Knowledge Discovery and Data Mining

Unit # 2

Acknowledgement

- Most of the slides in this presentation are taken from course slides provided by
  - Han and Kimber (Data Mining Concepts and Techniques) and
  - Tan, Steinbach and Kumar (Introduction to Data Mining)

Alternative Splitting Criteria based on Entropy

- Entropy at a given node t:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node 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

– Entropy based computations are similar to the GINI index computations

– Hint: $\log_2 p = \ln p / \ln(2)$

Entropy in a nut-shell

Low Entropy

High Entropy

Examples for computing Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

Splitting Criteria based on Classification Error

- Classification error at a node t :

$$Error(t) = 1 - \max_i P(i \mid t)$$

- 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 Error

$$Error(t) = 1 - \max_i P(i \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Error = 1 – max (0, 1) = 1 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

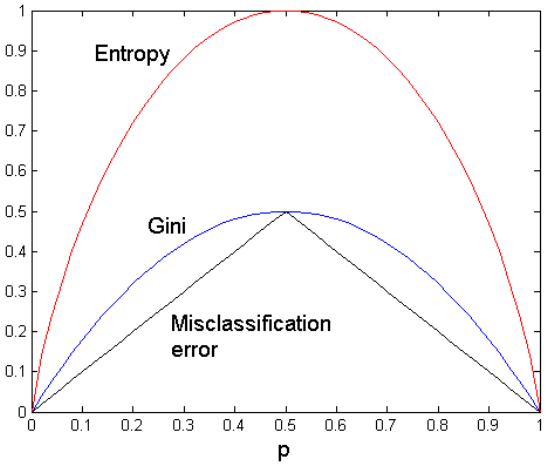
C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

Comparison among Splitting Criteria

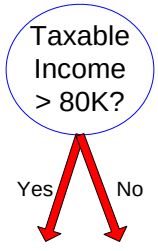
For a 2-class problem:



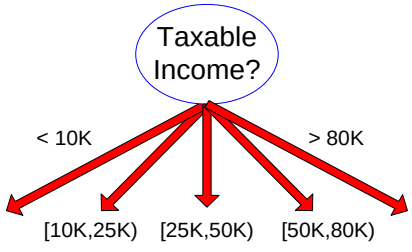
Splitting Based on Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Binary Decision: (A < v) or (A ≥ v)
    - consider all possible splits and finds the best cut
    - can be computationally intensive

Splitting Based on Continuous Attributes (Cont'd)



(i) Binary split



(ii) Multi-way split

Continuous Attributes: Computing GINI Index

- For efficient computation: for each attribute,
  - Sort the attribute on values
  - Linearly scan these values, each time updating the count matrix and computing 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	

Background

- ID3 (Iterative Dichotomiser 3)
  - published in 1986 but proposed in 1983
  - Only works on non-continuous (discrete) attributes
  - Uses Information Gain/Entropy as the splitting rule
- CART
  - Published in 1984
  - Uses Gini Index as the splitting rule
  - Binary trees
- C4.5
  - Extension of ID3 and published in 1993
  - Works on continuous attributes
  - Uses modified Gain/Entropy metric as the splitting rule to defy advantage to variables having multiple states

Categorical Attributes: Computing GINI Index

- From a historical perspective, Gini Index always created a binary tree.
- As a result, in case of multiple values, it merged them together to find the best binary split
- For each distinct value, gather counts for each class in the dataset

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	

Handling of Multi-state Variable

- The way both Gini Index and Entropy are presented, they become biased to variables having multiple states.
- To over this, the following approach was recommended (in C4.5 using Entropy but can be generalized to Gini Index as well).
  - Gain = SR (D) – SR_A (D)
  - Where SR = splitting rule metric
  - D = class variable
  - A = an attribute on which the splitting rule is conditioned

Buy Computer Example

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

SplitInfo

- Gini (buy) = 0.46
  - Gini $_{Age}$ (buy) = 0.34 : Gain = 0.12
  - Gini $_{inc}$ (buy) = 0.44 : Gain = 0.02
  - Gini $_{std}$ (buy) = 0.37 : Gain = 0.09
  - Gini $_{rat}$ (buy) = 0.43 : Gain = 0.03
- SplitInfo = unconditional splitting rules on the variables. If one is using Gini then it becomes
  - Splitinfo (age) = Gini (age) = 0.66
  - Splitinfo (inc) = Gini (inc) = 0.65
  - Splitinfo (std) = Gini (std) = 0.5
  - Splitinfo (rat) = Gini (rat) = 0.49

