CPSC 8650 Homework Assignment #3

8.3. Given a decision tree, you have the option of (a) converting the decision tree to rules and then pruning the resulting rules, or (b) pruning the decision tree and then converting the pruned tree to rules. What advantage does (a) have over (b)?

8.4. It is important to calculate the worst-case computational complexity of the decision tree algorithm. Given data set D, the number of attributes *n*, and the number of training tuples |D|, show that the computational cost of growing a tree is at most $n \times |D| \times \log(|D|)$.

8.5. Given a 1TB dataset with 50 attributes (each containing 100 distinct values) and 32GB main memory in your laptop, outline an efficient method that constructs decision trees in such large data sets. Justify your answer by rough calculation of your main memory usage.

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

8.7. The following table consists of training data from an employee database. The data have been generalized. For example, "31 . . . 35" for age represents the age range of 31 to 35. For a given row entry, count represents the number of data tuples having the values for department, status, age, and salary given in that row.

department	*status*	*age*	*salary*	*count*
sales	senior	31. . . 35	46K. . . 50K	30
sales	junior	26. . . 30	26K. . . 30K	40
sales	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
marketing	senior	36. . . 40	46K. . . 50K	10
marketing	junior	31. . . 35	41K. . . 45K	4
secretary	senior	46. . . 50	36K. . . 40K	4
secretary	junior	26. . . 30	26K. . . 30K	6

Let status be the class label attribute.

(a) How would you modify the basic decision tree algorithm to take into consideration the count of each generalized data tuple (i.e., of each row entry)?
(b) Use your algorithm to construct a decision tree from the given data.
(c) Given a data tuple having the values "systems", "26. . . 30", and "46–50K" for the attributes department, age, and salary, respectively, what would a naive Bayesian classification of the status for the tuple be?

8.10. Show that accuracy is a function of sensitivity and specificity, that is, prove Equation 8.25 in the textbook:

$$accuracy = sensitivity \cdot \frac{P}{P+N} + specificity \cdot \frac{N}{P+N}$$

8.12. The data tuples of Figure 8.25 are sorted by decreasing probability value, as returned by a classifier. For each tuple, compute the values for the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). Compute the true positive rate (TPR) and false positive rate (FPR). Plot the ROC curve for the data.

Tuple #	*Class*	*Prob.*
1	p	0.95
2	n	0.85
3	p	0.78
4	p	0.66
5	n	0.60
6	p	0.55
7	n	0.53
8	n	0.52
9	n	0.51
10	p	0.4

Figure 8.25: Tuples sorted by decreasing score, where the score is the value returned by a probabilistic classifier.

8.14. Suppose that we would like to select between two prediction models, M1 and M2. We have performed 10 rounds of 10-fold cross validation on each model, where the same data partitioning in round *i* is used for both M1 and M2. The error rates obtained for M1 are 30.5, 32.2, 20.7, 20.6, 31.0, 41.0, 27.7, 26.0, 21.5, 26.0. The error rates for M2 are 22.4, 14.5, 22.4, 19.6, 20.7, 20.4, 22.1, 19.4, 16.2, 35.0. Comment on whether one model is significantly better than the other considering a significance level of 1%.

DATA MINING ASSIGNMENT - 3

Parampreet Singh

8.3 Pruning is a decision tree that refers to the process of simplifying by tree structure by removing certain nodes while retaining the most essential ones.

(a) Converting the decision tree to rules and then pruning the resulting rules:

Initially, the decision tree is transformed into a set of rules, where each rule corresponds to a path from the root node to a leaf node. The rules are then pruned to eliminate redundant rules, resulting in a simplified rule set. Pruning rules may involve removing rules with low predictive accuracy.

Advantage:-

It can be more intuitive and interpretable for humans, as they are presented in a format that resembles "if-then" statements pruning rules before converting to a tree may lead to a more concise and human understandable model.

(b) Pruning the decision tree and then converting the pruned tree to rules:-

The decision tree is first pruned to reduce

its complexity. Pruning typically involves cutting off branches with low information gain that lead to overfitting. After pruning, the pruned decision tree is then converted into a set of rules.

