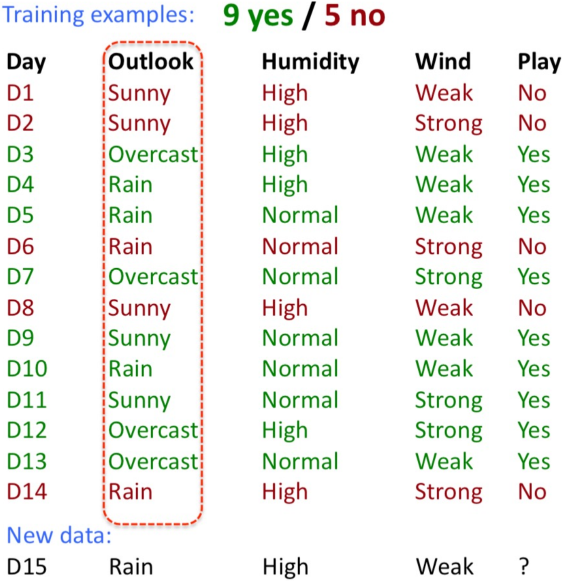
Decision Tree
Random Forest

Decision Tree Example



Training examples: **9 yes / 5 no**

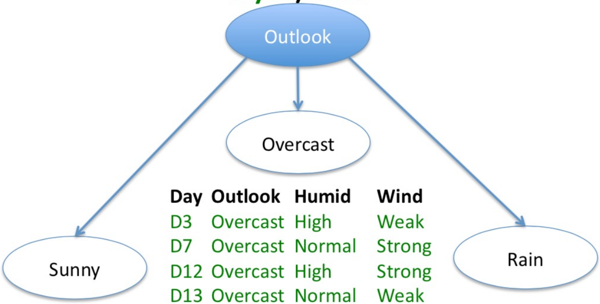
Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

New data:

D15	Rain	High	Weak	?

Decision Tree Example

9 yes / 5 no



Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

4 yes / 0 no

pure subset

Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

2 yes / 3 no

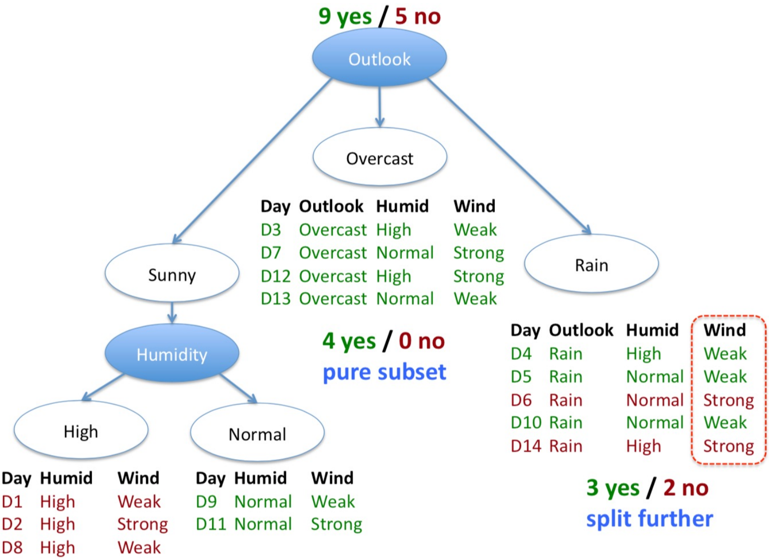
split further

Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

3 yes / 2 no

split further

Decision Tree Example



9 yes / 5 no

Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

4 yes / 0 no

pure subset

Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

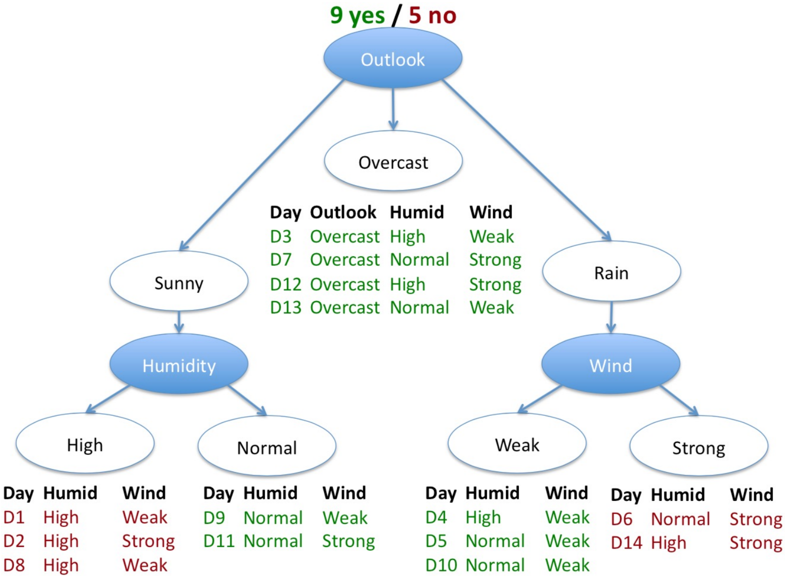
3 yes / 2 no

split further

Day	Humid	Wind
D1	High	Weak
D2	High	Strong
D8	High	Weak

Day	Humid	Wind
D9	Normal	Weak
D11	Normal	Strong

Decision Tree Example



Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

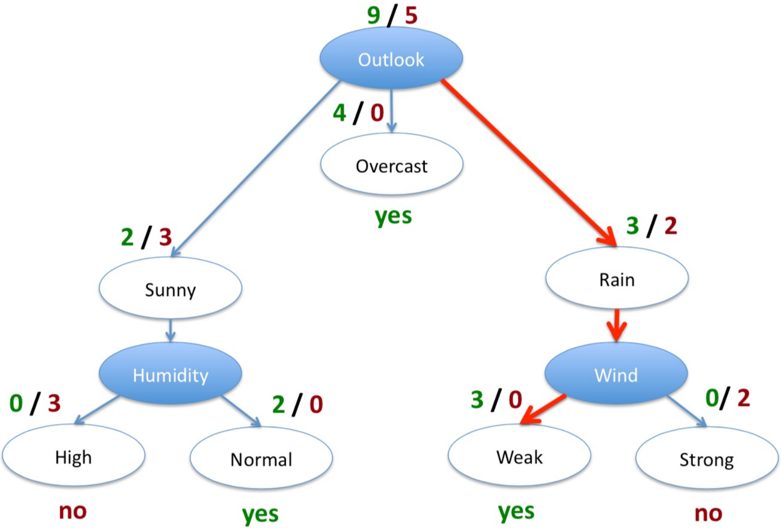
Day	Humid	Wind
D1	High	Weak
D2	High	Strong
D8	High	Weak