Gain_ratio

- To obtain gain ratio, we divide gain by splitinfo
  - Gain_ratio (age) = 0.12 / 0.66 = 0.18 (0.175)
  - Gain_ratio (inc) = 0.02 / 0.65 = 0.03
  - Gain_ratio (std) = 0.09 / 0.5 = 0.18 (0.184)
  - Gain_ratio (rat) = 0.03 / 0.49 = 0.06
- A similar computation would have been done if we were using Entropy or even Miclassification Error

Example

Attribute 1	Attribute 2	Attribute 3	Class
A	70	T	C1
A	90	T	C2
A	85	F	C2
A	95	F	C2
A	70	F	C1
B	90	T	C1
B	78	F	C1
B	65	T	C1
B	75	F	C1
C	80	T	C2
C	70	T	C2
C	80	F	C1
C	80	F	C1
C	96	F	C1

Example II

Height	Hair	Eyes	Class
Short	Blond	Blue	+
Tall	Blond	Brown	-
Tall	Red	Blue	+
Short	Dark	Blue	-
Tall	Dark	Blue	-
Tall	Blond	Blue	+
Tall	Dark	Brown	-
Short	Blond	Brown	-

Inducing a decision tree

- There are many possible trees
- How to find the most compact one
  - that is consistent with the data?
- The *key* to building a decision tree - which attribute to choose in order to branch.
- The *heuristic* is to choose the attribute with the minimum GINI/Entropy.

Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive manner
  - At start, all the training examples are at the root
  - Attributes are categorical
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., GINI/Entropy)
- Conditions for stopping partitioning
  - All examples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no examples left

Extracting Classification Rules from Trees

- Represent the knowledge in the form of IF-THEN rules
- One rule is created for each path from the root to a leaf
- Each attribute-value pair along a path forms a conjunction. The leaf node holds the class prediction
- Rules are easier for humans to understand
- Example

IF *age* = "<=30" AND *student* = "*no*" THEN *buys_computer* = "*no*"
IF *age* = "<=30" AND *student* = "*yes*" THEN *buys_computer* = "*yes*"
IF *age* = "31…40" THEN *buys_computer* = "*yes*"
IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*yes*"
IF *age* = "<=30" AND *credit_rating* = "*fair*" THEN *buys_computer* = "*no*"

Characteristics of Decision Tree Induction

- Decision tree induction is a non-parametric approach for building classification models. In other words, it doesn't require any prior assumptions regarding the type of probability distributions satisfied by the class and other attributes.
- Finding an optimal decision tree is an NP-complete problem. Many decision tree algorithms employ a heuristic-based approach to guide their search in the vast hypothesis space. For example, the algorithm discussed in this unit uses a greedy, top-down, recursive partitioning strategy for growing a decision tree.

Characteristics of Decision Tree Induction (Cont'd)

- Techniques developed for constructing decision trees are computationally inexpensive, making it possible to quickly construct models even when the training set size is very large. Furthermore, once a decision tree has been built, classifying a test record is extremely fast, with a worst-case complexity of O(w), where w is the maximum depth of the tree.
- Decision tree, specially smaller-sized trees, are relatively easy to interpret.
- Decision tree algorithms are quite robust to the presence of noise.

Characteristics of Decision Tree Induction (Cont'd)

- The presence of redundant attributes does not adversely affect the accuracy of decision trees. An attribute is redundant if it is strongly correlated with another attribute in the data. One of the two redundant attributes will not be used for splitting once the other attribute has been chosen.
- Studies have shown that the choice of impurity measures has little effect on the performance of decision tree induction algorithms.

Advantages of Decision Tree Based Classification

- 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

Data Preparation/Wrangling

- Data Transformation
  - Log
  - Aggregation
  - Normalization
- Missing Value Handling
  - Average / Most frequent
  - Rows deletion
- Feature Discretization
- Feature Selection/Ranking

Aggregation

- Combining two or more attributes (or objects) into a single attribute (or object)
- Purpose
  - Data reduction
    - Reduce the number of attributes or objects
  - Change of scale
    - Cities aggregated into regions, states, countries, etc
  - More “stable” data
    - Aggregated data tends to have less variability

Data Normalization

- Some data mining methods, typically those that are based on distance computation between points in an n-dimensional space, may need normalized data for best results.
- If the values are not normalized, the distance measures will overweight those features that have, on average, larger values.

Normalization Techniques

- Decimal Scaling
  - $v'(i) = v(i) / 10^k$
  - For the smallest k such that max $|v'(i)| < 1$.
- Min-Max Normalization
  - $v'(i) = [v(i) - \min(v(i))]/[\max(v(i)) - \min(v(i))]$
- Standard Deviation Normalization
  - $v'(i) = [v(i) - \text{mean}(v)]/\text{sd}(v)$

Normalization Example

- Given one-dimensional data set X = {-5.0, 23.0, 17.6, 7.23, 1.11}, normalize the data set using
  - Decimal scaling on interval [-1, 1].
  - Min-max normalization on interval [0, 1].
  - Standard deviation normalization.