Advantage → Pruning the tree before conversion can potentially result in a more accurate set of rules, as the pruning process eliminates overfitting and retains the most relevant tree structure.

The advantage of option 'a' over 'b' is that option 'a' allows for more nuanced pruning that can adjust individual rules without losing potentially useful conditions. This method preserves the complexity where it's beneficial while still offering a path to simplify the model. This flexibility is particularly advantageous when dealing with complex datasets where the relationships b/w features and outcomes are not entirely straightforward.

8.4 Given dataset D,
number of attributes (n)
number of training tuples (|D|)

Need to show the computational cost of a growing decision tree is

$$n \times |D| \times \log D$$

So, in the worst case scenario the maximum depth of a tree is $\log|D|$, so here we have to use many attributes before we classify group of tuples and also we need to compute the measurements of attribute at one place attribute, so for at each level of attributes the total no. of tuples is $|D|$. thus the computation is calculated as $O(n*|D|)$ this computation summation leads to the $n \times |D| \times \log |D|$

8.5 The rainforest algorithm constructs decision trees by efficiently managing memory through the use of attribute value counts sets. These sets summarize the data, reducing the need to keep the entire dataset in memory.

Each AVC list for an attribute contains counts for each of the 100 distinct values across all class labels. If we denote the number of class labels by C, the size of an AVC list for one attribute can be estimated as follows:

Size of one AVC list = 100 × C × size of one count entry. Assuming each count entry requires 4 bytes of the count & an additional overhead for the class label, the total size for one count entry is 8 bytes.

∴ Total size of the AVC set for the root node (considering all 50 attributes) is:

$$100 \times C \times 8 \times 50 = 40{,}000 \times C$$

Let C = 10, total memory required for AVC set

$$40000 \times 10 = 400{,}000 \times 8 \text{ bytes} = 3{,}200{,}000 \text{ bytes} \approx 3.2 \text{ MB}$$

Memory required for storing the AVC set for the root of the tree is approx. 3.2 MB, which is less than the 32 GB of the main memory available. This leaves ample room for the other necessary data structures & operations such as :-

Storing AVC sets for other nodes :-
As decision tree is constructed, AVC sets for child nodes will be generated. These sets will likely be smaller than the root AVC set since they pertain to subsets of the data.

Parallel computation :-
Memory can be efficiently utilized to compute AVC sets for multiple nodes in parallel, especially for nodes at the same level of the tree.

Handling overheads & Temporary structures :-
Additionally memory will be used for operational overheads including the storage of the decision tree structure itself, temporary data structures for sorting or splitting data, and any overhead introduced by the programming environment.

Conclusion:

The rainforest algorithm's use of AVC set makes it possible to construct a decision tree from a 1 TB dataset on a laptop with 32 GB of main memory. The detailed calculations shows that the memory requirements for AVC sets are well within the available memory limit, allowing for efficient processing of large datasets. This approach not only minimizes memory usage but also reduces the no. of disc accesses required, making it an efficient method of constructing decision tree in resource constrained environments.

8.1 Bayesian theorem states to compute the $p(H/x)$ which is a posterior probability where x is a data sample $x = x_1, x_2 \ldots x_5$ and hypothesis H belongs to class C is H, so posterior probability also called the probability that the hypothesis holds given the observed data sample x and $p(H)$ is called prior probability also known as initial probability

So in order to calculate $P[H/x)$ with Bayesian it is difficult with lot of unknown parameters, so they have simplified by introducing naive bayesian classifier which assumes that class is conditionally independent that is the effect on an attribute value on a given class is independent of the values of the other attributes so by this assumption there is a reduction in computational costs too. Hence, it is considered as 'naive'.