Day	Humid	Wind
D9	Normal	Weak
D11	Normal	Strong

Day	Humid	Wind
D4	High	Weak
D5	Normal	Weak
D10	Normal	Weak

Day	Humid	Wind
D6	Normal	Strong
D14	High	Strong

Decision Tree Example



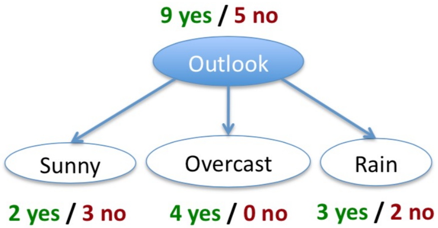
New data:

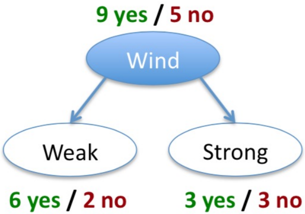
Day	Outlook	Humid	Wind	
D15	Rain	High	Weak	→ Yes

ID3 Algorithm

```
node = Root
examples = Training Set
Split (node, {examples} ):
1.  Find A, the best attribute for splitting the {examples}
2.  Create decision nodes for attribute A (i.e., child nodes
    of node)
3.  Split training {examples} to child nodes
4.  If examples perfectly classified (subset is pure): STOP
    else: iterate over new child nodes
          Split (child_node, {subset of examples} )
```

Find The Best Attribute





- Want to measure "purity" of the split
  - more certain about Yes/No after the split
    - pure set (4 yes / 0 no) => completely certain (100%)
    - impure (3 yes / 3 no) => completely uncertain (50%)
  - can't use P("yes" | set):
    - must be symmetric: 4 yes / 0 no as pure as 0 yes / 4 no

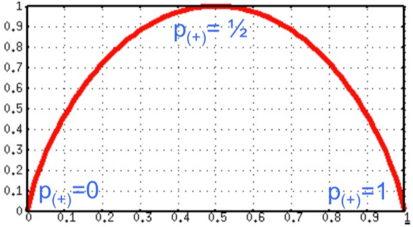
Entropy

- Entropy: $H(S) = - p_{(+)} \log_2 p_{(+)} - p_{(-)} \log_2 p_{(-)}$ bits
  - S … subset of training examples
  - $p_{(+)}$ / $p_{(-)}$ … % of positive / negative examples in S
- Interpretation: assume item X belongs to S
  - how many bits need to tell if X positive or negative
- impure (3 yes / 3 no):

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

- pure set (4 yes / 0 no):

$$H(S) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0 \text{ bits}$$



Information Gain

- Want many items in pure sets
- Expected drop in entropy after split:

$$Gain(S,A) = H(S) - \sum_{V \in Values(A)} \frac{|S_V|}{|S|} H(S_V)$$

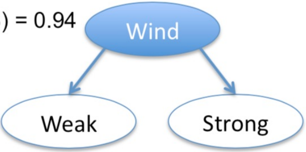
V … possible values of A
S … set of examples {X}
S_V … subset where X_A = V

- Mutual Information
  - between attribute A and class labels of S

Gain (S, Wind)
= H(S) – $^8/_{14}$ H(S_{weak}) – $^6/_{14}$ H(S_{Strong})
= 0.94 – $^8/_{14}$ * 0.81 – $^6/_{14}$ * 1.0
= 0.049

$-\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14}$ **9 yes / 5 no**

H(S) = 0.94



6 yes / 2 no

$-\frac{6}{8}\log_2\frac{6}{8} - \frac{2}{8}\log_2\frac{2}{8}$

H(S_{weak}) = 0.81

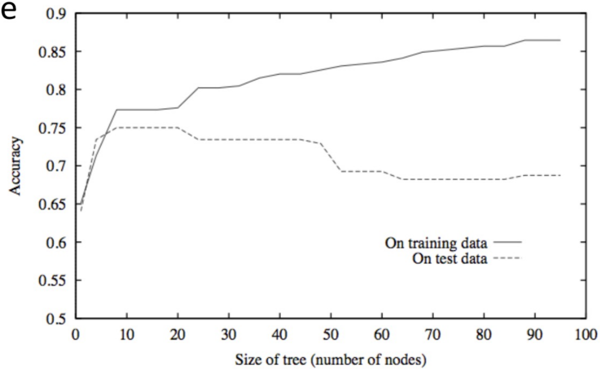
3 yes / 3 no

$-\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6}$

H(S_{strong}) = 1.0

Overfitting in Decision Tree

- Can always classify training examples perfectly
  - keep splitting until each node contains 1 example
  - singleton = pure
- Doesn’t work on new data



Avoid Overfitting

How can we avoid overfitting?

- Stop growing when data split is not statistically significant.
- Acquire more training data.
- Remove irrelevant attributes (manual process – not always possible)
- Grow full tree, then post-prune.

How to select "best" tree:

- Measure performance over training data
- Measure performance over separate validation data set
- Add complexity penalty to performance measure

Pre-pruning that stop growing the tree earlier, before it perfectly classifies the training set.
Post-pruning that allows the tree to perfectly classify the training set, and then post prune the tree.

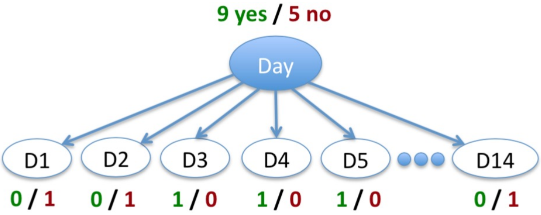
General Structure

Task: classification, discriminative

- Model structure: decision tree
- Score function
    - Information gain at each node
    - Preference for short trees
    - Preference for high-gain attributes near the root
- Optimization / search method
    - Greedy search from simple to complex
    - Guided by information gain

Problem With Information Gain

- Biased towards attributes with many values
- Won't work for new data: D15 Rain High Weak
- Use GainRatio:



all subsets perfectly pure => optimal split

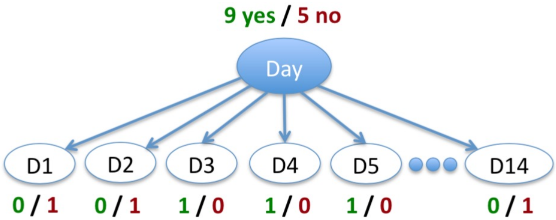
$$SplitEntropy(S,A) = -\sum_{V \in Values(A)} \frac{|S_V|}{|S|} \log \frac{|S_V|}{|S|}$$

