CSE521 DATA MINING TEST 1 SUMMER 2021
(70pts)

TEAM NUMBER: Leader **NAME** and **ID:**

Team members Leader **NAMES** and **IDs:**

FORMAT OF SUBMISSION

1. The TEAM LEADER submits **ONE PDF** file for the TEAM
2. Name your PDF file as < *TEAMLEADERID* >.pdf

Example: 11384578.pdf

All team members receive the same grade.

To make your **ONE PDF** submission file you proceed as follows

YOU CAN use a sheet of PAPER and HAND-WRITE YOUR SOLUTIONS clearly indicating which problem are you solving (you have to hand write the problem statement and points assigned)

Then you can either **scan** or take **pictures** of the pages and **make ONE PDF file** with multiple pages .

or YOU CAN PRINT the TEST (or use the .pdf editor) and write your answers it in the spaces provided;

Then you **make ONE PDF file** with multiple pages.

or YOU CAN use word / latex to write your answers. In this case, you answers should CLEARLY INDICATE the problem/question (you have to type the problem statement and points assigned)

Then you can either **scan** or take **pictures** of the pages and **make ONE PDF file** with multiple pages.

UNIVERSITY Honesty and Behavioral Expectation

Please read carefully and sign each statement below

1. I certify that the calculations/data/answers in this exam were generated independently, using only the tools and resources defined in the course and that I did not receive any external help, coaching, or contributions during the production of this work.

Names: VENKATA RAVI TEJA TAKKELLA, NAGIREDDY SUSMITHA REDDY, KRISHNA SHASHANK GORANTA

2. I understand the universitys academic integrity and discipline policies and promise to uphold them.

Names: VENKATA RAVI TEJA TAKKELLA, NAGIREDDY SUSMITHA REDDY, KRISHNA SHASHANK GORANTA

3. I understand that the instructor may use tools to check for plagiarism and cheating.

Names: VENKATA RAVI TEJA TAKKELLA, NAGIREDDY SUSMITHA REDDY, KRISHNA SHASHANK GORANTA

ACADEMIC INTEGRITY - UNIVERSITY STATEMENT

Academic integrity is expected of all students at all times, whether in the presence or absence of members of the faculty. Understanding this, I declare that I shall not give, use, or receive unauthorized aid in this examination. I have been warned that any suspected instance of academic dishonesty will be reported to the appropriate office and that I will be subjected to the maximum possible penalty permitted under University Guidelines

QUESTION 1 (5pts)

1. Describe shortly all stages of the Data Mining Process (you can also draw a picture).

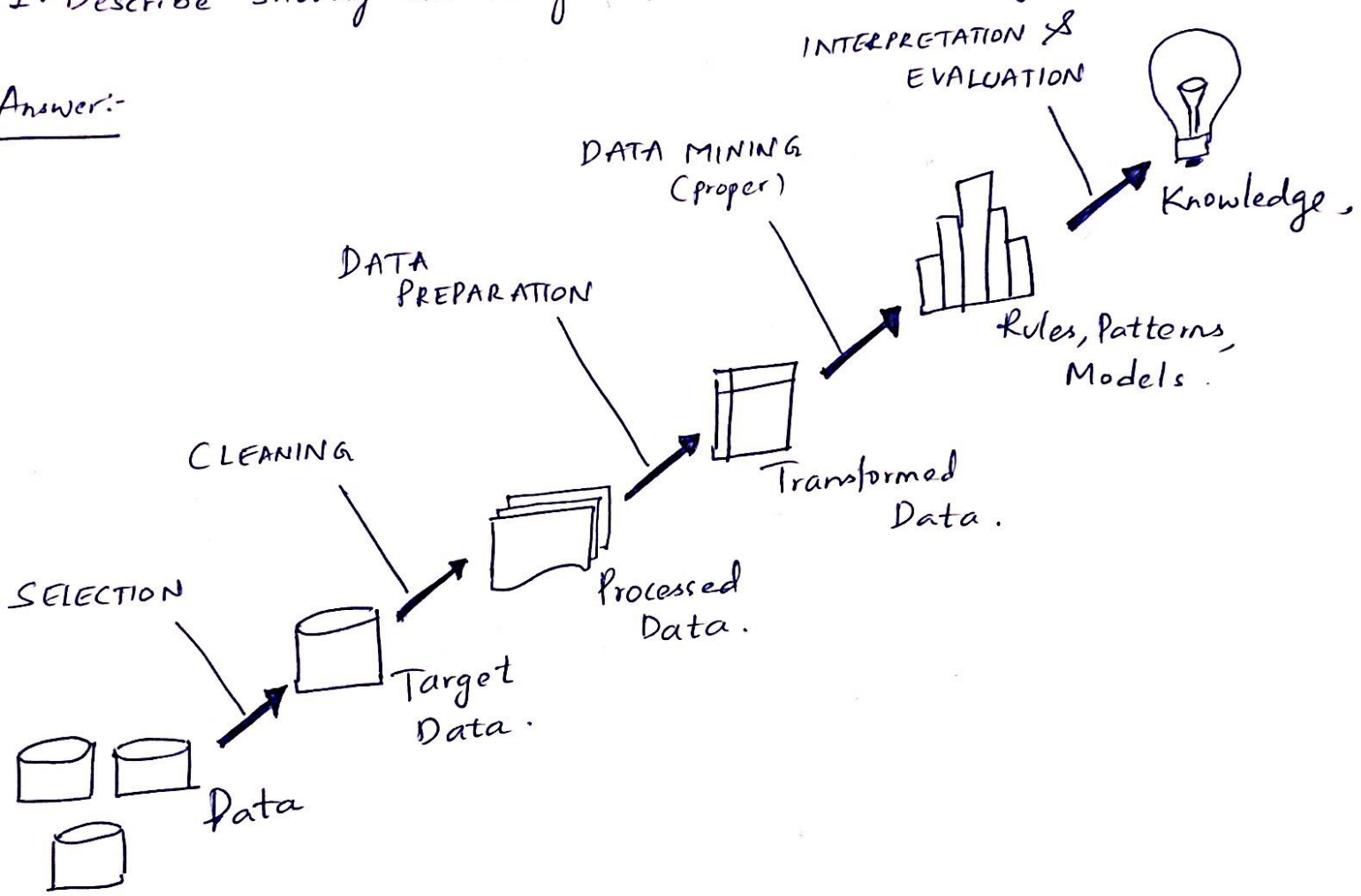
2. When you re-iterate the DM Process?

PART-1

Question1:-

1. Describe shortly all stages of the Data Mining Process?

Answer:-



The Data Mining Process is divided into three stages i.e (i) Preprocessing (ii) Data Mining (proper) (iii) Interpretation

⇒ PRE-PROCESSING: It includes all the operations that needs to be performed before a data mining algorithm is applied. The preprocessing involves data selection,

data cleaning, data integration, data reduction, data transformation and data discretization.

* DATA SELECTION :- In this step, data relevant to the analysis task are retrived from the database.

* DATA CLEANING: This step involves the removal of noisy or incomplete data from the collection. This step carries out the routine cleaning work by Filling in missing values, removing the noisy data [i.e Smooth noisy data by binning, clustering, regression techniques & Identify or Remove Outliers] and also resolve inconsistencies.

* DATA INTEGRATION :- In this step, multiple heterogeneous data sources such as databases, data cubes or files are combined for analysis. This step is helpful in improving the accuracy & speed of DM process.

* DATA REDUCTION :- This technique is applied to obtain relevant data for analysis from the collection of data. The size of representation is much smaller in volume but yet produce the same/almost same analytical results. Some strategies are data compression, dimensionality reduction etc.

* DATA TRASFORMATION: In this step, data are transformed or consolidated into forms appropriate

for mining. Data is consolidated so that the DM process is more efficient and patterns are easier to understand, Techniques such as normilization & aggregation are used here.

* DATA DISCRETIZATION :- This step is mainly important for numerical data. This involves reducing the number of values of attributes.

⇒ DATA MINING PROPER : This is an essential process where inteligent patterns are applied to extract the data patterns. The data is represented in the form of patterns & models are structured using classification & clustering techniques.

⇒ INTERPRETATION & EVALUATION : Visualization & knowledge representation techniques are used to represent the mined data to the user. Pattern evaluation is a process, where we identify the truly interesting patterns representing knowledge based on some interesting measures. The user decides if it is necessary to re-iterate the algorithms.

2. When you re-iterate the DM process?

Answer

→ Data Mining Propeo is a step in Data Mining process, in which algorithms / intelligent patterns are applied to extract the data patterns. The data is represented in the form of patterns and models are structured using classification & clustering techniques.

→ This process is re-iterated inorder to get more efficient results. We can also add new data to re-iterate

→ The results of any one step may indicate that a previous step needs to be refined. For example; Error rate of the pattern recognition step could be high, indicating that the features extracted are not representative enough of the patterns being considered or that the quality of training data could be improved. So, in this case inorder to improve efficiency of results, we re-iterate the DM process.

QUESTION 2 (5pts)

1. Define **classification data**

2. What is a **classifier**

3. Define 4 stages of the process of building a Classifier

Part-1 Question-2.

① Classification Data:

Classfication Data can be defined as a data set, which contains data tuples (or) records from given data along with their associated class labels.

Classification Data format:

- The Data table of classification Data has its key attribute removed.
- A special attribute, called as class attribute must be distinguished.
- The values of the class attribute are called as class labels.
- Class attributes are categorical, each value serves as a category, or a class.
- The class labels are discrete and unordered.
- A data tuple (or) record in classification data has attribute part and class part.
- The attribute part is called data tuple, or

attribute vector, data vector, sample, example, instance, data point (with associated class label).

