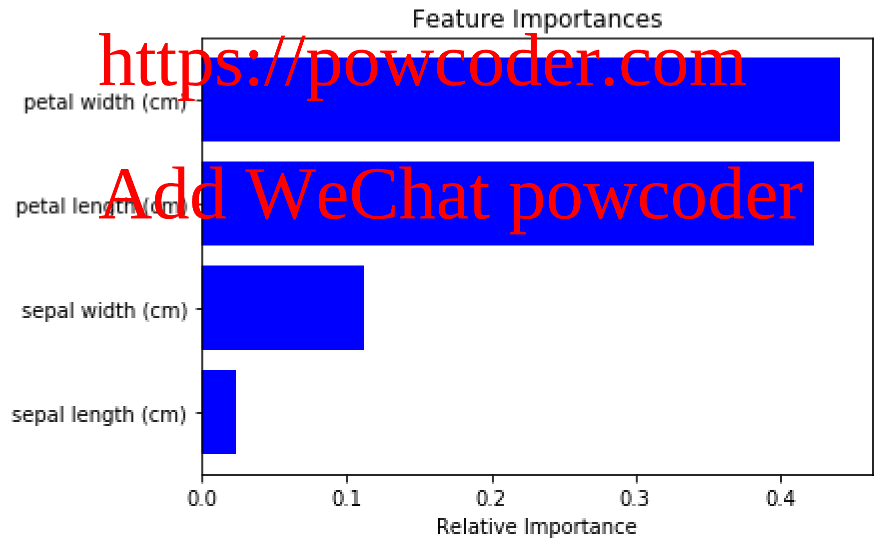
$$\hat{f}(\mathbf{x_0}) = \frac{1}{B}\sum_{b=1}^{B} T_b(\mathbf{x_0})$$

For classification:
Suppose the class prediction of the b_{th} random-forest tree is $C_b(\mathbf{x_0})$:

$$\hat{f}(\mathbf{x_0}) = \text{Mode}\{C_b(\mathbf{x_0})\}_{b=1}^{B}$$

Feature importance

At each split in each tree, the improvement in the split-criterion is the importance measure attributed to the splitting variable, and is accumulated over all the trees in the forest separately for each variable.



Review questions

- What are the intuitions of decision trees?
- How decision trees works? How to choose the feature to spit?
- What is decision stump?
- What is CART?
- What are the tree growing and pruning?
- How does ID3 algorithm work?
- What are Entropy and information gain?
- How does random forest work?