A … candidate attribute
V … possible values of A
S … set of examples {X}
S_V … subset where $X_A = V$

$$GainRatio(S,A) = \frac{Gain(S,A)}{SplitEntropy(S,A)}$$

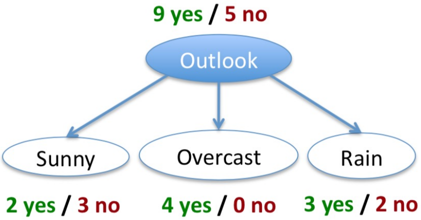
penalizes attributes with many values

Problem With Information Gain



$$SplitEntropy(S, Day)$$

$$= -14\left(\frac{1}{14}\,log\,\frac{1}{14}\right) = 3.80$$

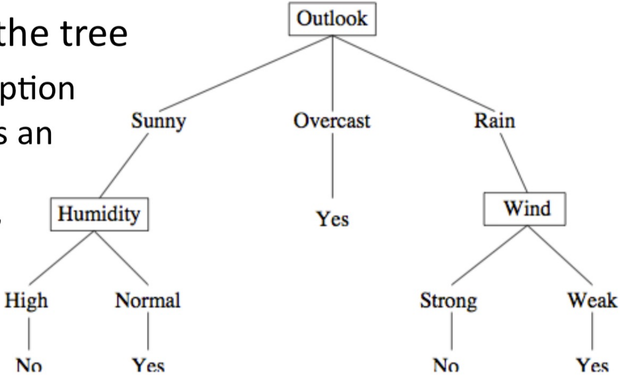


$$SplitEntropy(S, Outlook)$$

$$= -\frac{5}{14}\,log\,\frac{5}{14} - \frac{4}{14}\,log\,\frac{4}{14} - \frac{5}{14}\,log\,\frac{5}{14} = 1.57$$

Interpretation of Trees

- Read rules off the tree
  - concise description of what makes an item positive
- No "black box"
  - important for users

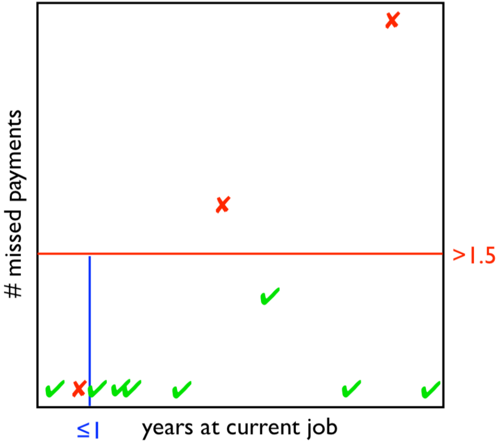


Rule: (Outlook = Overcast) ∨
(Outlook = Rain ∧ Wind = Weak) ∨
(Outlook = Sunny ∧ Humidity = Normal)

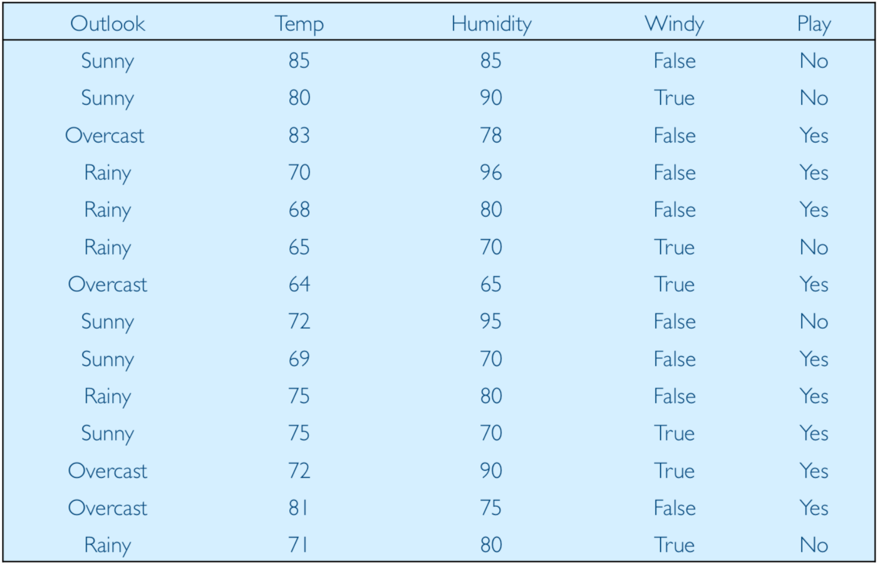
Continuous Variables

Predicting credit risk

years at current job	# missed payments	defaulted?
7	0	N
0.75	0	Y
3	0	N
9	0	N
4	2	Y
0.25	0	N
5	I	N
8	4	Y
I.0	0	N
I.75	0	N



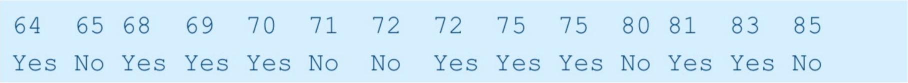
Continuous Variables



Outlook	Temp	Humidity	Windy	Play
Sunny	85	85	False	No
Sunny	80	90	True	No
Overcast	83	78	False	Yes
Rainy	70	96	False	Yes
Rainy	68	80	False	Yes
Rainy	65	70	True	No
Overcast	64	65	True	Yes
Sunny	72	95	False	No
Sunny	69	70	False	Yes
Rainy	75	80	False	Yes
Sunny	75	70	True	Yes
Overcast	72	90	True	Yes
Overcast	81	75	False	Yes
Rainy	71	80	True	No

Continuous Temperature Attribute

- First, sort the temperature values, including the class labels
- Then, check all the cut points and choose the one with the best information gain.



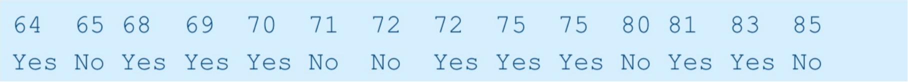
64	65	68	69	70	71	72	72	75	75	80	81	83	85
Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No

- Temp < 71.5: yes = 4, no = 2
- Temp ≥ 71.5: yes = 5, no = 3

How to compute:

- $H(Temp < 71.5) = ?$
- $H(Temp \geq 71.5) = ?$
- $H(S) = ?$
- $Gain(S, Temp) = ?$

Continuous Temperature Attribute



```
64   65  68   69   70   71   72   72   75   75   80  81   83   85
Yes  No  Yes  Yes  Yes  No   No   Yes  Yes  Yes  No  Yes  Yes  No
```

Temp < 71.5: yes = 4, no = 2

Temp ≥ 71.5: yes = 5, no = 3

$$H(Temp < 71.5) = -\frac{4}{6}\log\frac{4}{6} - \frac{2}{6}\log\frac{2}{6} = 0.918$$

$$H(Temp \geq 71.5) = -\frac{5}{8}\log\frac{5}{8} - \frac{3}{8}\log\frac{3}{8} = 0.954$$

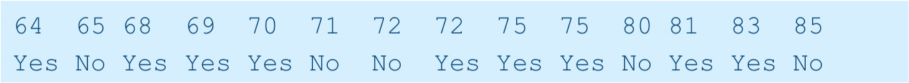
$$H(S) = -\frac{9}{14}\log\frac{9}{14} - \frac{5}{14}\log\frac{5}{14} = 0.940$$

$$Gain(S, Temp) = H(S) - \frac{6}{14}H(Temp < 71.5) - \frac{8}{14}H(Temp \geq 71.5)$$

$$= 0.940 - \frac{6}{14} * 0.918 - \frac{8}{14} * 0.954 = 0.001$$

Continuous Temperature Attribute

- First, sort the temperature values, including the class labels
- Then, check all the cut points and choose the one with the best information gain.