② Classifier:

A classifier is a black-box that is used to classify records for which the class label is unknown.

A classifier is a final product of a learning process that uses a classification dataset and a classification algorithm.

- A classifier is the end result of a process that uses training data and testing data and a classification algorithm to generate patterns that can classify new data.

The need for refinement, effectiveness and completion of a classifier is determined based on its predictive accuracy.

- We terminate the process of building a classifier if it has been trained and tested and the predictive accuracy is an acceptable level.

- The patterns that a classifier uses take different forms,

such as rules of trees, trained networks like neural or Bayesian.

(3) Building a Classifier:

Two data sets, namely training data set and testing data set have to be created inorder to build a classifier. These two data sets must be disjoint sets.

Training and Test data may differ in nature but must have same format.

Stages of building a classifier:

Stage 1 - Training:

In this stage, a classification algorithm uses the training data to build a basic structure for set of patterns.

These patterns can be called as discriminant and/or Characteristic rules or other pattern descriptors.

This structure can be called as basic or learned classifier (or) model.

Using these patterns we can classify unknown sets of objects (unknown tuples / records).

Stage 2 - Parameter tuning:

After building a basic or learned classifier, the parameter settings need to be optimized.

By testing the classification model using a dataset which is already used for training called as validation dataset, an error rate can be seen.

For example: Resubstitution (N:N) method.

This error rate expresses how good or bad the results are on the training data. It reflects the imprecision of training results.

Lower the error rate better will be the classifier.

Stage 3 - Testing:

In this stage, test data is used to evaluate the classifier which was built by predicting the class values of it.

Predictive Accuracy:

Predictive Accuracy of the model is the percentage of well classified data in testing data set.

Predictive accuracy gets computed in this stage.

If the predictive accuracy is higher the the model is good.

Stage 4 - Consolidation:

In this stage, consolidation of stages 1, 2 and 3 occurs in order to build a Classifier as the final product.

QUESTION 3 (5pts)

1. Define Holdout and Repeated Holdout (Random Sampling)

2. Define k-fold cross-validation (N- N/k ; N/k) holdout and repeated k-fold cross-validation holdout

3. Define Leave-one-out (N-1 ; 1) holdout

4. Bootstrap Holdout

5 .632 boostrap Holdout

PART-1

Question-3

(1) Define Holdout & Repeated Holdout ?

(A) These are two types of methods to evaluate the classifiers. these methods help to bring out the predictive accuracy of the classifier.

Holdout: In this method, the provided data is seperated into two independent disjoint sets. Typically, $\frac{2}{3}^{rd}$ of data is allocated for training & the rest $1/3^{rd}$ data for testing. So, the training data is used to train & build the classifier where as the test data is used to estimate the accuracy known as predictive accuracy. This estimate is pessimistic since the entire data is not used for the testing phase.

Repeated Holdout: This method is similar to the Holdout method but just that it's repeated for K times. The partioning of the data is done randomly ie the probability of any data point to be picked is exactly the same. Since this method is performed randomly, there might be cases where few data points might not be entirely present in the training phase. Finally when the Holdout testing/validation is performed for K-times (generally given), the overall accuracy is obtained as the average of all the accuracies obtained after each iteration.

(2) Define K-fold cross validation holdout & repeated K-fold cross validation holdout.

K-fold cross validation: This is also a method which is commonly used to evaluate a classifier. Over here, the initial data is divided randomly into K (given) disjoint sets (mutually exclusive). So, each datapoint is observed only once across all the subsets generated. In this method, there are K iterations & for each iteration, one of the K subsets is considered without repetitions as the test data. Once the test data is selected, rest of the subsets together form the training data. Since no. of subsets is K. for any iteration training data = N - N/K & test data = N/K where N is the initial number of data tuples. The accuracy estimate calculated is the total number of correct classifications across all the iterations divided by the

initial number of data points. Unlike as in "Holdout" method, each data point or data tuple is used for training & testing phase of the classifier. To be exact, each data tuple is in training phase for K-1 iterations & in testing phase just once. Most suggested number of folds is 10.

Repeated K-fold cross validation holdout:

It's the same procedure as K-fold but repeated multiple times (generally given). As, we seperate the datasets initially for K-fold; the predictive accuracy is totally dependent on this initial subsampling. This may turn out as bias & noisy estimate of the model performance. Different subsets may bring out different estimates. So Repeated K-fold repeats the whole procedure multiple times & estimates the accuracy as the mean estimate of each K-fold accuracy.

If the k-fold algorithm is repeated for n times. The total number of classifiers built are n * k. So this method is generally used for small or medium sized datasets.

(3) Leave One out holdout :

This is a special case of K-fold cross validation where K is taken as the initial number of data tuples. This means for each iteration, the testing data contains only 1 data tuple & the rest N-1 data tuples are considered as training data. This in general provides an unbiased accuracy estimate of the classifier but computationally expensive. So it is generally used for smaller sized datasets.

(4) Boot strap Holdout:

Unlike the methods mentioned above, this method subsamples the training data tuples with replacement. This means if a data point is selected randomly for training data, there's again same probability of getting selected for the data tuple similar to the first one. If we fix on number of data tuples to sample N. Then we need to select N data tuples (might not be unique) randomly from the data with repetition. The data tuples which finally do not belong to the training data turn to be the test data which can be used for predicting the accuracy. This method can be repeated multiple times till we get a meaningful statistics of accuracies. The final predictive accuracy would be the mean estimate of all the iterations.

(5) .632 Bootstrap Holdout :

This is a bootstrap method where we sample exactly the initial number of data tuples as training set. So, if the initial number of data tuples are d. Then we randomly select d data tuples from the data with repitition. Finally the data tuples which are not part of the training data will be the test dataset.

So, in this method the probability of a data point being selected is $1/d$ & the probability for not being selected is $1 - 1/d$. Since we need to do this d times, the probability of a data point not being picked in the whole phase is $(1 - 1/d)^d$. If d is large, this approaches to $e^{-1} = 0.368$. So there are 36.8% of data tuples which never come up in training data & be the test set & the remaining 63.2% will form the training set.

If we repeat this algorithm for k times. The final predictive accuracy would be.

Accuracy of Data D

$$= \sum_{i=1}^{k} (0.632 \times \text{Accuracy of } D_i \text{ bootstrap test set} + 0.368 \times \text{Accuracy of } D_i \text{ bootstrap training set}).$$

So because of the probability 0.632 of being picked in training dataset, this method is called as .632 bootstrap holdout. This works well for smaller datasets

PROBLEM 1 (15pts)

1. Classification Rules (5pts)

For the following **formulas** write and use the proper definitions to prove whether they are or they are not **discriminant** or **characteristic rules** in the following dataset DB

CLASSIFICATION DB

O	a1	a2	a3	a4	C
o1	1	1	1	0	1
o2	2	1	2	0	2
o3	0	0	0	0	0
o4	0	0	2	1	0
o5	2	1	1	0	1

Characteristic Rule Definition

Discriminant Rule Definition

Formulas

f1 $a1 = 1 \cap a2 = 1 \Rightarrow C = 1$

f1 is / is not Discriminant Rule because

f2 $C = 1 \Rightarrow a1 = 0 \cap a2 = 1 \cap a3 = 1$

f2 is / is not Characteristic Rule because

f3 $a1 = 1 . \Rightarrow C = 1$

f3 is / is not Discriminant Rule because

f4 $C = 1 \Rightarrow a1 = 1$

f4 is / is not Characteristic Rule because

f5 $a1 = 2 \cap a2 = 1 \cap a3 = 1 \Rightarrow C = 0$

f5 is / is not Characteristic Rule because

2. (5pts)

Prove that in any classification DB the inverse implication to the discriminant rule is a characteristic rule

3. (5pts)

Given classification DB

Find a simple condition (example) under which the inverse implication to a characteristic rule

is ALWAYS a discriminant rule

PART-2

PROBLEM-1:

1. Given :-

O	a_1	a_2	a_3	a_4	C
o_1	1	1	1	0	1
o_2	2	1	2	0	2
o_3	0	0	0	0	0
o_4	0	0	2	1	0
o_5	2	1	1	0	1

CHARACTERISTIC-RULE Definition :- A characteristic formula CLASS ⇒ DESCRIPTION is called a "Characteristic Rule" of the classification dataset DB if it is "true" in DB, i.e when the following holds :-

$$\{o : \text{DESCRIPTION}\} \cap \{o : \text{CLASS}\} \text{ not} = \text{empty set}$$

where $\{o : \text{DESCRIPTION}\}$ is a set of all records of DB corresponding to the description "DESCRIPTION" & $\{o : \text{CLASS}\}$ is the set of all records of DB corresponding to the description "CLASS"

DISCRIMINANT-RULE Definition :- A discriminant formula DESCRIPTION ⇒ CLASS" is called a "Discriminant Rule" of DB, if it is "True" on DB, i.e the following two conditions

hold:-

(i) $\{O: \text{DESCRIPTION}\}$ not = empty set

(ii) $\{O: \text{DESCRIPTION}\}$ included in $\{O: \text{CLASS}\}$

(i) $a_1 = 1 \cap a_2 = 1 \Rightarrow C = 1$

DESCRIPTION $\Rightarrow$ CLASS

$a_1 = 1$

Record	a_2	a_3	a_4	C
O_1	1	1	0	1

$a_1 = 1 \cap a_2 = 1$

Record	a_3	a_4	C
O_1	1	0	1

$C = 1$

Record	a_1	a_2	a_3	a_4
O_1	1	1	1	0
O_5	2	1	1	0

$\{O: a_1 = 1 \wedge a_2 = 1\} = \{O_1\}$

$\Rightarrow \{O: \text{DESCRIPTION}\}$ not empty set

$\{O: C = 1\} = \{O_1, O_5\}$

$\{O: a_1 = 1 \cap a_2 = 1\} = \{O_1\} \subseteq \{O: C = 1\} = \{O_1, O_5\}$

$\Rightarrow \{O: \text{DESCRIPTION}\} \subseteq \{O: \text{CLASS}\}$

$\therefore$ Since $\{O_1\}$ is a subset of $\{O_1, O_5\} \Rightarrow$ So this is "DISCRIMINANT" rule

Answer.

