Assignment - II

Data Mining & warehouse

— Abhishek Roy
roll - 58 Department of CSE (3rd yr)

Mar 2, 2023

Q.

department	status	age	salary	count
sales	senior	31 — 35	46K — 50K	30
sales	junior	26 — 30	26K — 30K	40
systems	junior	31 — 35	31K — 35K	40
systems	junior	21 — 25	46K — 50K	20
systems	senior	31 — 35	66K — 70K	5
systems	junior	26 — 30	46K — 50K	3
systems	senior	41 — 45	66K — 70K	3
systems	senior	36 — 40	46K — 50K	10
marketing	senior	36 — 40	46K — 50K	4
marketing	junior	31 — 35	41K — 45K	4
secretary	senior	46 — 50	36K — 40K	6
secretary	junior	26 — 30	26K — 30K	

a] How would you modify the basic decision tree algorithm to take into consideration the count of each generalized data tuple?

Ans → The basic decision tree algo should be modified as follows to take into consideration the count of each generalized data tuple —

- The count of each tuple must be integrated into the calculation of the attribute selection measure (such as information gain)
- Take the count into consideration to determine the most common class among the tuples.

b] Use your algo to construct a decision tree from given data.

Ans First we calculate gini (Gini)- Index for entire dataset

Total - count = 150
sales - count = 110
system - count = 28
marketing - count = 14
secretary - count = 10

→ Gini- Total = [1 - ((sales-count/total-count)×2 + (system-count/total-count)×2 + (marketing-count/total-count)×2 + (secretary-count/total-count))]

Gini_total = [1 - (0.7332 + 0.1872 + 0.0932 + 0.0672)]

$= \boxed{0.612}$

→ Now calculate Gini_Index for each attribute →

- department

→ sales: senior = 0

junior = $\left[1 - \left(\left(\frac{40}{70}\right)\times 2 + \left(\frac{30}{70}\right)\times 2\right]\right.$ $= \boxed{0.489}$

→ System : Senior = $\left[1-\left(\left(\frac{5}{8}\right)\times 2+\left(\frac{3}{8}\right)\times 2\right)\right] = \boxed{0.469}$

Junior = $\left[1-\left(\left(\frac{23}{28}\right)\times 2+\left(\frac{5}{28}\right)\times 2\right)\right] = \boxed{0.408}$

→ Marketing : senior = 0 Junior = 0

→ secretary : senior = $\boxed{0.375}$ Junior = $\boxed{0.5}$

→ Status : Senior → sales = 0.469
systems = 0.375
Marketing = 0.5
Secretary = 0

Junior → sales = 0.489
systems = 0.408
Marketing = 0
secretary = 0.5

→ Age : (21—25) = 0
(26-30) → sales = 0.489
systems = 0.408
Marketing = 0
secretary = 0.5

(31-35) →
sales = 0.489
systems = 0.469
Marketing = 0
secretary = 0

(36-40) = 0
(41-45) = 0
(46-50) →
secretary = 0.375

→ salary : (26k-30k) →
sales = 0.489
systems = 0
Marketing = 0
secretary = 0.5

(31k-35k) →
sales = 0.489
system = 0.469
Marketing = 0
secretary = 0

(36k—40k) →
secretary = 0.375
(41k-45k) →
Marketing = 0
(46k-50k) →
sales = 0.489
system = 0.469
Marketing = 0
secretary = 0

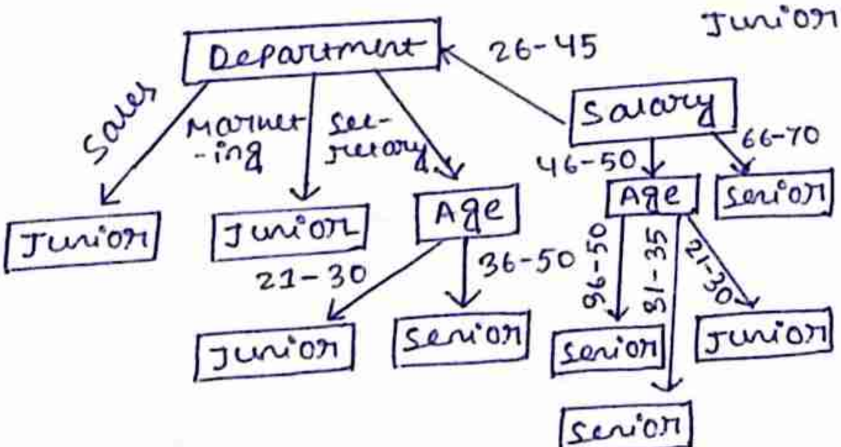
→ Attribute of lowest Gini-Index is Department with value = $\boxed{0.373}$
split dataset with based on department attribute → sales : status →

Senior = 0

Junior = $\left[1-\left[\left(\frac{40}{70}\right)\times 2+\left(\frac{30}{70}\right)\times 2\right]\right]$

$= \left[1-(1.14+0.85)\right]$

$= \boxed{0.997}$



← - - - [Decision-tree]

c] Given a data-tuple having the values "systems", "26-30" & "46-50k" for the attributes department / age / salary respectively what would be ~~aeg~~ naive Bayesian classification?