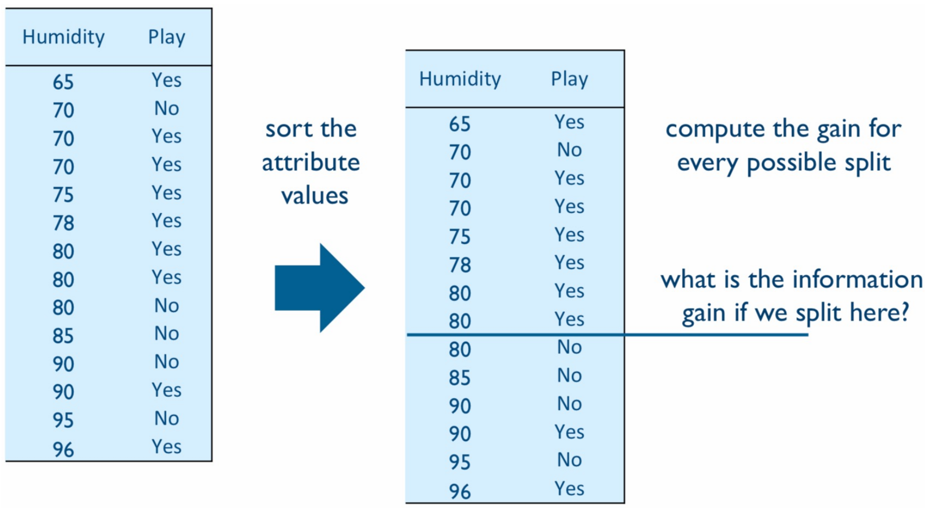


```
64   65  68   69   70   71   72   72   75   75   80  81   83   85
Yes  No  Yes  Yes  Yes  No   No   Yes  Yes  Yes  No  Yes  Yes  No
```

- Temp < 71.5: yes = 4, no = 2
- Temp ≥ 71.5: yes = 5, no = 3

- Place split points **halfway** between values.
- Able to evaluate all split points in **one pass.**

Information Gain for Humidity



Humidity	Play
65	Yes
70	No
70	Yes
70	Yes
75	Yes
78	Yes
80	Yes
80	Yes
80	No
85	No
90	No
90	Yes
95	No
96	Yes

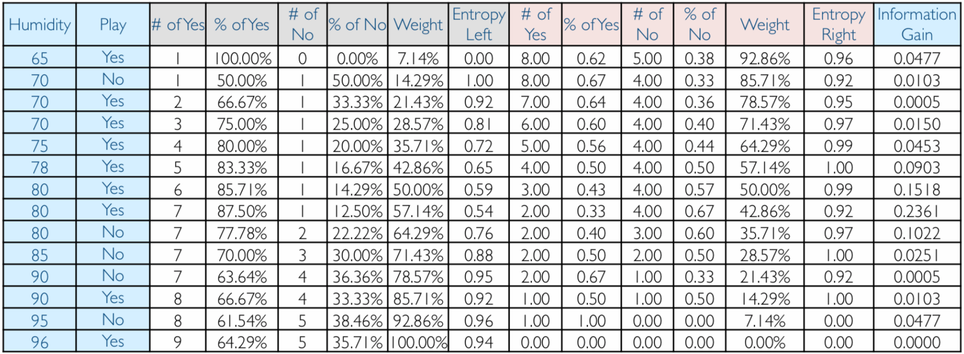
sort the attribute values

Humidity	Play
65	Yes
70	No
70	Yes
70	Yes
75	Yes
78	Yes
80	Yes
80	Yes
80	No
85	No
90	No
90	Yes
95	No
96	Yes

compute the gain for every possible split

what is the information gain if we split here?

Information Gain for Humidity



Humidity	Play	# of Yes	% of Yes	# of No	% of No	Weight	Entropy Left	# of Yes	% of Yes	# of No	% of No	Weight	Entropy Right	Information Gain
65	Yes	1	100.00%	0	0.00%	7.14%	0.00	8.00	0.62	5.00	0.38	92.86%	0.96	0.0477
70	No	1	50.00%	1	50.00%	14.29%	1.00	8.00	0.67	4.00	0.33	85.71%	0.92	0.0103
70	Yes	2	66.67%	1	33.33%	21.43%	0.92	7.00	0.64	4.00	0.36	78.57%	0.95	0.0005
70	Yes	3	75.00%	1	25.00%	28.57%	0.81	6.00	0.60	4.00	0.40	71.43%	0.97	0.0150
75	Yes	4	80.00%	1	20.00%	35.71%	0.72	5.00	0.56	4.00	0.44	64.29%	0.99	0.0453
78	Yes	5	83.33%	1	16.67%	42.86%	0.65	4.00	0.50	4.00	0.50	57.14%	1.00	0.0903
80	Yes	6	85.71%	1	14.29%	50.00%	0.59	3.00	0.43	4.00	0.57	50.00%	0.99	0.1518
80	Yes	7	87.50%	1	12.50%	57.14%	0.54	2.00	0.33	4.00	0.67	42.86%	0.92	0.2361
80	No	7	77.78%	2	22.22%	64.29%	0.76	2.00	0.40	3.00	0.60	35.71%	0.97	0.1022
85	No	7	70.00%	3	30.00%	71.43%	0.88	2.00	0.50	2.00	0.50	28.57%	1.00	0.0251
90	No	7	63.64%	4	36.36%	78.57%	0.95	2.00	0.67	1.00	0.33	21.43%	0.92	0.0005
90	Yes	8	66.67%	4	33.33%	85.71%	0.92	1.00	0.50	1.00	0.50	14.29%	1.00	0.0103
95	No	8	61.54%	5	38.46%	92.86%	0.96	1.00	1.00	0.00	0.00	7.14%	0.00	0.0477
96	Yes	9	64.29%	5	35.71%	100.00%	0.94	0.00	0.00	0.00	0.00	0.00%	0.00	0.0000

Left branch, e.g., < 67.5

Right branch, e.g., ≥ 67.5

Multi-class Classification

Multiclass classification:

- Predict the **best attribute** to split on.
- Entropy:

$$\boldsymbol{H(S) = -\sum_c p_c\, log_2(p_c).}$$

- $\mathrm{p_c}$: % of examples of class **c** in S.

