Measures of Node Impurity

- Gini Index

$$Gini(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2$$

$p(i|t)$: the probability that an instance t in D_t belongs to class C_i, estimated by $\frac{n_i}{|D_t|}$, where n_i is the number of instances with class label C_i.

- Entropy

$$Entropy(t) = - \sum_{i=0}^{c-1} p(i|t) log_2 p(i|t)$$

Log uses base 2: information is encoded in bits.

- Classification error

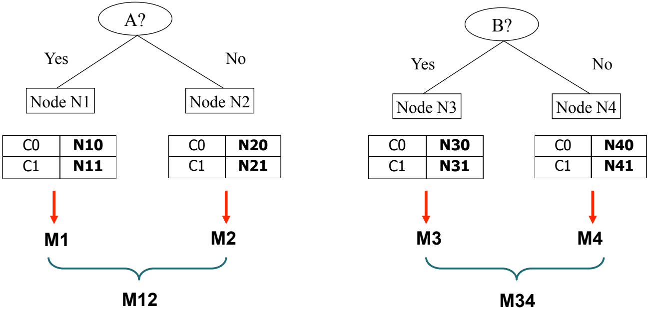
$$Classification\ error(t) = 1 - max_i[p(i|t)]$$

Determine the Best Split– Information Gain

Before splitting:

C0	N00
C1	N01

$M0 = I(N)$



- Gain: $\Delta = I(parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$
- $Gain = M0 - M12 \ vs \ \ M0 - M34$

Measures of Node Impurity – Gini

$$Gini(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2$$

$p(i|t)$: the relative frequency of class i at node t.

- Maximum? $(1 - \frac{1}{n_c})$ when records are equally distributed among all classes, implying least interesting information
- Minimum? (0.0) when all records belong to one class, implying most interesting information
- First used in CART, which allows only binary splitting

Measures of Node Impurity – Gini

C1	0
C2	6

C1	1
C2	5

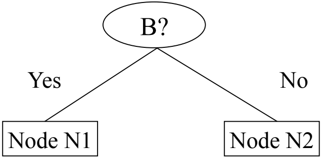
C1	2
C2	4

C1	3
C2	3

Gini?

- $1 - (\frac{0}{6})^2 - (\frac{6}{6})^2 = 0$
- $1 - (\frac{1}{6})^2 - (\frac{5}{6})^2 = \frac{10}{36} = 0.278$
- $1 - (\frac{2}{6})^2 - (\frac{4}{6})^2 = \frac{16}{36} = 0.444$
- 0.5

Binary Attributes: Computing Gini Index



	Parent	N1	N2
C1	6	5	1
C2	6	2	4

- $Gini(Parent) = 0.5$
- $Gini(N1) = 1 - (5/7)^2 - (2/7)^2$
- $Gini(N2) = 1 - (1/5)^2 - (4/5)^2$
- $Gini(Children) = 7/12 * Gini(N1) + 5/12 * Gini(N2)$
- $Gain = Gini(Parent) - Gini(Children)$

Categorical Attributes: Computing Gini Index

- For each distinct value, gather counts for each class in the dataset
- Use the count matrix to make decisions

Multi-way split

	CarType		
	Family	Sports	Luxury
C1	1	2	1
C2	4	1	1
Gini	0.393		

Two-way split
(find best partition of values)

	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	CarType	
	{Sports}	{Family, Luxury}
C1	2	2
C2	1	5
Gini	0.419	

Calculation example?

Categorical Attributes: Computing Gini Index

- Gini(family) = 1-(1/25)-(16/25)=(8/25)
- Gini(sports) = 1-(4/9)-(1/9)=(4/9)
- Gini(luxury) = 1/2
- Gini(all) = (5/10)*(8/25)+(3/10)*(4/9)+(2/10)*(1/2) =(4/25)+(2/15)+(1/10) = (24+20+15)/150 = 59/150 = 0.393

Continuous Attributes: Computing Gini Index

- Use binary decisions based on one value
- Several choices for the splitting value
  - Number of possible splitting values = Number of distinct values
- Each splitting value has a count matrix associated with it
  - Class counts in each of the partitions, $A < v$ and $A \geq v$
- Simple method to choose best v
  - For each v, scan the database to gather count matrix and compute its Gini index
  - Computationally Inefficient! Repetition of work.

Continuous Attributes: Computing Gini Index

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Continuous Attributes: Computing Gini Index

- For efficient computation: for each attribute,
  - Sort the attribute on values
  - Linearly scan these values, each time update the count matrix and compute gini index
  - Choose the split position that has the least Gini index

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No
	Taxable Income									
Sorted Values	60	70	75	85	90	95	100	120	125	220

Split Positions	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		*0.300*		0.343		0.375		0.400		0.420	

Entropy

$$Entropy(t) = -\sum_{i=0}^{c-1} p(i|t) log_2 p(i|t)$$

- Measures homogeneity of a node.
  - Maximum ($log(n_c)$) when records are equally distributed among all classes implying least information
  - Minimum (0.0) when all records belong to one class, implying most information

Examples for Computing Entropy

$$Entropy(t) = -\sum_{i=0}^{c-1} p(i|t) log_2 p(i|t)$$

C1	0
C2	6

C1	1
C2	5

C1	2
C2	4

C1	3
C2	3

Entropy?