(ii) $C = 1 \Rightarrow a_1 = 0 \cap a_2 = 1 \cap a_3 = 1$

CLASS $\Rightarrow$ DESCRIPTION

$C = 1$

Record	a_1	a_2	a_3	a_4
o_1	1	1	1	0
o_5	2	1	1	0

$a_1 = 0$

Record	a_2	a_3	a_4	C
o_3	0	0	0	0
o_4	0	2	1	0

$a_1 = 0 \cap a_2 = 1$

None

$a_1 = 0 \cap a_2 = 1 \cap a_3 = 1$

None

$\{o : a_1 = 0 \cap a_2 = 1 \cap a_3 = 1\} = \phi$

$\{o : C = 1\} = \{o_1, o_5\}$

$\{o : a_1 = 0 \cap a_2 = 1 \cap a_3 = 1\} = \phi \cap \{o : C = 1\} = \{o_1, o_5\} = \phi$

$\{o : \text{DESCRIPTION}\} \cap \{o : \text{CLASS}\} = \phi$

$\therefore$ since $\{o : a_1 = 0 \& a_2 = 1 \& a_3 = 1\}$ is a empty set, so this "not a CHARACTERISTIC" rule.

Answer

(iii) $a_1 = 1 \Rightarrow C = 1$

DESCRIPTION $\Rightarrow$ CLASS

$a_1 = 1$

Record	a_2	a_3	a_4	C
O_1	1	1	0	1

$C = 1$

Record	a_1	a_2	a_3	a_4
O_1	1	1	1	0
O_5	2	1	1	0

$\{O : a_1 = 1\} = \{O_1\}$

$\Rightarrow \{O : \text{DESCRIPTION}\}$ not empty set

$\{O : C = 1\} = \{O_1, O_5\}$

$\{O : a_1 = 1\} = \{O_1\} \subseteq \{O : C = 1\} = \{O_1, O_5\}$

$\Rightarrow \{O : \text{DESCRIPTION}\} \subseteq \{O : \text{CLASS}\}$

$\therefore$ Since $\{O_1\}$ is a subset of $\{O_1, O_5\} \Rightarrow$ This is a "DISCRIMINANT RULE"

Answer.

(iv) $C=1 \Rightarrow a_1 = 1$

CLASS $\Rightarrow$ DESCRIPTION

$C=1$

Record	a_1	a_2	a_3	a_4
O_1	1	1	1	0
O_5	2	1	1	0

$a_1 = 1$

Record	a_2	a_3	a_4	C
O_1	1	1	0	1

$\{O : a_1 = 1\} = \{O_1\}$

$\{O : C = 1\} = \{O_1, O_5\}$

$\{O : a_1 = 1\} = \{O_1\} \cap \{O : C = 1\} = \{O_1, O_5\} = \{O_1\}$

$\Rightarrow \{O : \text{DESCRIPTION}\} \cap \{O : \text{CLASS}\}$ not = empty set

$\therefore$ Since $\{O_1\} \cap \{O_1, O_5\} = \{O_1\}$ which is not empty set

$\therefore$ This is a CHARACTERISTIC Rule

Answer.

(v) $a_1 = 2 \cap a_2 = 1 \cap a_3 = 1 \Rightarrow C = 0$

DESCRIPTION $\Rightarrow$ CLASS

$a_1 = 2$

Record	a_2	a_3	a_4	C
O_2	1	2	0	2
O_5	1	1	0	1

$a_1 = 2 \cap a_2 = 1 \Rightarrow \{O_2, O_5\}$.

$a_1 = 2 \cap a_2 = 1 \cap a_3 = 1 \Rightarrow \{O_5\}$

$C = 0$

Record	a_1	a_2	a_3	a_4
O_3	0	0	0	0
O_4	0	0	2	1

$\{O: a_1 = 2 \cap a_2 = 1 \cap a_3 = 1\} = \{O_5\}$

$\Rightarrow \{O:$ DESCRIPTION$\}$ not empty set

$\{O: C = 0\} = \{O_3, O_4\}$

$\Rightarrow \{O: a_1 = 2 \cap a_2 = 1 \cap a_3 = 1\} = \{O_5\}$ is not a subset of

$\{O: C = 0\} = \{O_3, O_4\}$.

$\therefore$ Since $\{O_5\}$ is not subset of $\{O_3, O_4\}$, This is "not a DISCRIMINANT" rule

Answer.

And since the format is wrong i.e DESCRIPTION ⇒ CLASS, It is "not a "characteristic Rule". For the Characteristic Rule, the format should be CLASS ⇒ DESCRIPTION.

∴ This is neither CHARACTERISTIC RULE (due to wrong format) nor DISCRIMINANT RULE (due to conditions not satisfying)

Answer.

2. Prove that in any classification DB, the inverse implication to the discriminant rule is a characteristic rule ??

Answer:-

By definition, for any database DB:

DESCRIPTION $\Rightarrow$ CLASS

is a discriminant rule iff the below conditions holds true:

1) $\{O: \text{DESCRIPTION}\}$ is not empty

2) $\{O: \text{DESCRIPTION}\}$ is included in $\{O: \text{CLASS}\}$

We know that for any non-empty sets A, B, if A is included in B, then their intersection is non-empty.

Hence, $\{O: \text{DESCRIPTION}\}$ intersection with $\{O: \text{CLASS}\}$ is not empty and by definition, inverse application i.e,

CLASS $\Rightarrow$ DESCRIPTION

is a CHARACTERISTIC RULE

Example

RULE: $a_1 = 1 \Rightarrow c = 1$

DESCRIPTION $\Rightarrow$ CLASS

Since $\{O: a_1 = 1\} = \{O_1\}$ is a subset of $\{O: c = 1\} = \{O_1, O_3\}$, This is a Discriminant Rule

DB

O	a_1	a_2	c
O_1	1	1	1
O_2	2	1	2
O_3	2	1	1

INVERSE IMPLICATION:- $c = 1 \Rightarrow a_1 = 1$

[CLASS $\Rightarrow$ DESCRIPTION]

Since $\{O_1\} \cap \{O_1, O_3\} = \{O_1\}$ which is not empty set, It is a Characteristic Rule

Hence proved

3. Given a classification DB, find a simple condition, under which inverse implication to a characteristic rule is always a discriminant rule.

Answer: By definition, for any database DB:

CLASS ⇒ DESCRIPTION

is a characteristic rule, if below condition holds good:

1) {O: DESCRIPTION} ∩ {O: CLASS} not = empty set

We know that for any sets A, B, if A ∩ B is not an empty set, then neither of A and B are empty sets

⇒ A and B are non-empty sets. ——— condition (1)

For inverse implication rule ⇒ DESCRIPTION ⇒ CLASS to be a DISCRIMINANT RULE, as per the definition, we have two conditions to be satisfied, i.e

Condition 1: {O: DESCRIPTION} is not empty

Condition 2: {O: DESCRIPTION} is included in {O: CLASS}

From (1), condition 1 is already met. Therefore, condition 2 needs to be satisfied for this to be true.

⇒ Inverse implication to a characteristic rule is always a Discriminant Rule if {O: DESCRIPTION} is included in {O: CLASS}.

Example:-

DB

O	a_1	a_2	C
O_1	1	1	1
O_2	2	1	2
O_3	2	1	1

Rule: $C=1 \Rightarrow a=1$ is a characteristic Rule since.

$\{O: C=1\} = \{O_1, O_3\} \cap \{O: a=1\} = \{O_1\} = \{O_1\}$ is not empty

Inverse Implication: $a=1 \Rightarrow C=1$ also holds good here, i.e

It is a Discriminant rule because. $\{O: a=1\} = \{O_1\}$ is included in $\{O: C=1\} = \{O_1, O_3\}$

$\therefore \{O: \text{DESCRIPTION}\} \subseteq \{O: \text{CLASS}\}$ For inverse implication of Characteristic to be always a discriminant rule.

PROBLEM 2 DECISION TREES (30pts)]

Part 1 (5pts)

1. List and explain shortly the Decision Tree Algorithm Attribute Selection Measures.

2. Describe in your words what is their role in the Decision Tree Construction and which kind of trees they produce". Draw a picture as an example.

Part 2 (25pts) BUILDING DECISION TREE CLASSIFIER

Given Classification DB

O	a1	a2	C
o1	1	1	1
o2	0	0	0
o3	0	1	0
o4	0	0	0
o5	1	1	1
o6	1	1	0
o7	0	0	0
o8	1	0	1

Use the above DB and **repeated two fold cross validation holdout** to build a CLASSIFIER using the DT BASIC Algorithm with **a1** as the root. Follow **Stages 1- 4** of the process of building a classifier.

Remember that division into 2-folds is an ARBITRARY partition of records into 2 disjoint sets; so there may be many answers depending on the partitions.