Ideas -

It tries to classify data to maximize $P(x/c_i)P(c_i)$ as $p(x)$ is a constant for all attributes. Here i is a index of a class. So for a given set of

unknown data tuples.
Each tuple represented with n dimensional vector

$X = (x_1, x_2, \ldots x_n)$, n measurements on tuples from n attributes as $A_1, A_2, \ldots A_n$ with m classes $\rightarrow C_1, C_2, \ldots C_m$ with Bayes' theorem, naive bayesian classifier calculate the posterior probability of each class conditioned on X.

X is assigned the class label of the class with the maximum posterior probability conditioned on X.

So we try to maximize

$$P(C_i/X) = \frac{P(X/C_i)\,P(C_i)}{P(X)}$$

~~P(X)~~ $P(X/C_i) \cdot P(C_i)$ needs to be maximized

In order to reduce computation in evaluating $P(X/C_i)$, the ~~was~~ naive assumption of class conditional independence is made. This presumes that the value of the attributes are conditionally independent to

one another, given the class of the tuple i.e, that there are no dependence relationships among attributes

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) * P(x_2|C_i) * \cdots * P(x_n|C_i)$$

(so this greatly reduces computational cost, only counts the class distribution)

→ If A_k is categorical

$P(x_k|C_i)$ = no. of tuples in C_i having value x_k for A_k is divided by $|C_{i,D}|$ (no of tuples of C_i in D)

→ If A_k is continuous valued

$P(x_k|C_i)$ = usually computed based on gaussian distribution with a mean μ and std. σ

So,

$$P(x_k|C_i) = g(x_k, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$P(X|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

8.7. (a) The basic decision Tree should be modified as follows:

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

(b) To construct the decision tree, first finalize the root attribute.

Decision tree can be constructed using ID3 which is an attribute selection measure by calculating information gain, this process is called decision tree induction.

we need to select attribute with highest information gain

class label → status

↙ Junior (J) ↘ senior (S)

Expected Information (Entropy) = Info(D)

$$= \sum_{i=1}^{m} p_i \log_2(p_i)$$

$I(6,5) \Rightarrow (6J \& 5S)$

each J & S has a particular frequency associated to them. So,

total : Juniors $\Rightarrow 40+40+20+3+4+6$

$= 113$

Seniors $\rightarrow 30+5+3+10+4 = 52$

$$\text{Info}(D) = -\frac{113}{165}\log_2\left(\frac{113}{165}\right) - \frac{52}{165}\log_2\left(\frac{52}{165}\right)$$

$$= (-0.685)\log_2(0.685) - (0.32)\log_2(0.32)$$

$$= (-0.685)(-0.546) + (-0.32)(-1.64)$$

$$= 0.374 + 0.525$$

$$\approx 0.89$$

$$\text{Info}_{dept}(D) = \frac{160}{165} I(2_J, 1_S) + \frac{31}{165} I(2_J, 2_S) + \frac{14}{165} I(1_J, 1_S) + \frac{10}{165} I(1_J, 1_S)$$

Calculating I :

Dept	J	S	I(J,S)
Sales	80	30	0.8484
System	23	8	0.82
Marketing	4	10	0.86
Secretary	6	4	0.96

$Info_{dept}(D) = (0.66)(0.85) + (0.19)(0.82) + (0.08)(0.86) + (0.06)(0.96)$

$Info_{dept}(D) = 0.56 + 0.16 + 0.068 + 0.052 = 0.846$

$Info_{age}(D)$:

Age	J_i	S_i	$I(J_i, S_i)$
31—35	44	35	$I(44,35) = 0.99$
26—30	49	0	$I(49,0) = 0$
21—25	20	0	$I(20,0) = 0$
41—45	0	3	$I(0,3) = 0$
36—40	0	10	$I(0,10) = 0$
46—50	0	4	$I(0,4) = 0$

$Info_{age}(D) = \frac{79}{165}(0.99) + 0 + 0 - \cdots = 0.474$

$Info_{salary}(D)$ =

	J	S	$I(J_i, S_i)$
46k—50k	23	40	0.94
26k—30k	46	0	0
31K—35k	40	0	0
66k—70k	0	8	0
41k—45k	40	0	0
36k—40k	0	4	0