Outlier Detection

- Statistics-based Methods (*for one dimensional data*)
  - Threshold = Mean $\pm$ K x Standard Deviation
  - Age = {3, 56, 23, 39, 156, 52, 41, 22, 9, 28, 139, 31, 55, 20, -67, 37, 11, 55, 45, 37}
- Distance-based Methods (*for multidimensional data*)
  - Distance-based outliers are those samples which do not have enough neighbors, where neighbors are defined through the multidimensional distance between samples.

Outlier Detection (Distance-based)

- S = {s1, s2, s3, s4, s5, s6, s7} = {(2, 4), (3, 2), (1, 1), (4, 3), (1, 6), (5, 3), (4, 2)}
- Threshold Values: p ≥ 4, d ≥ 3

	S1	S2	S3	S4	S5	S6	s7
S1		2.236	3.162	2.236	2.236	3.162	2.828
S2			2.236	1.414	4.472	2.236	1.000
S3				3.605	5.000	4.472	3.162
S4					4.242	1.000	1.000
S5						5.000	5.000
s6							1.414

Sample	p
S1	2
S2	1
S3	5
S4	2
S5	5
s6	3

Outlier Detection Example II

- The number of children for different patients in a database is given with a vector C = {3, 1, 0, 2, 7, 3, 6, 4, -2, 0, 0, 10, 15, 6}.
  - Find the outliers in the set C using standard statistical parameters mean and variance.
  - If the threshold value is changed from ±3 standard deviations to ±2 standard deviations, what additional outliers are found?

Outlier Detection Example III

- For a given data set X of three-dimensional samples, X = [{1, 2, 0}, {3, 1, 4}, {2, 1, 5}, {0, 1, 6}, {2, 4, 3}, {4, 4, 2}, {5, 2, 1}, {7, 7, 7}, {0, 0, 0}, {3, 3, 3}].
- Find the outliers using the distance-based technique if
  - The threshold distance is 4, and threshold fraction p for non-neighbor samples is 3.
  - The threshold distance is 6, and threshold fraction p for non-neighbor samples is 2.
- Describe the procedure and interpret the results of outlier detection based on mean values and variances for each dimension separately.

Data Reduction

- The three basic operations in a data-reduction process are:
  - Delete a row
  - Delete a column (dimensionality reduction)
  - Reduce the number of values in a column (smooth a feature)
- The main advantages of data reduction are
  - ***Computing time*** – simpler data can hopefully lead to a reduction in the time taken for data mining.
  - ***Predictive/descriptive accuracy*** – We generally expect that by using only relevant features, a data mining algorithm can not only learn faster but with higher accuracy. Irrelevant data may mislead a learning process.
  - ***Representation of the data-mining model*** – The simplicity of representation often implies that a model can be better understood.

Feature Subset Selection

- Another way to reduce dimensionality of data
- Redundant features
  - duplicate much or all of the information contained in one or more other attributes
  - Example: purchase price of a product and the amount of sales tax paid
- Irrelevant features
  - contain no information that is useful for the data mining task at hand
  - Example: students' ID is often irrelevant to the task of predicting students' GPA

Mean and Variance based Feature Selection

- Suppose A and B are sets of feature values measured for two different classes, and n1 and n2 are the corresponding number of samples.
  - SE(A – B) = Sqrt (var(A)/n1 + var(B)/n2)
  - TEST: |mean(A) – mean(B)|/SE(A – B) > threshold value
- It is assumed that the given feature is independent of the others.

Mean-Variance Example

- $SE(X_A - X_B) = 0.169$
- $SE(Y_A - Y_B) = 0.0875$
- $|mean(X_A) - mean(X_B)| / SE(X_A - X_B) = 0.0375 < 0.5$
- $|mean(Y_A) - mean(Y_B)| / SE(Y_A - Y_B) = 2.2667 > 0.5$

X	Y	C
0.3	0.7	A
0.2	0.9	B
0.6	0.6	A
0.5	0.5	A
0.7	0.7	B
0.4	0.9	B

Feature Ranking Exercise

- Given the data set X with three input features and one output feature representing the classification of samples

I1	I2	I3	O
2.5	1.6	5.9	0
7.2	4.3	2.1	1
3.4	5.8	1.6	1
5.6	3.6	6.8	0
4.8	7.2	3.1	1
8.1	4.9	8.3	0
6.3	4.8	2.4	1

- Rank the features using a comparison of means and variances