Repeat the two fold cross validation holdout 4 times.

Build your final CLASSIFIER using the following folds for each repetitions round **1.- 4.**.

1. **f1** = $\{o1, o2, o3, o4\}$ for training - rest for testing.
2. **f2** = $\{o5, o6, o7, o8\}$ for training - rest for testing.
3. **f2** = $\{o1, o3, o5, o7\}$ for training - rest for testing.
4. **f2** = $\{o2, o4, o6, o8\}$ for training - rest for testing.

Here are the **STEPS** you must follow

STEP 1 (10pts)

Follow **Stages 1-3** to build, for each repetitions round **1.- 4.**, a learned classifier (base classifier, learned model) and name them **F1** , **F2** , **F3** , **F4** , respectively.

Write the learned classifiers **F1 - F4** in as a set of **discriminant rules** in the predicate form.

STEP 2 (10pts)

Perform the **Stage 4** as follows: use the learned classifiers **F1 - F4** and their predictive accuracy and rules accuracy as metrics to choose ONE as your final CLASSIFIER **F**

STEP 3 (5pts)

Construct as your final CLASSIFIER the **bagged** Ensemble Classifier $\mathbf{F}^*$

Use your CLASSIFIERS **F** and $\mathbf{F}^*$ to classify the following records and compare the results.

O	a1	a2
o1	0	1
o2	0	0
o3	1	0
o4	1	1

Problem-2 Decision Trees.

Part-1.

① Decision Tree Algorithm Attribute Selection measures:

Given a training data set, there are many ways to choose the root and nodes attributes while constructing a decision tree. These methods of choosing attributes are called Attribution selection measures.

Some possible choices of selecting attributes can be Random, Attribute with Smallest/Largest number of values etc.

The three attribute selection methods that give good results are:

i) Information Gain:

In this measure we have a special order i.e information gain as a measure of goodness of split. The attribute with highest information gain is always chosen as the split decision attribute for the current node while building a tree.

Let p_i be the probability that the arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$

- Expected information (entropy) needed to classify a tuple in D is

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

- Information needed (after using A to split D into v partitions) to classify D is

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)$$

- Information gained by branching on attribute is

$$Gain(A) = Info(D) - Info_A(D)$$

For continues-valued attribute A, to determine the best split point for A:

- sort the value A in increasing order.
- Typically, the midpoint between each pair of adjacent values is considered as possible split point.
- The point with the minimum expected information requirement for A is selected as split point for A.

- This attribute with highest information gain is selected as splitting attribute.

2) Gain Ratio:

Information gain attribute selection measure is biased towards attributes with a large number of values. Gain ratio is used to overcome this problem.

Gain ratio simply applies normalization to information gain.

$$\text{GainRatio}(A) = \text{Gain}(A) / \text{SplitInfo}(A)$$

$$\text{SplitInfo}_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

The attribute with maximum gain ratio is selected as splitting attribute.

3) Gini index:

If a data set D contains examples from n classes, gini index, gini(D) is defined as

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

where p_j is the relative frequency of class j in D

- If a dataset D is split on A into two subsets D_1 and D_2, the giniIndex gini(D) is defined as

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- Reduction in impurity $\Delta gini(A) = gini(D) - gini_A(D)$
- The attribute that provides smallest $gini_{split}(D)$ i.e (the largest reduction in impurity) is chosen to split the node.
- This approach is biased to multivalued attributes.
- When the number of classes increases, this approach faces difficulty. It also favors tests that result in equal sized partitions.

Some other attribute selection measures are:

CHAID - The measure is based on χ^2-test.

C-SEP - This measure sometimes may perform better than information gain and gini index.

G-statistics - This gives close approximation to x^2-distribution.

MD2 (Minimal Description Length) - This measure prefers the simplest solution.

CART - This measure will perform multivariant split based on linear combination of attributes

Most of the attribute selection methods give good results.

② Splitting Criterion:

Splitting Criterion tells us which attributes to test at Node N by determining the "best" way to separate or partition the tuples in D into individual classes.

The splitting criterion indicates the splitting attribute may also indicate either a

split point or a splitting subset.

- Using any of the three attribute selection measures out of Information Gain, Gain Ratio and Gini Index, the splitting criterion can be determined.
  - The kind of tree that gets produced while building the decision tree depends on the type of data that exists in the splitting attribute.
  - There are three scenarios on the type of data that can present for an attribute.

  i) Discrete valued:

  In this case, the outcomes of the test at Node N correspond directly to the known values of A. A branch is created for each known value a_j, of A and labeled with that value.

  ii) Continuous valued:

  In this case, the test at node N has two possible outcomes, corresponding to the conditions $A \leq$ split-point and $A >$ split-point, where

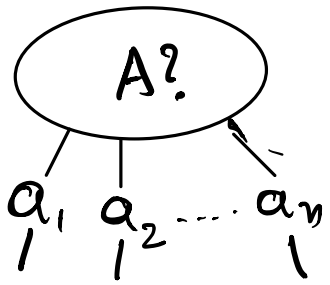
split_point is the split-point returned by Attribute_selection_measure as part of splitting criterion.

iii) Discrete valued & Binary tree must be produced:

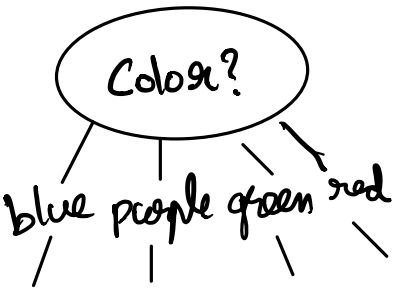
In this case the test at node N is of the form "$A \in S_A$?", where S_A is splitting subset of A, returned by Attribute selection measure as part of splitting criterion. Two branches are grown from N.

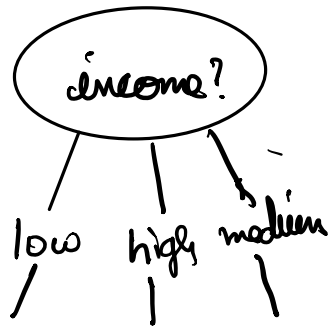
Partitioning scenarios | examples.

(a)

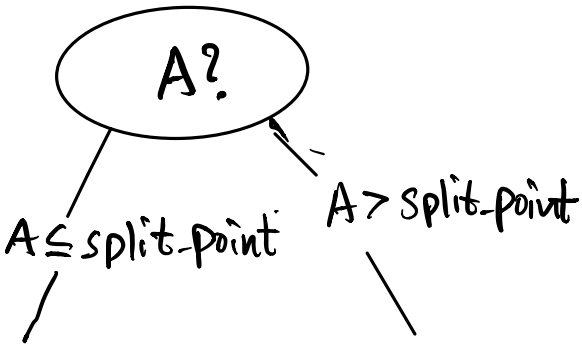


(discrete valued)

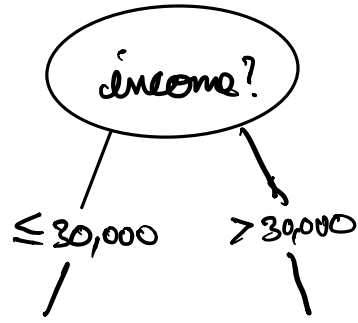




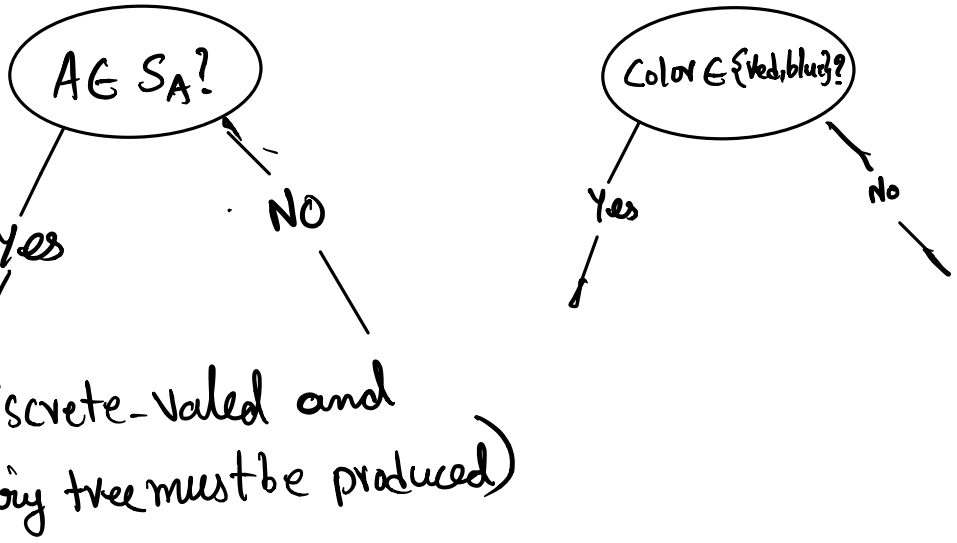
(b)



(Continuous valued)



(G)



Detailed examples for various attribute selection measures are described below:

Information Gain:

In Information Gain type of attribute selection measure, the attribute with highest information gain is always chosen as the split decision attribute for the current node while building a tree.

Information gain measure is biased towards multivalued attributes.

The decision tree that gets built using this measure need not be a binary tree or a skewed tree to one side.