$Info_{salary}(D) = \frac{6}{16}3 \times 0.94 + 0 + \cdots$

$= 0.36$

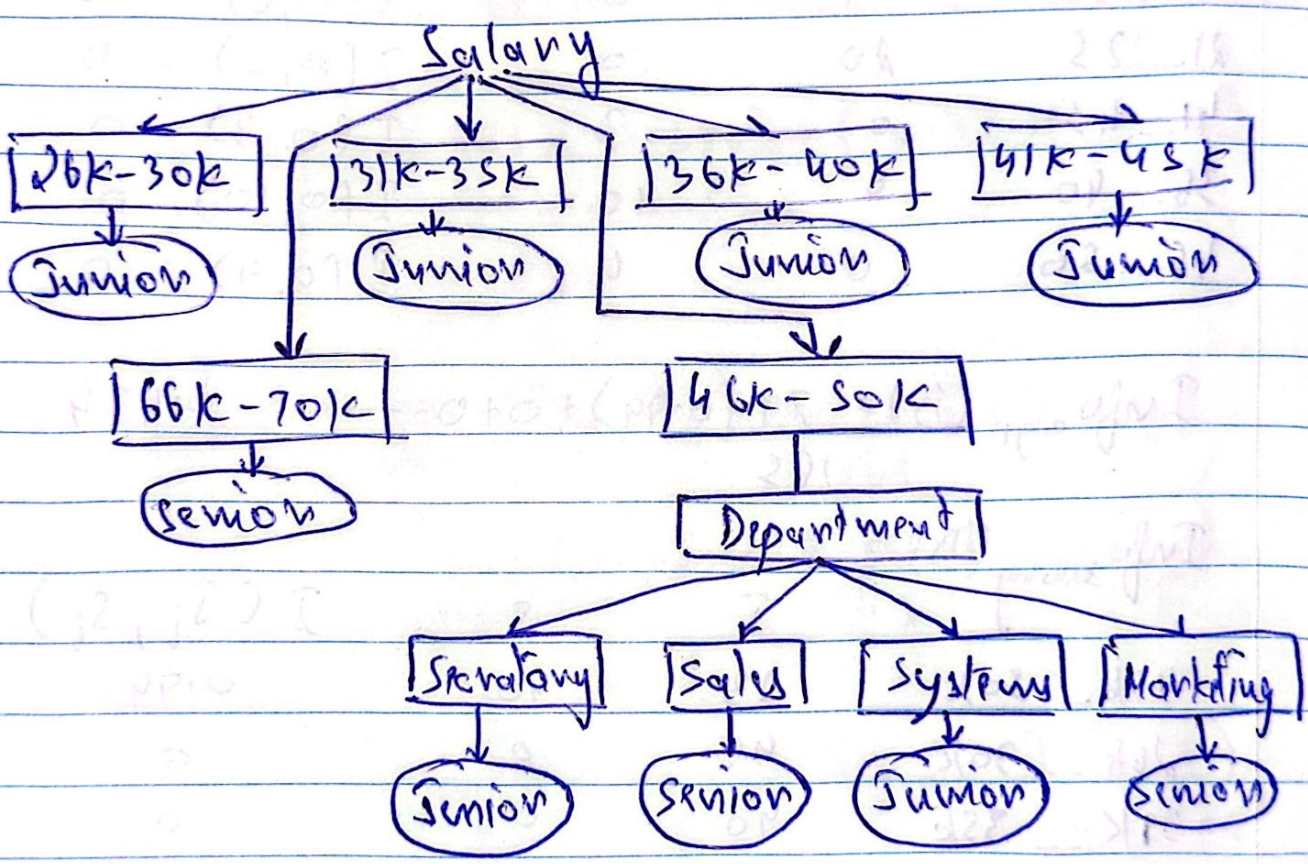
So, Gain →

Gain (Dept) = 0.044

Gain (Age) = 0.41

Gain (salary) = 0.53 } maximum

Similary we can find best attribute for sub division. } So decision Tree :



(c) For sw attributed :

Dept = 'system'

Age = '26-30'