Random (Decision) Forest

Training: grow **K** different decision trees:

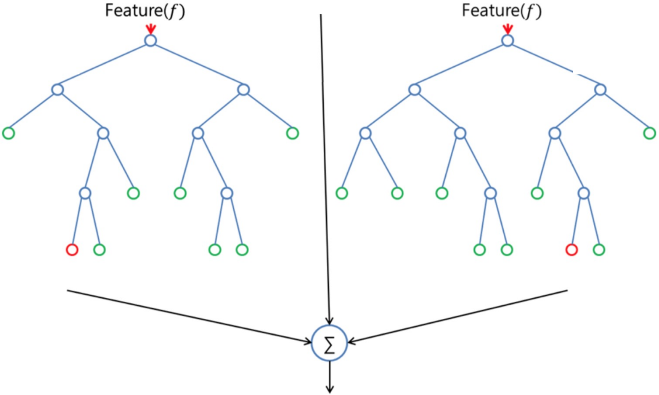
- Pick a **random subset $\mathbf{S}_{\mathbf{random}}$** of training examples.
- Grow a full decision tree (no pruning), compute information gain based on $\mathbf{S}_{\mathbf{random}}$ instead of full set.
- Repeat for **K decision trees.**

Inference: given a new data point X:

- Classify X using each of the K trees.
- Use majority vote: class predicted most often.

Fast, scalable, state-of-the-art performance.

Random (Decision) Forest



Random forest with majority voting

Gini impurity – An alternative of Entropy

Gini impurity (or **Gini index**) of a dataset D indicates the likelihood of new, random data sample being misclassified if the sample is given a random class label according to the class distribution in the dataset.

$$\boxed{Gini(D) = \sum_{i=1}^{k} p_i(1 - p_i) = 1 - \sum_{i=1}^{k} p_i^2}$$

Probability of the sample belonging to class *i*

Probability of the sample **not** being classified to class *i* (i.e., misclassified).

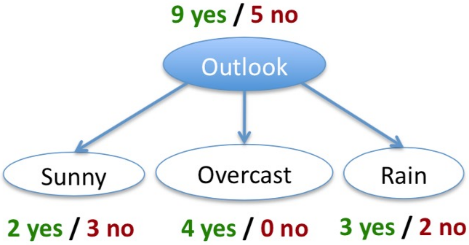
Gini impurity of a dataset

Gini impurity (or **Gini index**) of a dataset D indicates the likelihood of new, random data sample being misclassified if the sample is given a random class label according to the class distribution in the dataset.

$$\boxed{Gini(D) = \sum_{i=1}^{k} p_i(1 - p_i) = 1 - \sum_{i=1}^{k} p_i^2}$$

Probability of the sample belonging to any class is given by $\sum_{i=1}^{k} p_i = 1$

Gini impurity of a dataset



9 yes / 5 no

Outlook

Sunny

Overcast

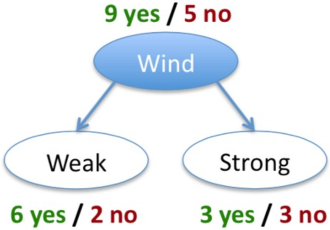
Rain

2 yes / 3 no

4 yes / 0 no

3 yes / 2 no

$Gini(Sunny) = 0.48$ $Gini(Overcast) = 0$ $Gini(Rain) = 0.48$



9 yes / 5 no

Wind

Weak

Strong

6 yes / 2 no

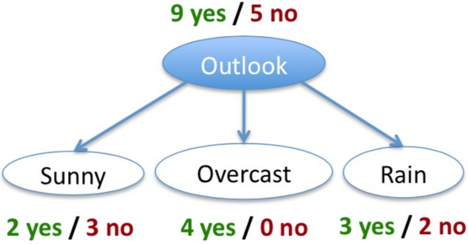
3 yes / 3 no

$Gini(Weak) = 0.375$ $Gini(Strong) = 0.5$

$$Gini(Sunny) = \frac{2}{5}\left(1 - \frac{2}{5}\right) + \frac{3}{5}\left(1 - \frac{3}{5}\right) = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = 0.48$$

$$Gini(Weak) = \frac{6}{8}\left(1 - \frac{6}{8}\right) + \frac{2}{8}\left(1 - \frac{2}{8}\right) = 1 - \left(\frac{6}{8}\right)^2 - \left(\frac{2}{8}\right)^2 = 0.375$$

Gini impurity of a dataset



9 yes / 5 no

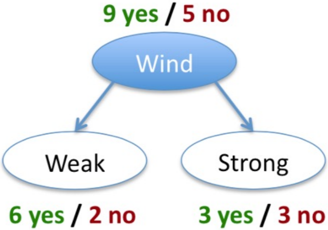
Outlook

Sunny — 2 yes / 3 no

Overcast — 4 yes / 0 no

Rain — 3 yes / 2 no

$Gini(Sunny) = 0.48$ $Gini(Overcast) = 0$ $Gini(Rain) = 0.48$



9 yes / 5 no

Wind

Weak — 6 yes / 2 no

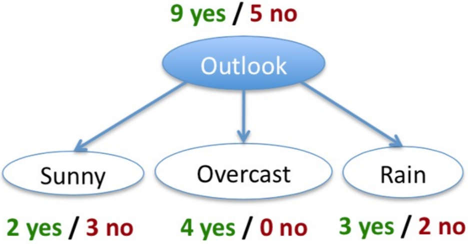
Strong — 3 yes / 3 no

$Gini(Weak) = 0.375$ $Gini(Strong) = 0.5$

In multi-class classification problem, Gini impurity varies between values 0 and 1 (take an example of 10 data samples and 5 classes):

- Gini = 0: **pure** dataset (e.g., 0/10/0/0/0).
- Gini = 0.5: **equal** distribution over *some* classes (e.g., 5/0/5/0/0).
- Gini = 1: **random** distribution across *all* classes (e.g., 1/2/3/2/2).