Example:

RID	age	income	Student	Credit_rating	Class: buys_Comp
1.	Youth	high	no	fair	no
2.	Youth	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.	Youth	medium	no	fair	no
9.	Youth	low	yes	fair	yes
10.	senior	medium	yes	fair	yes
11.	Youth	medium	yes	excellent	yes
12.	middle-aged	medium	no	excellent	yes
13.	middle-aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Computing gain on each attribute:

$$Info_{age}(D) = \frac{5}{14} \times \left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right)$$

$$+ \frac{4}{14} \times \left(-\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4}\right)$$

$$+ \frac{5}{14} \times \left(-\frac{3}{5}\log_2 \frac{3}{5} - \frac{2}{5}\log_2 \frac{2}{5}\right)$$

$$= 0.694$$

$$\text{Info}(D) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940 \text{ bits}$$

$$\therefore \text{Gain}(age) = \text{Info}(D) - \text{Info}_{age}(D) = 0.940 - 0.694$$
$$= 0.246 \text{ bits}$$

Similarly, calculating gain for other attributes we get

$$\text{Gain}(income) = \text{Info}(D) - \text{Info}_{income}(D)$$

$$= 0.029 \text{ bits}$$

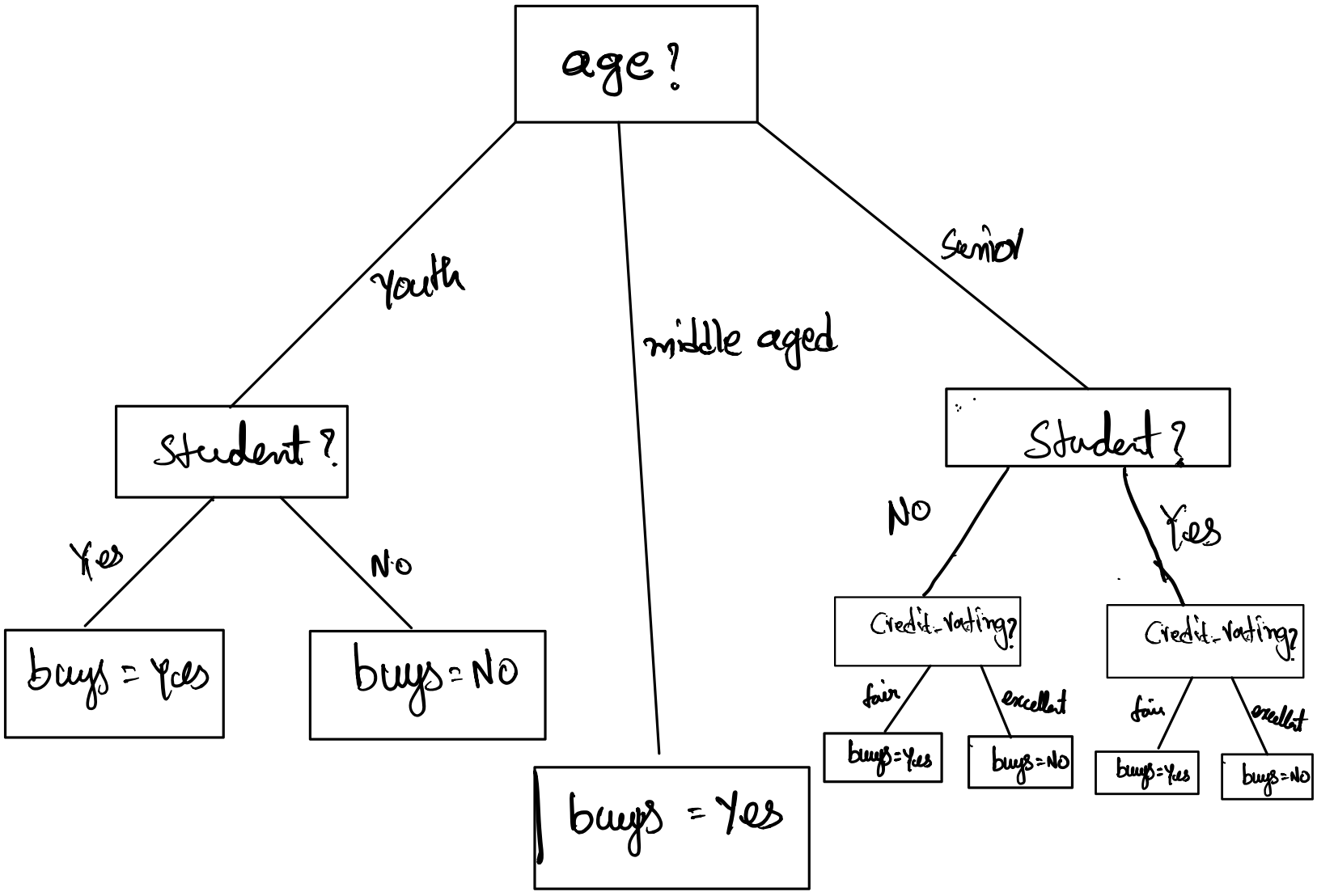
$$\text{Gain}(student) = \text{Info}(D) - \text{Info}_{student}(D)$$

$$= 0.151 \text{ bits}$$

$$\text{Gain}(credit\_rating) = \text{Info}(D) - \text{Info}_{credit\_rating}(D)$$

$$= 0.048 \text{ bits.}$$

Since the age attribute has the highest information gain and therefore becomes the splitting attribute at the root node of decision tree.



Gain Ratio:

In this measure, the attribute with maximum gain ratio is selected as the splitting attribute for the current node.

- Gain Ratio tends to prefer unbalanced splits in which one partition is much smaller than others.

For the previous table, calculating the Gain Ratio's to determine the split:

From the previous example we know

$$\text{Gain (income)} = 0.029$$

$$\text{Gain (age)} = 0.246$$

$$\text{Gain (student)} = 0.151$$

$$\text{Gain (credit\_rating)} = 0.048.$$

$$\text{SplitInfo}_{(\text{income})}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \log_2\left(\frac{4}{14}\right)$$

$$= 1.557$$

$$\text{Gain Ratio (income)} = 0.029/1.557 = 0.019.$$

$$\text{SplitInfo}_{(\text{age})}(D) = -\frac{5}{14} \log_2\left(\frac{5}{14}\right) - \frac{4}{14} \log_2\left(\frac{4}{14}\right) - \frac{5}{14} \log_2\left(\frac{5}{14}\right)$$

$$= 0.5305 + 0.5163 + 0.5305$$

$$= 1.5773$$

$$\therefore \text{Gain Ratio (age)} = 0.246/1.5773$$

$$= 0.1559.$$

$$\text{SplitInfo}_{(\text{student})}(D) = -\frac{1}{2}(-1) - \frac{1}{2}(1)$$

$$= 1$$

$$\therefore \text{Gain Ratio (student)} = 0.151/1 = 0.151$$

$$\text{SplitInfo}_{\text{Credit\_rating}}(D) = -\frac{8}{14}\log_2(8/14) - \frac{6}{14}\log_2(6/14)$$

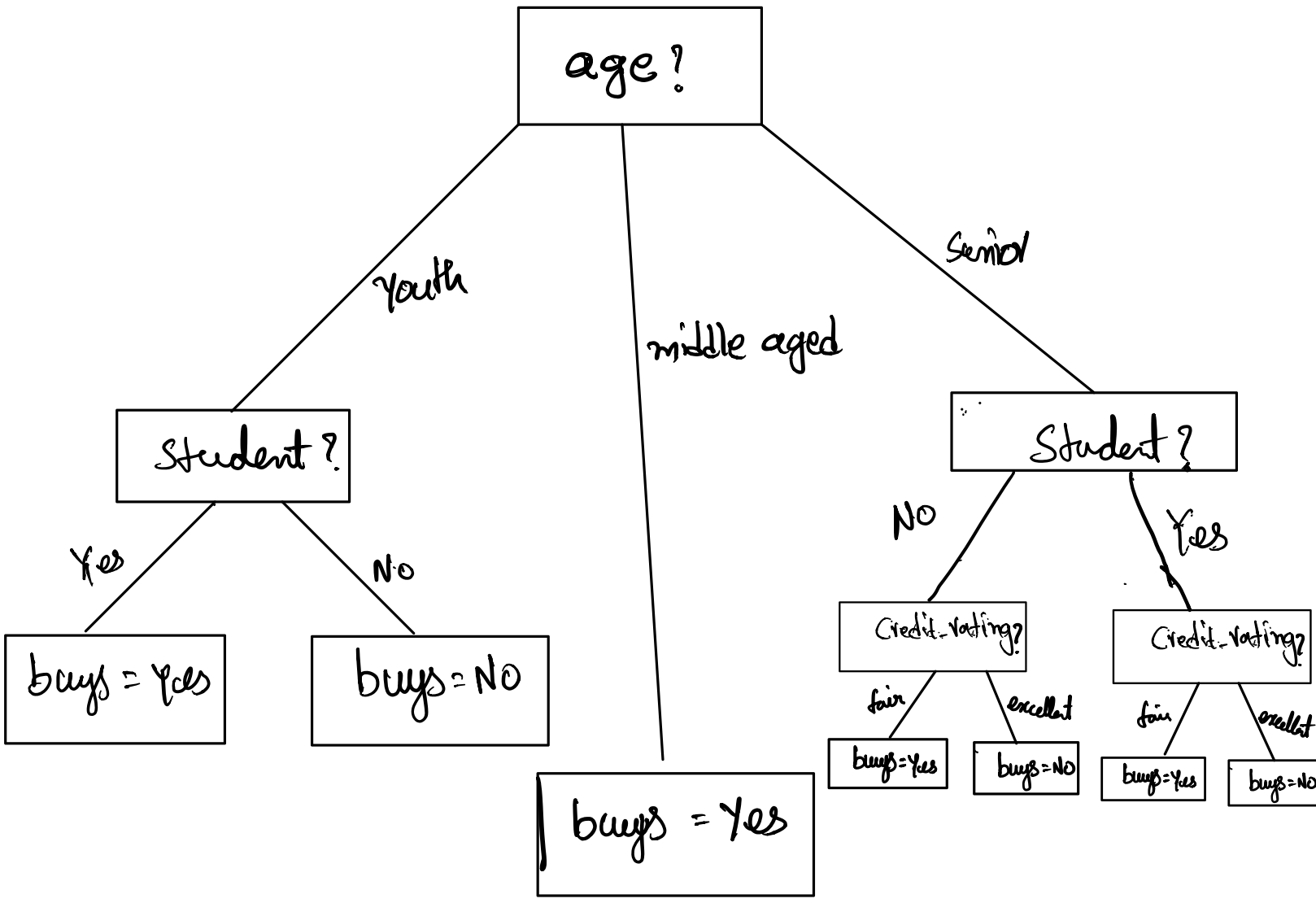
$$= 0.4613 + 0.5238$$

$$= 0.9851$$

$$\text{Gain Ratio (Credit\_rating)} = 0.048/0.9851$$

$$= 0.0487.$$

We see that highest Gain ratio is for age, followed by student & then followed by Credit_rating and then by income.