→ Given a data tuples with the values — "System" "Junior" "26---30" for the attribute department status / age respectively what would a naive Bayesian classification for salary tuple be →

P(X | senior) = 0 | P(X | Junior) = 0.018

- Thus a naive Bayesian classification predicts Junior (Ans)

Q Why is tree pruning useful in decision tree induction? What is a drawback of using a separate set of tuples to evaluate pruning?

Ans, The decision tree built may overfit the training data. There could be too many branches, some of which may reflected anomalies in the training data due to noise. Tree pruning addresses this issue of overfitting the data by removing the least reliable branches. This generally result in a more compact and reliable decision tree that is faster and more accurate in it's classification.

The drawback of using a separate set of tuples to evaluate pruning is that it may not be representative of the training tuple used to create the original decision tree. If the separate set of tuples are skewed then using them to evaluate the pruned tree would not be a good indicator of the pruned tree's classification accuracy. Furthermore, using a separate set of tuple to evaluate pruning means there are less tuples to use for creation & testing of the tree. While this is also considered a drawback in machine learning. It may not be so in data mining due to the availability of larger data sets.

Q Briefly outline the major steps of decision tree classification?

Ans, Step-I : Determine the root of the tree

Step-II : Calculate Entropy for the classes.

Step-III : Calculate Entropy after split for each attribute.

Step-IV : Calculate information gain for each split.

Step-V : Perform further split.

Step-VI : Complete the decision tree.

Calculation formula →

- $\text{Gini} = 1 - \sum_{i=1}^{n} p^2(C_i)$
- $\text{Entropy} = \sum_{i=1}^{n} - P(C_i) \log_2(P(C_i))$

[$P(C_i)$ ← probability / Percentage of class (C_i) ← In a node]

$E(S) = -(P_+) * \log_2(P_+) - (P_-) * \log_2(P_-)$

Entropy (S)

[where (P_+) positive sample
(P_-) negative sample
(S) sample of attribution]

Q. Why is naive Bayesian classification called "naive"? Briefly outline the major ideas of naive Bayesian classification?

Ans. Naive Bayesian classification is called naive cause, it assumes class conditional independence.

- That is, the effect of an attribute value on a given class is independent of the value of other attributes.
- The assumption is made to reduce computational costs, and hence is considered "naive"
- The major idea behind "na'ive" Bayesian classification is to try and classify data by maximizing $P(X|C_i)P(C_i)$ [where, i = index of the class] using the Baye's theorem of posterior probability.
- We are given a set of unknown data tuples, where each tuple is represented by an n-dimensional vector. $X = (X_1, X_2 \ldots X_n)$ depicting n-measurement made on the tuple from n-attribute, respectively $(A_1, A_2 \ldots A_n)$. also given a set of m-classes $(C_1, C_2, \ldots C_m)$
- using Bayes theorem, the naive bayesian classifier calculates the posterior probability of each class conditioned on X. (X ← assigned the class label of the class with max posterior)

try to maximize $P(C_i|X) = P(X|C_i)P(C_i)/P(X)$

However since, $P(X)$ is constant for all classes. only the $P(X|C_i)P(C_i)$ need be maximized. If the-

- class prior probability are not unknown. then it's common assumesed that the classes are equally likely-

$$P(C_1) = P(C_2) = \ldots P(C_m)$$

Therefore, should be maximize $P(X|C_i)$

- otherwise, maximize $P(X|C_i)P(C_i)$ the class prior probabilities may be estimated by $P(C_i) = s_i$ (where 's_i' ← num of training tuples of class C_i, s ← total num of training tuples)

- In order of reduce computation in evaluating $P(X|C_i)$
If A_k is categorical attribute then $P(X_k|C_i)$ equal to the num of training tuples in 'C_i'. that have 'X_k' as the val for that attribute, divided by total num of training tuples in (C_i)
- If A_k is continous attribute then $P(X_k|C_i)$ can be calculated using Gaussian density function.

Q what is information gain, Gain ratio, Gini-Index?

Ans, Information gain :→

⟨i⟩ Information gain is used for determining the best features / attribute that render information about a class.

⟨ii⟩ It follows the concept of entropy while aiming at decreasing the level of entropy, begining from root node to the leaf node.

$$\left[\text{Information Gain} = \left(\begin{array}{c}\text{Entropy before}\\ \text{splitting}\end{array}\right) - \left(\begin{array}{c}\text{Entropy after}\\ \text{splitting}\end{array}\right) \right]$$

Gain ratio :→

⟨i⟩ First, determine the information gain of all the attributes, and then compute the avg of info gain.

⟨ii⟩ second, calculate the gain ratio of all of the attribute whose calculated information Gain is larger / equal to the computed avg information Gain. then pick to the attribute of higher gain ratio.

$$\left[\text{Gain ratio} = \left(\frac{\text{Information gain}}{\text{Entropy}} \right) \right]$$

Gini-Index :→

⟨1⟩ Gini-Index computes the degree of probabilty of a specific variables that is wrongly being classified when chosen randomly and a variation of the gini-coefficient. It works on categorical variables provides outcome.