Gini impurity of an attribute



9 yes / 5 no

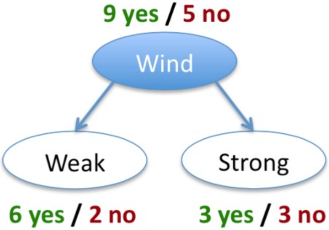
Outlook

Sunny — 2 yes / 3 no

Overcast — 4 yes / 0 no

Rain — 3 yes / 2 no

$Gini(Sunny) = 0.48$ $Gini(Overcast) = 0$ $Gini(Rain) = 0.48$



9 yes / 5 no

Wind

Weak — 6 yes / 2 no

Strong — 3 yes / 3 no

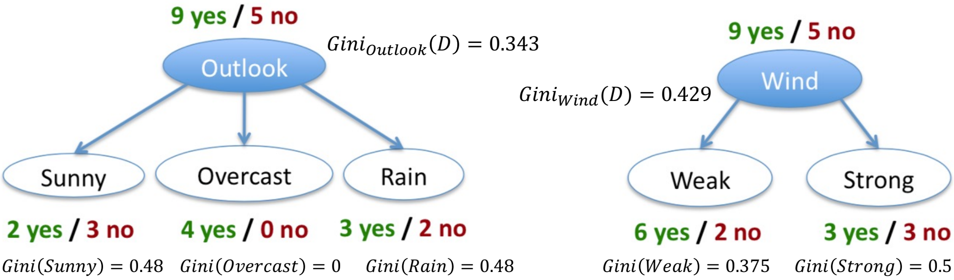
$Gini(Weak) = 0.375$ $Gini(Strong) = 0.5$

If a dataset D is split on an attribute A into m subsets {D_1, …, D_m}, the Gini impurity can be defined as:

$$\boxed{Gini_A(D) = \sum_{s=1}^{m} \frac{|D_s|}{|D|} Gini(D_s)}$$

Attribute **minimizing the Gini impurity** is chosen to split the node.

Gini impurity of an attribute



$Gini_{Outlook}(D) = 0.343$

$Gini(Sunny) = 0.48$ $Gini(Overcast) = 0$ $Gini(Rain) = 0.48$

$Gini_{Wind}(D) = 0.429$

$Gini(Weak) = 0.375$ $Gini(Strong) = 0.5$

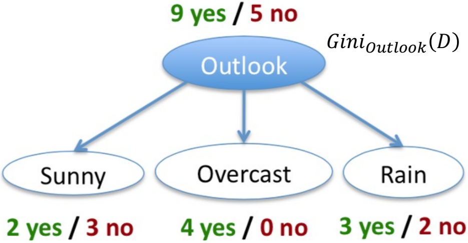
$$Gini_{Outlook}(D) = \frac{5}{14}0.48 + \frac{4}{14}0 + \frac{5}{14}0.48 = 0.343$$

$$Gini_{Wind}(D) = \frac{8}{14}0.375 + \frac{6}{14}0.5 = 0.429$$

Outlook is chosen!

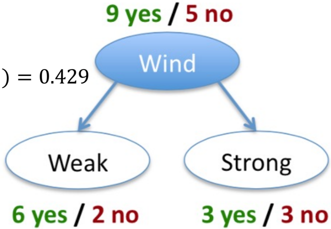
Gini information gain

$$Gini(D) = \frac{9}{14}\left(1-\frac{9}{14}\right)+\frac{5}{14}\left(1-\frac{5}{14}\right) = 0.459$$



$Gini_{Outlook}(D) = 0.343$

$Gini(Sunny) = 0.48$ $Gini(Overcast) = 0$ $Gini(Rain) = 0.48$



$Gini_{Wind}(D) = 0.429$

$Gini(Weak) = 0.375$ $Gini(Strong) = 0.5$

Gini information gain for an attribute (A) is the weighted impurities of the branches is subtracted from the original impurity.

$$\boxed{\Delta Gini(A) = Gini(D) - Gini_A(D)}$$

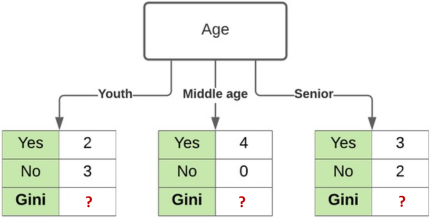
Attribute **maximizing the Gini information gain** is chosen to split.

$$\Delta Gini(Outlook) = 0.459 - 0.343 = 0.116$$

Outlook is chosen!

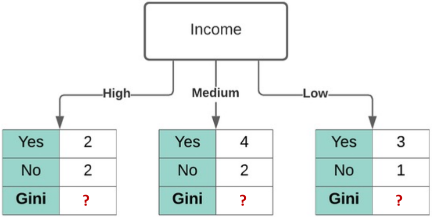
$$\Delta Gini(\mathrm{Wind}) = 0.459 - 0.429 = 0.03$$

Gini impurity exercise



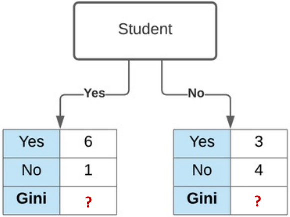
	Youth	Middle age	Senior
Yes	2	4	3
No	3	0	2
Gini	?	?	?

Gini Impurity for Age is ?



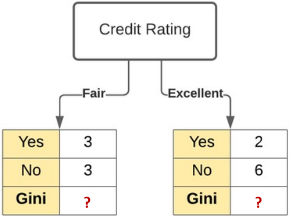
	High	Medium	Low
Yes	2	4	3
No	2	2	1
Gini	?	?	?

Gini Impurity for Income is ?



	Yes	No
Yes	6	3
No	1	4
Gini	?	?

Gini Impurity for Student is ?

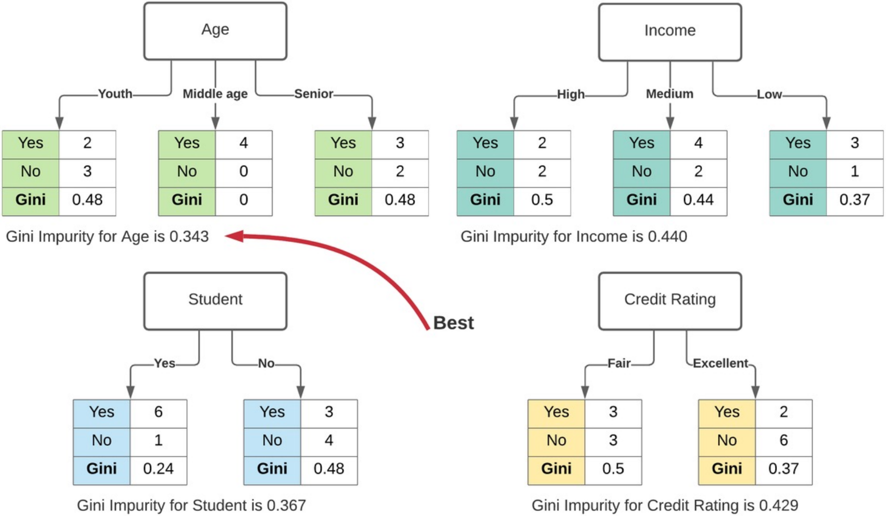


	Fair	Excellent
Yes	3	2
No	3	6
Gini	?	?

Gini Impurity for Credit Rating is ?