The order of attributes for splitting remained same for Information Gain and Gain Ratio, so the resulting tree looks similar.

Gini Index:

The attribute that provides smallest $gini_{split}(D)$ i.e (the largest reduction in impurity) is chosen to split the node.

- Gini Index is biased to multivalued attributes
- It has difficulty when number of classes is large.
- tends to favor tests that result in equal sized partitions and purity in both partitions.
- It forms a binary tree.

Example:

A1	A2	Class C
3.1	1.6	Yes

2.3	1.5	Yes
2.8	4.7	NO
3.1	4.6	NO
3.4	1.6	Yes
3.5	1.5	No.

Selecting random values for each attribute to calculate Gini Index.

for A1 A2

>=3 >=4

<3 <4

Calculating Gini Index for A1:

Attribute A1 >=3 & Class Yes: 2/6

A1 >=3 & Class No: 2/6

$$\text{Gini}(2,2) = 1 - \left[\left(\frac{2}{6}\right)^2 + \left(\frac{2}{6}\right)^2\right]$$

$$= 0.777$$

Attribute $A_1 < 3$ & class Yes : 1/6

$A_1 < 3$ & class No : 1/6

$$\text{Gini}(1,1) = 1 - \left[\left(\tfrac{1}{6}\right)^2 + \left(\tfrac{1}{6}\right)^2\right]$$

$$= 0.944$$

By adding weight and sum each of gini indices :

$$\text{gini}(\text{Target}, A_1) = \left(\frac{4}{16}\right) \times (0.777) + \left(\frac{2}{16}\right) \times (0.944)$$

$$= 0.194 + 0.118$$

$$= 0.312.$$

Calculating Gini Index for A2 :

Attribute $A_2 >= 4$ & Class Yes : 0/6

$A_2 >= 4$ & Class No : 2/6

$$\text{Gini}(0,2) = 1 - \left[\left(\tfrac{0}{6}\right)^2 + \left(\tfrac{2}{6}\right)^2\right]$$

$$= 0.888$$

Attribute $A_2 < 4$ & class Yes : 3/6

$A_2 < 4$ & class No : 1/6

$$Gini(3,1) = 1 - \left[\left(\frac{3}{6}\right)^2 + \left(\frac{1}{6}\right)^2\right]$$

$$= 0.722$$

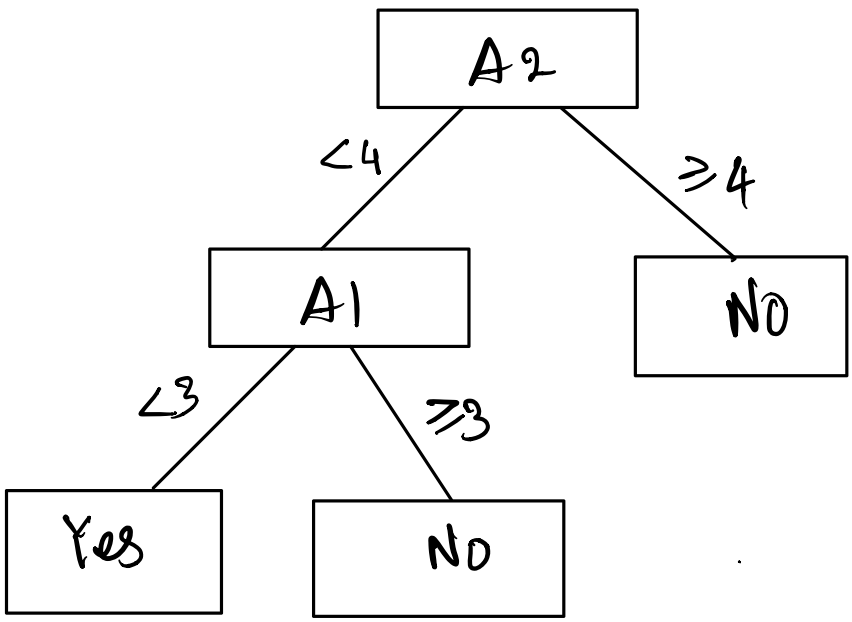
By adding weight and sum each of gini indices:

$$gini(Target, A_2) = \left(\frac{2}{16}\right) \times (0.888) + \left(\frac{4}{16}\right) \times (0.722)$$

$$= 0.111 + 0.1805$$

$$= 0.2915.$$

The least Gini Index is for A2 followed by A1.



Tree constructed using Gini Index for above data.

PART-2

PROBLEM-2

PART-2

We were asked to use this DB & implement repeated two fold cross validation holdout.

We were also given the folds for each repitition too & the no. of repititions given are 4.

We need to use a_1 as the root.

We also considered the class C as nominal.

O	a_1	a_2	C
O_1	1	1	1
O_2	0	0	0
O_3	0	1	0
O_4	0	0	0
O_5	1	1	1
O_6	1	1	0
O_7	0	0	0
O_8	1	0	1

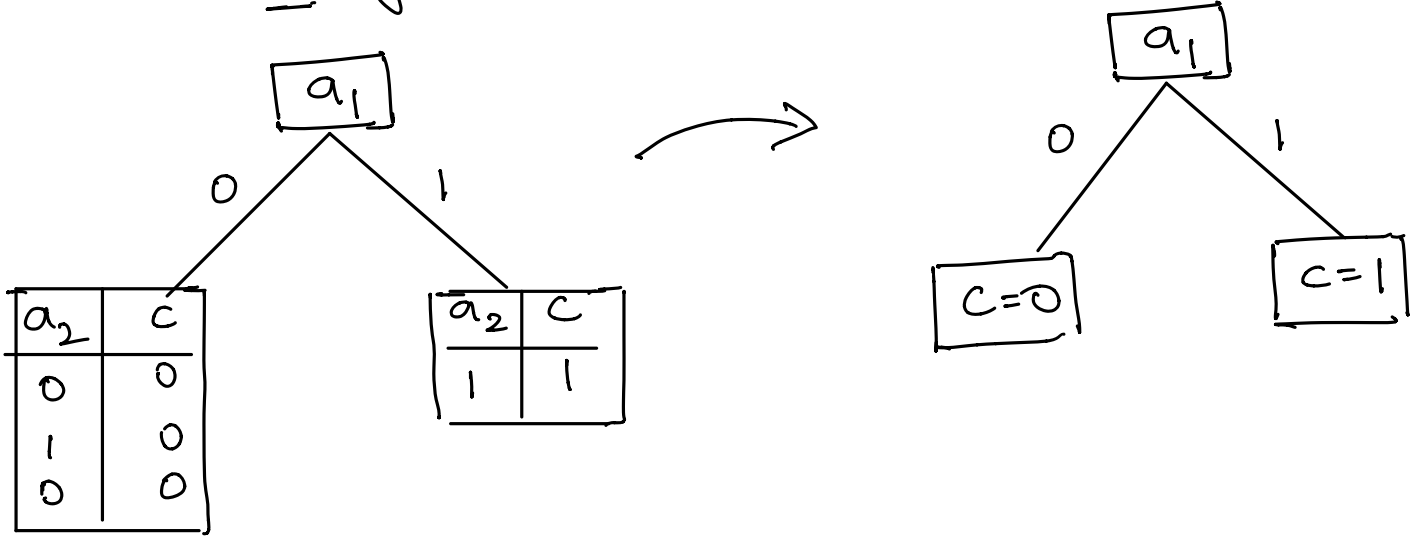
Step 1:

Repetition 1:

(1) train = $\{O_1, O_2, O_3, O_4\}$ test = $\{O_5, O_6, O_7, O_8\}$

So following the stages.

Training: (Stage 1)



As the decision tree is completed.

Discriminant rules :

Rule 1 : IF $a_1(x_1 = 0)$ THEN $c(x, = 0)$

Rule 2 : IF $a_1(x_1 = 1)$ THEN $c(x, = 1)$

We'll try to resubstitute as the part of Stage 2.

train : $\{O_1, O_2, O_3, O_4\}$

The rule accuracy can be calculated as : 100%

O_1 is classified well using Rule 2.

O_2, O_3, O_4 are also classified well using Rule 1.

So stage 3 :

test : $\{O_5, O_6, O_7, O_8\}$.