- $-0log0 - 1log1 = 0$ (prob. is 0 means it does not happen, let $0log0 = 0$)
- $-\frac{1}{6}log(\frac{1}{6}) - \frac{5}{6}log(\frac{5}{6}) = 0.65$
- $-\frac{2}{6}log(\frac{2}{6}) - \frac{4}{6}log(\frac{4}{6}) = 0.92$

Splitting Based on Information Gain

$$Information\ Gain = Entropy(p) - \sum_{j=1}^{k} \frac{N(v_j)}{N} Entropy(v_j)$$

- Used in ID3
- Disadvantage: Tends to prefer splits that result in large number of partitions, each being small but pure.
- Consider partition on ID? $Entropy(children) = 0$

Splitting Based on Gain Ratio

$$Gain\ Ratio = \frac{Information\ Gain}{Split\ Info}$$

$$Split\ Info = \sum_{j=1}^{k} (\frac{n_j}{n} log(\frac{n_j}{n}))$$

- Parent node p is split into k partitions
- n_j is the number of records in partition j
- Higher entropy partitioning (large number of small partitions) is penalized!
- Used in C4.5, successor of ID3
- Designed to overcome the disadvantage of Information Gain

Classification Error

$$Error(t) = 1 - max_i p(i|t)$$

- Measures misclassification error made by a node.
  - Maximum $(1 - 1/n_c)$ when records are equally distributed among all classes, implying least interesting information
  - Minimum (0.0) when all records belong to one class, implying most interesting information

Examples for Computing Classification Error

$$Error(t) = 1 - max_i p(i|t)$$

C1	0
C2	6

C1	1
C2	5

C1	2
C2	4

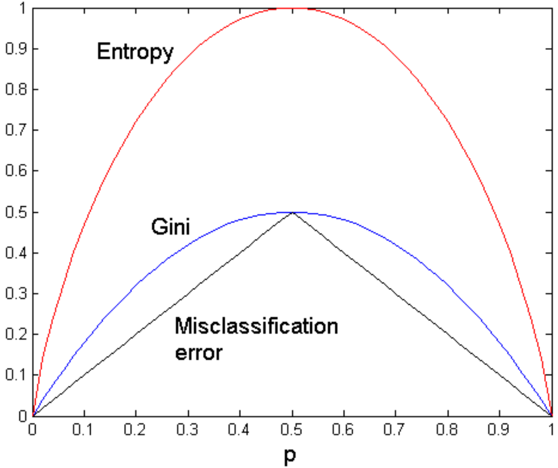
C1	3
C2	3

Classification Error?

- $1 - max(0, 1) = 0$
- $1 - max(1/6, 5/6) = 1 - 5/6 = 1/6$
- $1 - max(2/6, 4/6) = 1 - 4/6 = 1/3$
- 0.5

Comparison among Splitting Criteria

For a 2-class problem:



Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values
- Early termination: use threshold

Examples

- ID3: Use Entropy, Information gain
- C4.5: Use Entropy, Normalized Information Gain (Gain Ratio) Download the software from: `http://www.cse.unsw.edu.au/~quinlan/c4.5r8.tar.gz`
- CART: Uses Gini Index, only binary splits

Which measure is the best?

- Time complexity of DT induction increases exponentially with tree height $\rightarrow$ shallower trees
- Shallow trees tend to have a large number of leaves and higher error rates

Example: C4.5

- Simple depth-first construction.
- Uses Information gain
- Sorts continuous attributes at each node.
- Needs entire data to fit in memory.
- Unsuitable for large datasets.
- Download the software from:
  `http://www.cse.unsw.edu.au/~quinlan/c4.5r8.tar.gz`

DT Example (1)

RID	age	income	student	credit_rating	buys_computer
1	$<= 30$	high	no	fair	no
2	$<= 30$	high	no	excellent	no
3	$31 \cdots 40$	high	no	fair	yes
4	> 40	medium	no	fair	yes
5	> 40	low	yes	fair	yes
6	> 40	low	yes	excellent	no
7	$31 \cdots 40$	low	yes	excellent	yes
8	$<= 30$	medium	no	fair	no
9	$<= 30$	low	yes	fair	yes
10	> 40	medium	yes	fair	yes
11	$<= 30$	medium	yes	excellent	yes
12	$31 \cdots 40$	medium	no	excellent	yes
13	$31 \cdots 40$	high	yes	fair	yes
14	> 40	medium	no	excellent	no