What is the best attribute to split?

Gini impurity exercise



Age

Youth

Yes	2
No	3
Gini	0.48

Middle age

Yes	4
No	0
Gini	0

Senior

Yes	3
No	2
Gini	0.48

Gini Impurity for Age is 0.343

Income

High

Yes	2
No	2
Gini	0.5

Medium

Yes	4
No	2
Gini	0.44

Low

Yes	3
No	1
Gini	0.37

Gini Impurity for Income is 0.440

Best

Student

Yes

Yes	6
No	1
Gini	0.24

No

Yes	3
No	4
Gini	0.48

Gini Impurity for Student is 0.367

Credit Rating

Fair

Yes	3
No	3
Gini	0.5

Excellent

Yes	2
No	6
Gini	0.37

Gini Impurity for Credit Rating is 0.429

Regression tree

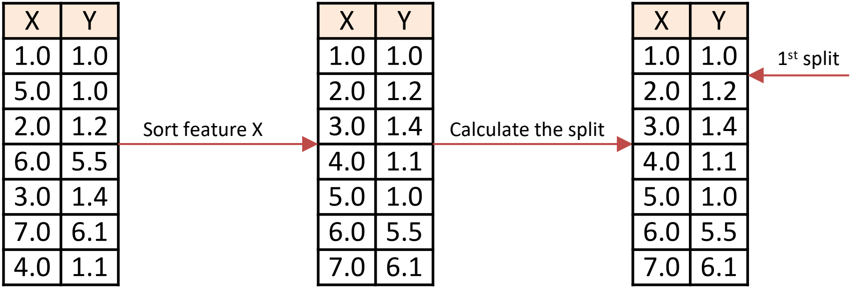
What if the label is not categorial but numeric?

There are two types of decision tree:

- Classification tree: is used when the label is categorical (discrete values).
- Regression tree: is used when the label is numeric (continuous values)

X	Y
1.0	1.0
5.0	1.0
2.0	1.2
6.0	5.5
3.0	1.4
7.0	6.1
4.0	1.1

Regression tree



X	Y
1.0	1.0
5.0	1.0
2.0	1.2
6.0	5.5
3.0	1.4
7.0	6.1
4.0	1.1

Sort feature X

X	Y
1.0	1.0
2.0	1.2
3.0	1.4
4.0	1.1
5.0	1.0
6.0	5.5
7.0	6.1

Calculate the split

X	Y
1.0	1.0
2.0	1.2
3.0	1.4
4.0	1.1
5.0	1.0
6.0	5.5
7.0	6.1

1st split

Idea: find the best point to split the dataset into 2 parts so that the Mean Squared Error (MSE) is minimized at that point.

Steps:

- **Sort** the data according to its features (n data samples in total).
- **Brute-force** all possible split points, calculate corresponding MSE values.
- After have n-1 MSE calculated, **choose** the split/threshold of minimum MSE.

Regression tree

X	Y
1.0	1.0
5.0	1.0
2.0	1.2
6.0	5.5
3.0	1.4
7.0	6.1
4.0	1.1

Sort feature X

X	Y
1.0	1.0
2.0	1.2
3.0	1.4
4.0	1.1
5.0	1.0
6.0	5.5
7.0	6.1

Calculate the split

X	Y
1.0	1.0
2.0	1.2
3.0	1.4
4.0	1.1
5.0	1.0
6.0	5.5
7.0	6.1

1st split

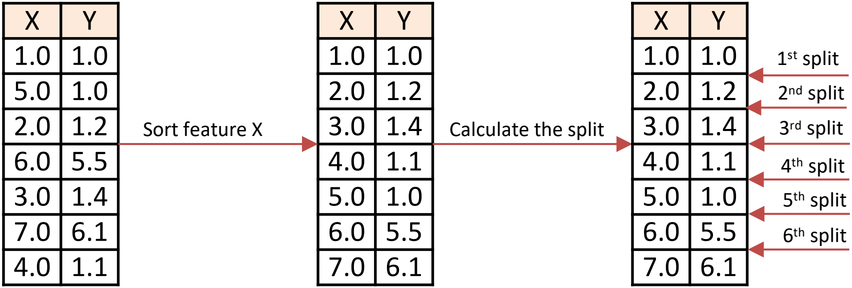
1st split at value (1+2)/2 = 1.5. There are two groups:

- Group 1 ($X < 1.5$): {(1.0,1.0)}. Average Y value is 1.0 ($\hat{y}_{left}$).
- Group 2 ($X \geq 1.5$): all other points. Average Y value is 2.72 ($\hat{y}_{right}$).

Calculate the MSE of the 1st split by (given $n=n_{left}+n_{right}$):

$$\mathrm{MSE}(1^{\mathrm{st}}\,\mathrm{split}) = \mathrm{MSE}(1^{\mathrm{st}}\,\mathrm{left}) + \mathrm{MSE}(1^{\mathrm{st}}\,\mathrm{right}) = \frac{1}{n}\left(\sum_{i=1}^{n_{left}} (y_i - \hat{y}_{left})^2 + \sum_{j=1}^{n_{right}} (y_j - \hat{y}_{right})^2\right)$$

Regression tree



X	Y
1.0	1.0
5.0	1.0
2.0	1.2
6.0	5.5
3.0	1.4
7.0	6.1
4.0	1.1

Sort feature X

X	Y
1.0	1.0
2.0	1.2
3.0	1.4
4.0	1.1
5.0	1.0
6.0	5.5
7.0	6.1

Calculate the split

X	Y
1.0	1.0
2.0	1.2
3.0	1.4
4.0	1.1
5.0	1.0
6.0	5.5
7.0	6.1

1st split

2nd split

3rd split

4th split

5th split

6th split

Similarly calculate MSE of all other possible splits. The chosen split/threshold is the one **having minimum MSE**.

Feature importance - A single decision tree

Idea: calculate a score representing the “importance” of each data feature. The **higher the score**, the **larger the effect** of that feature on the model prediction.

$$\boxed{I_i = \sum_{i_n=i} \left(p(n) purity(n) - \sum_{n_{child}} p(n_{child}) purity(n_{child}) \right)}$$

Importance of feature i.

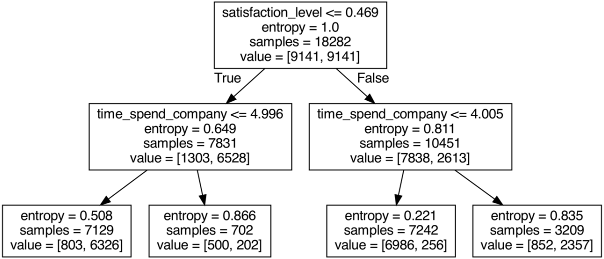
Sum over all nodes of the tree that use feature i.