Salary = '46k... 50k'

Using Naive Bayesian for tuple with 2 classes : $P(X|Jr)$, $P(X|Sr)$

$= \frac{23}{113} \cdot \frac{49}{113} \cdot \frac{23}{113}$ (Total Jr = 113)

≈ 0.018

$P(X|Senior) = 0$

so, classification → Junior (same as decision tree)

8.10 Given accuracy = Sensitivity $\left(\frac{P}{P+N}\right)$ + Specificity $\left(\frac{N}{P+N}\right)$

So, accuracy = $\frac{TP+TN}{(TP+FN)+(FP+TN)}$ — (1)

Let's consider $TP+FN = P$

$FP+TN = N$

Sensitivity = $\frac{TP}{TP+FN}$

Specificity = $\frac{TN}{TN+FP}$

from (1)

accuracy = $\frac{TP+TN}{P+N}$

$= \frac{TP}{P+N} + \frac{TN}{P+N}$

$= \frac{TP}{P+N} \times \frac{P}{P} + \frac{TN}{P+N} \times \frac{N}{N}$

Accuracy = Sensitivity $\left(\frac{P}{P+N}\right)$ + Specificity $\left(\frac{N}{P+N}\right)$

Hence Proved.

8.13 From the data tuple and sorted probability value and class labels, need to compute TP, FP, TN, FN, TPR, FPR

Tuple	class	prob.	TP	FP	TN	FN	TPR	FPR
1	p	0.95	1	0	5	4	0.2	0
2	n	0.85	1	1	4	4	0.2	0.2
3	p	0.78	2	1	4	3	0.4	0.2
4	p	0.66	3	1	4	2	0.6	0.2
5	n	0.60	3	2	3	2	0.6	0.4
6	p	0.55	4	2	3	1	0.8	0.4
7	n	0.53	4	3	2	1	0.8	0.6
8	n	0.52	4	4	1	1	0.8	0.8
9	n	0.51	4	5	0	1	0.8	1
10	p	0.40	5	5	0	0	1	1

Calculations :-

So, probability predictions are classified in such a way that for tuple X be $f(x) \rightarrow [0, 1]$ for a binary problem, a threshold t is typically selected so that tuples where $f(x) \geq t$ are considered positive and the remaining other tuples are considered negative thus we can calculate TP, FP with positive class and TN, FN with negative class prediction & actual values.