⟨11⟩ It varies from 0 → 1 where —

- 0 depicts that all the elements be allied to a certain class or only / only one class exists there.
- Gini-Index of values as 1 signifies that all the elements are randomly distribute across various classes.
- value of (0.5) denotes the elements are uniformly distributed into some classes.

Q Generate decision-tree algorithm?

Ans, Input:

- Data partition, D which is a set of training tuples and their associated class labels.
- attribute_list, the set of condidate attribute.
- Attribute_selection_method, a procedure to determine the spiting criterion that "best" partitions the data tuples into individual classes. The criterion consists of a spitting_attribute and posibly, either a spitting subnet.

Output: A decision tree

Method:

1. create a node N
2. If tuples in 'D' are all of the same class C, then return N as a leaf node labelled with class 'C'.
3. If attribute_list is empty then
4. return N as a leaf node labelled with the majority class in D // Majority voting
5. Apply attribute_selection_method (D, attribute_list) to find the best spiliting_criterion;
6. Label node N with spitting criterion;
7. If spitting_attribute is discrete-valued and multiway spits allowed then
8. attribute_list ← attribute_list - spilting attribute
9. for each outcome 'j' of spitting_creation. // partition the tuples & grow subtree
10. let 'Dj' be the set of data tuples in D satisfying outcome 'j'; // a partition.
11. If 'Dj' is empty then
12. attach a leaf labelled with the majority class in D to node N;
13. else attach a leaf labelled with the majority class return by Generate_decision_tree(D, attribute_list) to node N;
14. return N;

Q Explain Bayesian classification?

Ans

"Bayesian classification"

→ Baye's theorem →

i) Posterior probability [P(H/X)]
ii) Prior probability [P(H)]

where,
X ← data tuple
H ← hypothesis

According to Baye's theorem —

$$P(H/X) = P(X/H)\,P(H) / P(X)$$

→ Bayesian belief network →

i) A belief network allows class conditional independencies to be defined between subset of variables.

ii) It provides a graphical model of casual relationship on which learning can be performed.

iii) we can use a trained Bayesian network for classification.

→ Components of Baye'sian network —

(i) Directed acyclic graph

(ii) A set of conditional probability table

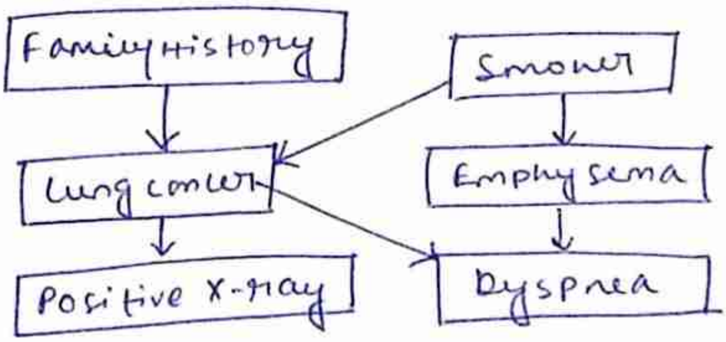
→ Directed acyclic graph →

i) Each node in a directed acyclic graph represents a random variable.

ii) These variable may be discrete valued.

iii) These variable may correspond to the actual attribute given in the data.

→ conditional probability table →



The conditional probability table for the values of the variable lung cancer (LC) showing each possible comb of values of it's parent nodes Family History (FH) | Smoker (S)

	(FH, S)	(FH, -S)	(-FH, S)	(-FH, -S)
LC	0.8	0.5	0.7	0.1
-LC	0.2	0.5	0.3	0.9

Q What is the use of regression? What may be the reason for not using the linear regression model to estimate the output data?

Ans → Regression analysis is a statistical method that is used to estimate the relationship between a dependent variable and one/more independent variables.
The primary use of regression analysis is to predict/estimate the value of the dependent variable based on the values of independent variable.
For example, researcher may use regression analysis to estimate sales of product.

→ Linear regression popularly & widely used for →
i) nonlinear relationships
ii) outliers
iii) multicollinearity
iv) categorical variables
v) Appropriateness of linear regression.

Q 8.3 → If pruning a subtree, we would remove the subtree completely with method (b). However with method (a) if pruning a rule, we may remove any precondition of it. The latter is less restrictive.

Q 8.4 → The worst case scenario occurs when we have to use as many attributes as possible before being able to classify each group of tuples. The max depth of the tree = log(|D|). At each level we will have to compute the attribute selection measure = O(n) times.
The total num of tuples at each level of tree O(n × |D|)
Summing over all of the levels obtain O(n × |D| × log(|D|)).

Q 8.5 → We will use the rainforest algo for this problem, assumes there are 'c' class labels. Most memory required will be for AVC-set for the root of the tree. To compute the AVC-set for the root node we scan the database once & construct AVC-list (100×c)
Total size of AVC-set = (100×c×50) & which will easily fit into 512 MB of memory for reasonable c.
The computation of other AVC set is done in a smaller way but they will be smaller cause will be less attribute present. To reduce the num of scans we can compute the AVC-set for nodes at the same level of the tree in parallel with such small AVC set.