O_5, O_8 are classified well using Rule 2.

O_7 is classified well using Rule 1.

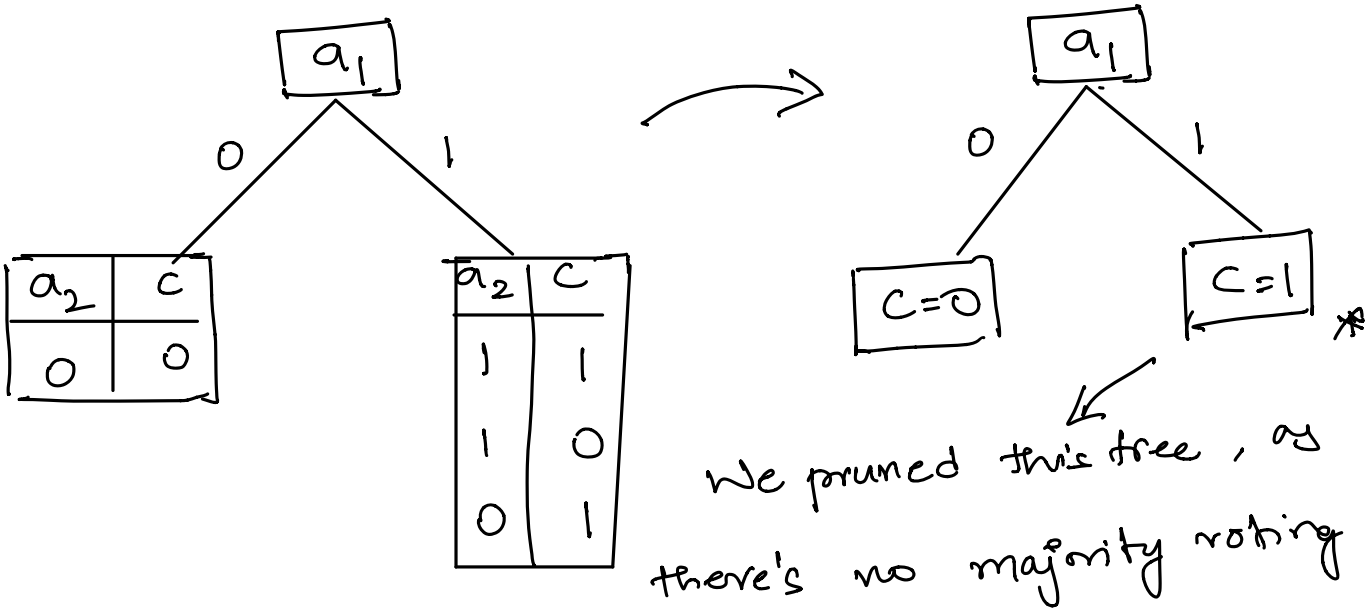
O_6 is misclassified from Rule 2.

So the predictive accuracy would be $3/4 = 75\%$.

(2) Now for the second fold as training set.

train = $\{O_5, O_6, O_7, O_8\}$ test = $\{O_1, O_2, O_3, O_4\}$

So following the stages.



We pruned this tree, as there's no majority voting present when $a_2 = 1$ $\{1, 0\}$ where as for $a_2 = 0$ $\{1\}$. So when $a_2 = 1$, we went forward by keeping c=1 or it is seen first.

As the decision tree is completed.

Discriminant rules:

Rule 1: IF $a_1(x_i = 0)$ THEN $c(x_i = 0)$

Rule 2: IF $a_1(x_i = 1)$ THEN $c(x_i = 1)$

We'll try to resubstitute as the part of stage 2.

train = $\{O_5, O_6, O_7, O_8\}$

The rule accuracy can be calculated as : 3/4 : 75%

O_5 is well classified using Rule 2

O_6 is misclassified from Rule 2.

O_7 is well classified using Rule 1.

O_8 is well classified using Rule 2.

So stage 3:

test : $\{O_1, O_2, O_3, O_4\}$.

O_1 is classified well using Rule 2.

O_2, O_3, O_4 are classified well using Rule 1.

so the predictive accuracy would be 100%.

So our classifier would be the union of these rules. F_1.

Rule 1 : IF $a_1(x_1=0)$ THEN $c(x_1=0)$

Rule 2 : IF $a_1(x_1=1)$ THEN $c(x_1=1)$

Rule 3 : IF $a_1(x_1=0)$ THEN $c(x_1=0)$

Rule 4 : IF $a_1(x_1=1)$ THEN $c(x_1=1)$

After removing the repetitions:

It would be

Rule 1 : IF $a_1(x_1=0)$ THEN $c(x_1=0)$

Rule 2: IF $a_1(x_1=1)$ THEN $c(x_1=1)$

The classifier's rule accuracy is $\frac{100+75}{2} = 87.5\%$.

& prediction accuracy is $\frac{100+75}{2} = 87.5\%$.

Repetition 2 :

If we clearly check the given fold is

test = $\{O_5, O_6, O_7, O_8\}$, test = $\{O_1, O_2, O_3, O_4\}$

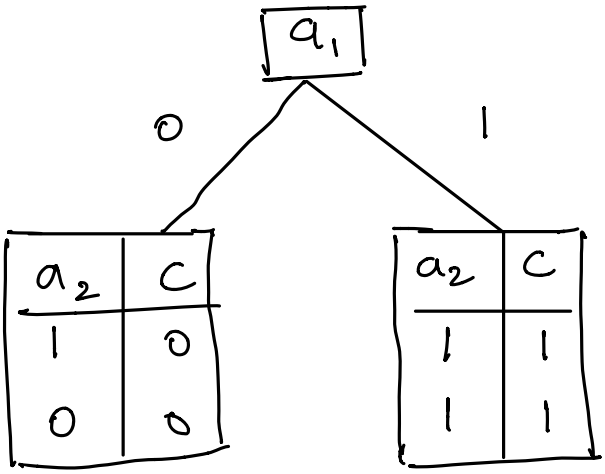
is exactly same as the 2nd fold in the previous repetition. So, similarly the 2nd fold in this's case would be the same as the 1st fold in previous scenario. so the classifier would be the same.

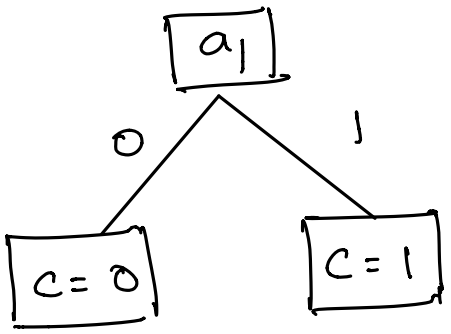
So $F_2 = F_1$

Repetition 3 :

(1) training = $\{O_1, O_3, O_5, O_7\}$ test = $\{O_2, O_4, O_6, O_8\}$.

Training: Stage 1 :





Discriminant rules:

Rule 1 : IF $a_1(x_i = 0)$ THEN $C(x_i = 0)$

Rule 2 : IF $a_1(x_i = 1)$ THEN $C(x_i = 1)$

So the rule accuracy is obtained by substituting the training data.

O_1 & O_5 are well classified using Rule 2.

O_3 & O_7 are well classified using Rule 1.

So the rule accuracy is 100%.

so during stage 3 for calculating predictive accuracy

O_2 & O_4 are well classified using Rule 1.

O_8 is well classified using Rule 2.

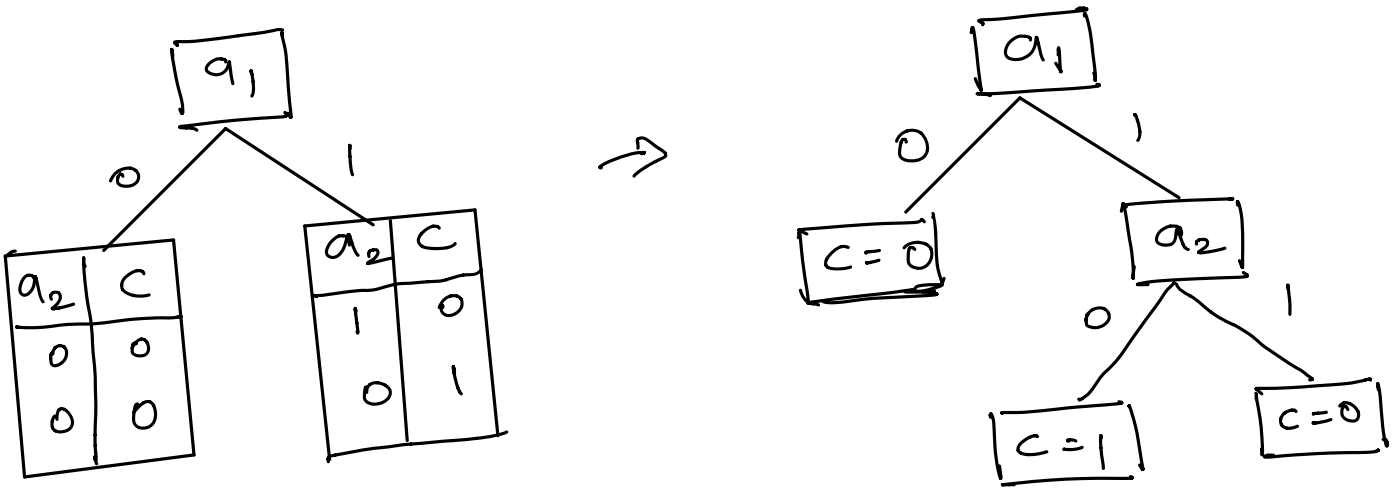
O_6 is misclassified from Rule 2.

so prediction accuracy = 75%.