Class label attribute: *buys_computer*
Preprocess *age* attribute:

- $<= 30$: young
- $31 \cdots 40$: middle_aged
- > 40: senior

DT Example (2)

RID	age	income	student	credit_rating	buys_computer
1	young	high	no	fair	no
2	young	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	young	medium	no	fair	no
9	young	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	young	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

DT Example (3) – Information gain based on Gini Index

Number of classes:

- C_1: for *yes*, 9
- C_2: for *no*, 5

$Gini(D) = 1 - (\frac{9}{14})^2 - (\frac{5}{14})^2 = 0.459$

Consider using attribute *income* as splitting attribute:

- D_1: *income* is {low, medium}
  $Gini(D_1) = 1 - (\frac{6}{10})^2 - (\frac{4}{10})^2 = \frac{48}{100}$
- D_2: *income* is {high}
  $Gini(D_2) = 1 - (\frac{2}{4})^2 - (\frac{2}{4})^2 = \frac{1}{2}$
- $Gini(children) = \frac{10}{14} * \frac{48}{100} + \frac{4}{14} * \frac{1}{2} = \frac{12}{35} + \frac{1}{7} = \frac{17}{35} = 0.486$
- $Gain = Gini(D) - Gini(children) = -0.027$

DT Example (4) – Information gain based on Entropy

$$Entropy(D) = -\frac{9}{14}log_2(\frac{9}{14}) - \frac{5}{14}log_2(\frac{5}{14}) = 0.94$$

Compute the expected entropy for *each* attribute.
Start with attribute *age*.
Consider multi-split (i.e., 3-split). D_{young}: 2 yes, 3 no;

$$Avg\ Entropy = 0.694$$

Calculation?

$$Gain(age) = 0.94 - 0.694 = 0.246$$

Compute gain for other attributes: Gain(income) = 0.029, Gain(student), etc.

DT Example (5) – Gain ratio

Consider attribute *income*.

- low: 4
- medium: 6
- high: 4

$$Split\ Info(D) = -\frac{4}{14}log_2(\frac{4}{14}) - \frac{6}{14}log_2(\frac{6}{14}) - \frac{4}{14}log_2(\frac{4}{14}) = 1.56$$

$$Gain\ ratio(income) = \frac{0.029}{1.56} = 0.019$$

Building a decision tree - Python example

Building a decision tree does not need to standardize the attribute values.

```
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier

tree_model = DecisionTreeClassifier(criterion='gini',
                                    max_depth=4,
                                    random_state=1)
tree_model.fit(X_train, y_train)

tree.plot_tree(tree_model)
plt.show()
```

Prediction - Python example

- `predict(self,X,check_input=True)`
  Predict class (or regression value) for X.

- `predict_proba(self,X,check_input=True)`
  Predict class probabilities of the input samples X.

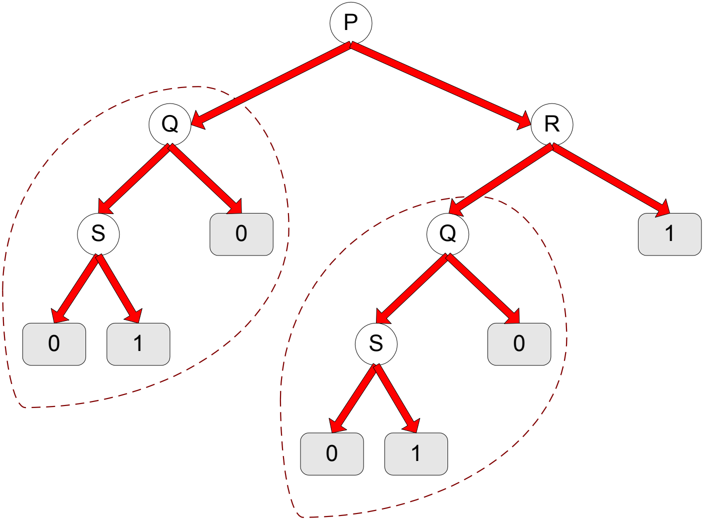
Decision Tree Based Classification – Advantages

- Inexpensive to construct
- Extremely fast at classifying unknown records
- Easy to interpret for small-sized trees
- Accuracy is comparable to other classification techniques for many simple data sets

Decision Boundary

- Decision boundary: border line between two neighboring regions of different classes
- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time.
- **Oblique decision tree**
  - Test condition may involve multiple attributes $x + y < 1$
  - More expressive representation
  - Finding optimal test condition is computationally expensive
- **Constructive induction**
  - Purpose: partition the data into homogeneous, non-rectangular regions.
  - Composite attribute

Tree Replication



References

- Chapter 3: Introduction to Data Mining (2nd Edition) by Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, and Vipin Kumar

- DecisionTreeClassifier: `https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html`