Probability of using of a node for a sample data point.

Purity of a node, such as Entropy, Gini, Squared Error.

Feature importance – Example with Entropy

Assume that this is the tree after training


satisfaction_level <= 0.469
entropy = 1.0
samples = 18282
value = [9141, 9141]

True

False

time_spend_company <= 4.996
entropy = 0.649
samples = 7831
value = [1303, 6528]

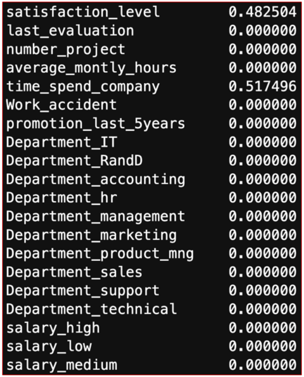
time_spend_company <= 4.005
entropy = 0.811
samples = 10451
value = [7838, 2613]

entropy = 0.508
samples = 7129
value = [803, 6326]

entropy = 0.866
samples = 702
value = [500, 202]

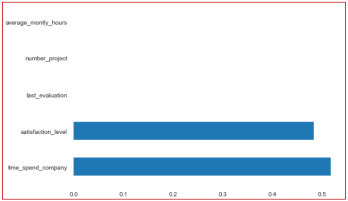
entropy = 0.221
samples = 7242
value = [6986, 256]

entropy = 0.835
samples = 3209
value = [852, 2357]


```
satisfaction_level        0.482504
last_evaluation           0.000000
number_project            0.000000
average_montly_hours      0.000000
time_spend_company        0.517496
Work_accident             0.000000
promotion_last_5years     0.000000
Department_IT             0.000000
Department_RandD          0.000000
Department_accounting     0.000000
Department_hr             0.000000
Department_management     0.000000
Department_marketing      0.000000
Department_product_mng    0.000000
Department_sales          0.000000
Department_support        0.000000
Department_technical      0.000000
salary_high               0.000000
salary_low                0.000000
salary_medium             0.000000
```

satisfaction_level = 18282/18282*1.0-7831/18282*0.649-10451/18282*0.811
= 0.2584

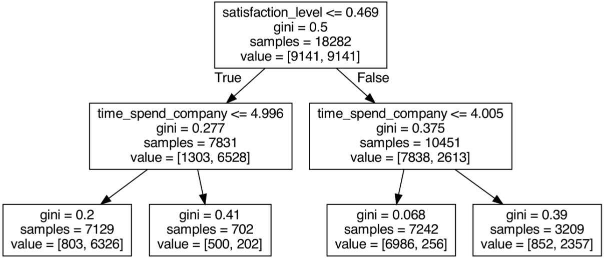
time_spend_company
= (7831/18282*0.649-7129/18282*0.508-702/18282*0.866)
+ (10451/18282*0.811-7242/18282*0.221-3209/18282*0.835)
= 0.0466 + 0.2295 = 0.2761



Then normalize the sum value to 1, we have:
satisfaction_level = 0.2584/(0.2584+0.2761) = **0.4834**
time_spend_company = 0.2761/(0.2584+0.2761) = **0.5166**

Feature importance – Example with Gini

Assume that this is the tree after training

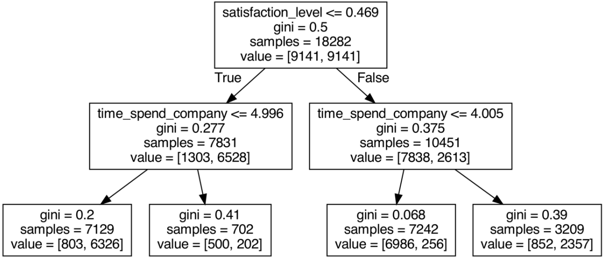


What is the feature importance score of

- satisfaction_level = ?
- time_spend_company = ?

Feature importance – Example with Gini

Assume that this is the tree after training



satisfaction_level <= 0.469
gini = 0.5
samples = 18282
value = [9141, 9141]

True

False

time_spend_company <= 4.996
gini = 0.277
samples = 7831
value = [1303, 6528]

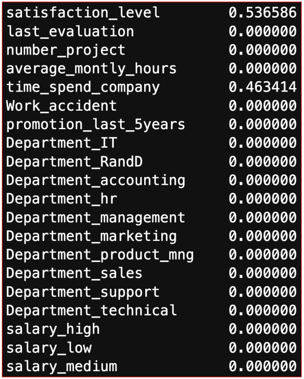
time_spend_company <= 4.005
gini = 0.375
samples = 10451
value = [7838, 2613]

gini = 0.2
samples = 7129
value = [803, 6326]

gini = 0.41
samples = 702
value = [500, 202]

gini = 0.068
samples = 7242
value = [6986, 256]

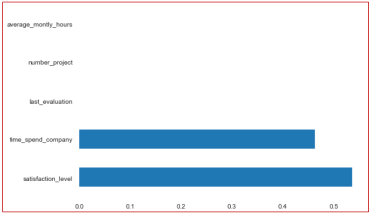
gini = 0.39
samples = 3209
value = [852, 2357]



```
satisfaction_level        0.536586
last_evaluation           0.000000
number_project            0.000000
average_montly_hours      0.000000
time_spend_company        0.463414
Work_accident             0.000000
promotion_last_5years     0.000000
Department_IT             0.000000
Department_RandD          0.000000
Department_accounting     0.000000
Department_hr             0.000000
Department_management     0.000000
Department_marketing      0.000000
Department_product_mng    0.000000
Department_sales          0.000000
Department_support        0.000000
Department_technical      0.000000
salary_high               0.000000
salary_low                0.000000
salary_medium             0.000000
```

satisfaction_level = 18282/18282*0.5-7831/18282*0.277-10451/18282*0.375
= 0.1670

time_spend_company
= (7831/18282*0.277-7129/18282*0.2-702/18282*0.41)
+ (10451/18282*0.375-7242/18282*0.068-3209/18282*0.39)
= 0.0250 + 0.1190 = 0.1440



Then normalize the sum value to 1, we have:
satisfaction_level = 0.1670/(0.1670+0.1440) = **0.5369**
time_spend_company = 0.1440/(0.1670+0.1440) = **0.4631**

Feature importance – Random forest

Idea: importance of a feature in a random forest is the sum of importance scores of that feature in all trees in the random forest.

$$\boxed{I_i = \frac{1}{|B|} \sum_{T \in B} I_i(T)}$$

Importance of feature i.

Total number of trees in random the forest.

Importance of feature i for a single tree T

Summary

- Decision tree.
- Entropy, information gain, gain ratio.
- Continuous variable in decision tree.
- Random forest.
- Gini impurity of a dataset/attribute, Gini information gain.
- Regression tree.
- Feature importance.

Q&A

Thank you