So for tuple 1 $f(1) \geq 0.85$

Predicted +ve values are → 1

Predicted -ve values are → 9

Actual +ve → 1

Actual -ve → 5

TP = 1 TN = 5

FP = 0 FN = 4

Confusion Matrix →

Actual \ Predicted	P	N
P	1	0
N	4	5

Tuple 2, $f(2) \geq 0.85$

Predicted +ve → 2 Predicted -ve → 8

Actual +ve → 1 Actual -ve → 4

TP = 1 TN = 4 FP = 1 FN 4

Confusion matrix

	Predicted P	Predicted N
Actual P	1	1
Actual N	4	4

Tuple 3, $f(3) \geq 0.78$

Predicted +ve → 3 Predicted -ve = 7

Actual +ve → 2 Actual -ve → 4

TP = 2 FP = 1 TN = 4 FN = 3

Confusion Matrix

A.V \ P.V	P	N
P	2	1
N	3	4

Tuple 4 $f(4) \geq 0.66$

Predicted : $+w \rightarrow 4$ $-w \rightarrow 6$

Actual : $+w \rightarrow 3$ $-w \rightarrow 4$

$TP = 3$ $TN = 4$ $FP = 1$ $FN = 2$

Confusion Matrix

A.V \ P.V	P	N
P	3	1
N	2	4

Tuple 5 $f(5) \geq 0.60$

Predicted : $+w \rightarrow 5$ $-w \rightarrow 5$

Actual : $+w \rightarrow 3$ $-w \rightarrow 3$

$TP = 3$ $TN = 3$ $FP = 2$ $FN = 2$

Confusion Matrix

A.V \ P.V	P	N
P	3	2
N	2	3

Tuple 6 $f(6) \geq 0.55$

Predicted : +ve → 6 -ve → 4

Actual : +ve → 4 -ve → 3

TP = 4 TN = 3 FP = 2 FN = 1

Confusion Matrix

A.V \ P.V	P	N
P	4	2
N	1	3

Tuple 7 $f(7) \geq 0.53$

Predicted : +ve → 7 -ve → 3

Actual : +ve → 4 -ve → 2

TP = 4 TN = 2 FP = 3 FN = 1

Confusion Matrix

A.V \ P.V	P	N
P	4	3
N	1	2

Tuple 8 $f(8) \geq 0.52$

Predicted : +ve → 8 -ve → 2

Actual : +ve → 4 -ve → 1

TP = 4 TN = 1 FP = 4 FN = 1

Confusion Matrix

A.V \ P.V	P	N
P	4	4
N	1	1

Tuple 9 $f(9) \geq 0.51$

Predicted : +ve → 9 −ve → 1

Actual : +ve → 4 −ve → 0

TP = 4 TN = 0 FP = 5 FN = 1

Confusion Matrix

A.V \ P.V	P	N
P	4	5
N	1	0

Tuple 10 $f(10) \geq 0.4$

Predicted : +ve → 10 −ve → 0

Actual : +ve → 5 −ve → 0

TP = 5 TN = 0 FP = 5 FN = 0

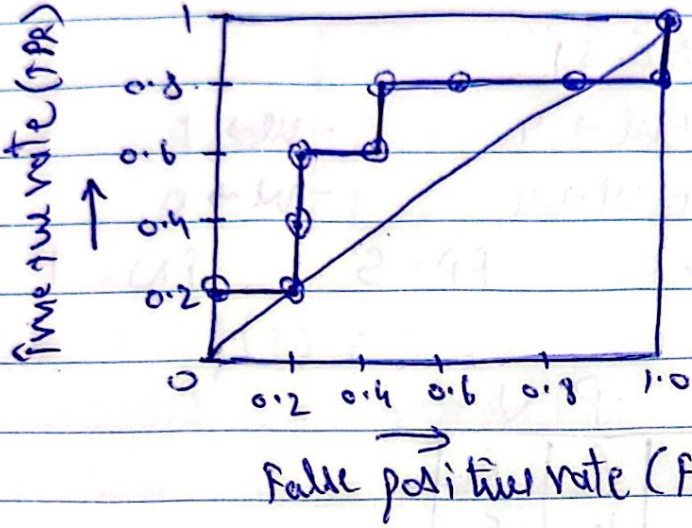
Confusion Matrix

A.V \ P.V	P	N
P	5	5
N	0	0

Then TPR can be calculated with TP/P
↳ total positives

FPR can be calculated with FP/N
↳ total negatives

ROC Curve



8.14 Given, need to select b/w 2 prediction models M_1 & M_2

Error rates obtained for M_1 are 30.5, 32.2, 20.7, 20.6, 31.0, 41.0, 27.7, 26.0, 21.5, 26.0

Error rates obtained for M_2 are 22.4, 14.5, 22.4, 19.6, 20.7, 20.4, 22.1, 19.4, 16.2, 35

$\alpha = 0.01$ $\quad df = n - 1 = 10 - 1 = 9$

$df = 9$

Hypothesis:

$H_0: \bar{x}_1 - \bar{x}_2 = 0$

$H_a: \bar{x}_1 - \bar{x}_2 \neq 0$

$$t \text{ distribution} = \frac{\bar{x}_1 - \bar{x}_2}{s_d/\sqrt{n}}$$

$$= \frac{\bar{d}}{s_d/\sqrt{n}} = \frac{6.45}{8.25\sqrt{10}} = 2.47$$

$t^* = 2.47$

$t_{\alpha/2, n-1} = t_{0.005, 9} = 3.25$

So, $-3.25 < 2.47 < 3.25$

We fail to reject the Null hypothesis

So, there is not enough evidence to say that one model is significantly better than the other.