(2)

training = $\{O_2, O_4, O_6, O_8\}$ test = $\{O_1, O_3, O_5, O_7\}$.

Training: Stage 1:



Discriminant Rules:

Rule 1: IF $a_1(x_1=0)$ THEN $c(x_1=0)$

Rule 2: IF $a_1(x_1=1)$ AND $a_2(x_2=0)$ THEN $c(x_1=1)$

Rule 3: IF $a_1(x_1=1)$ AND $a_2(x_1=1)$ THEN $c(x_1=0)$.

For stage 2 & resubstituting, the rule accuracy is

O_2 & O_4 are well classified using Rule 1.

O_6 is well classified using Rule 3.

O_8 is well classified using Rule 2.

So Rule Accuracy is 100%.

prediction Accuracy during stage 3 is

O_3 & O_7 are well classified using Rule 1.

O_1 & O_5 are misclassified from Rule 3.

So predictive accuracy is 50%.

So the combined rules of our classifier F_3 would be without repetitions.

Rule 1: IF $a_1(x_1=0)$ THEN $c(x_1=0)$

Rule 2: IF $a_1(x_1=1)$ AND $a_2(x_2=0)$ THEN $c(x_1=1)$

Rule 3: IF $a_1(x_1=1)$ AND $a_2(x_1=1)$ THEN $c(x_1=0)$.

Final Rule accuracy = $\frac{100+100}{2}$ = 100%.

Predictive accuracy would be = $\frac{50+75}{2}$

= 62.5%.

Repetition 4: The folds same as repetition 3 & hence the classifier would be the same too

$F_4 = F_3$

Step 2:

After calculating the predictive & rule accuracy in the previous step.

We know that $F_1 = F_2$ & $F_3 = F_4$.

So let's compare F_1 & F_3.

	Rules Accuracy	Predictive Accuracy
F_1	87.5%	87.5%
F_3	100%	62.5%

Based on these details, Rule Accuracy is almost similar for both these classifiers but the predictive accuracy of F_3 is very low compared to F_1. So I would pick F_1 or F_2 as my classifier F. Let it be F_1

So $F_1 = F$.

Step 3 :

We need to construct the bagged ensemble classifier F^*.

F^* contains F_1, F_2, F_3 & F_4. Since $F_1 = F_2$ & $F_3 = F_4$. The majority voting is calculated just between F_1 & F_3.

Given data.

O	a_1	a_2
o_1	0	1
o_2	0	0
o_3	1	0
o_4	1	1

Since our F is F_1

the rules of this classifier are :

F :

Rule 1 : IF $a_1(x_1 = 0)$ THEN $c(x_1 = 0)$

Rule 2 : IF $a_1(x_1 = 1)$ THEN $c(x_1 = 1)$

And our F^* consists of F_1, F_2 $(F_1 = F_2)$ & F_3, F_4 $(F_3 = F_4)$

f^* : F_1

Rule 1 : IF $a_1(x_1 = 0)$ THEN $c(x_1 = 0)$

Rule 2 : IF $a_1(x_1 = 1)$ THEN $c(x_1 = 1)$

F_3

Rule 1: IF $a_1(x_1=0)$ THEN $c(x_1=0)$

Rule 2: IF $a_1(x_1=1)$ AND $a_2(x_4=0)$ THEN $c(x_1=1)$

Rule 3: IF $a_1(x_1=1)$ AND $a_2(x_1=1)$ THEN $c(x_1=0)$.

Let's first classify using F.

O_1 is classified 0 using Rule 1.

O_2 is classified 0 using Rule 1.

O_3 is classified 1 using Rule 2.

O_4' is classified 1 using Rule 2.

for F^*

F_1 is classified similar to F as above.

where as F_3.

O_1 is classified 0 using Rule 1.

O_2 is classified 0 using Rule 1.

O_3 is classified 1 using Rule 2.

O_4 is classified 0 using Rule 3.

Since we need to take majority vote.

of F_1 & F_3 for F^{**}. The final result would be.

O_1 & O_2 are classified as 0.

O_3 & O_4 are classified as 1.

(O_4 doesn't have a majority vote, so we went with 1 as it's seen first).

	F	F^{**}
O_1	0	0
O_2	0	0
O_3	1	1
O_4	1	1

So based on our results, both these classifiers provided the same results.

PROBLEM 3 (10pts) NEURAL NETWORKS

Part 1 (7pts)

1. Give a short general description what is a Neural Network

2. Give a short general description how Neural Network learns

3. Given a classification data **D** with **attributes a1, a2, ... an** and classes **c1, c2, .. ck**

Which is the number of INPUT nodes of any NN for **D**?

Which is the number of OUTPUT nodes of any NN for full classification for **D**?

Which is he number of hidden layers?

Which is the number of nodes in the hidden layers?

4. Given the following

CLASSIFICATION DATA **D**

a1	a2	C
1	0	c1
0	1	c2
1	1	c1
1	0	c2
0	0	c3
0	0	c3

Design 3 NEURAL NETWORKS for **D**

One for full classification

Two for contrast learning (for your chosen classes)

Draw pictures and **explain** correctness of your topology

FOR SOLUTION USE your additional page

Part 2 (3pts)

1. Give a general description of the following STEPS of the **Backpropagation Algorithm**

Step 1: initialize

Step 2: feed

Step 3: propagate

Step 4: backpropagate

Step 5: backpropagate

Step 6: repeat

Step 7; terminate when

PART - 2

Question - 3

PART - 1

(1) Neural Network is basically a set of nodes which are connected and mimics the functionality of a human brain. There nodes start with the input nodes, internal nodes & finally with the output nodes. The nodes which are at present at the same depth from the input nodes form layers. So there are generally three layers Input, hidden & output. The inputs are the attributes of the data tuple which are fed simultaneously through the input nodes. These layers are connected with each other through nodes & these connections are called edges & each edge is given a weight that gets adjusted as the learning proceeds. There is also a non-linear / activation function which takes the weighted summation

output from a layer & non-linearizes it before sending it to the next layer. As a Neural Network tends to mimic a human brain, it's also called as a Perceptron and a Neural Network with multiple layers is called as the Multi-Layered Perceptron. It is a supervised learning.

(2) Neural Network learning is referred as "Connectionist Learning" as the learning parameters are the weights of the edges or connections between the layers. So basically a Neural Network tend to adjust it's weights in order to correctly classify the input through the output layer. This in general a time taking process & possible where we can compromise on the learning phase. They also require few parameters to move through the learning phase like learning rate, momentum e.t.c. these parameters together called as the Network Topology. But these algorithms have high tolerance to noisy & incomplete data.

(3) Given the number of attributes as $a_1, a_2, \ldots a_n$ & classes $C_1, C_2, \ldots C_k$ for D.

(a) Number of input nodes of any NN for D:

This would be same for all Neural Networks & it is the number of attributes i.e n.

(b) Number of output nodes of any NN for full classification:

As it was given for full classification, the number of output nodes would the number of classes i.e k.

(c) Number of Hidden Layers:

This totally depends on the implementation & can't be determined by the number of attributes & classes but atleast need to be 1.

(d) Number of nodes in the hidden layers:

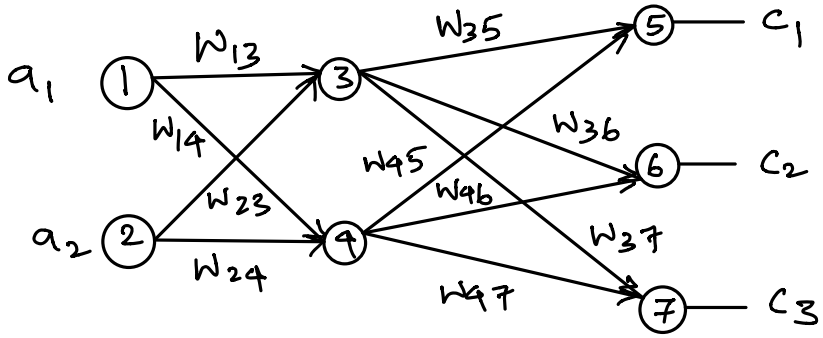
This also totally depends on the implementation & can't be determined by the number of attributes & classes but atleast need to be 1.

(4) Given classification data D.

a_1	a_2	C
1	0	c_1
0	1	c_2
1	1	c_1
1	0	c_2
0	0	c_3
0	0	c_3

We are asked to design 3 neural networks.

(1) Full classification.

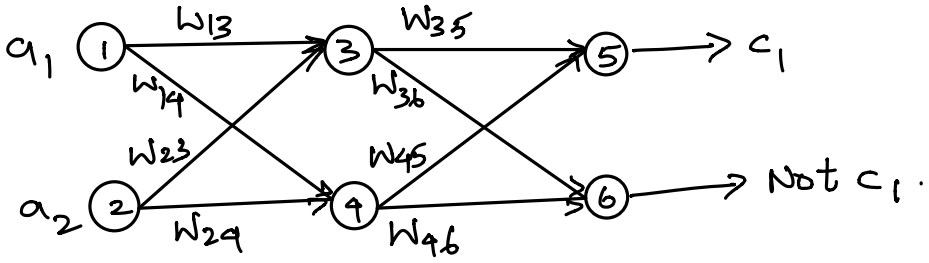


We have designed a Neural Network with 1 hidden layer containing 2 hidden nodes.

Since there are two attributes, there are 2 units (1,2) in the input layer. there are 3 classes to be classified, so there are 3 (5,6,7)

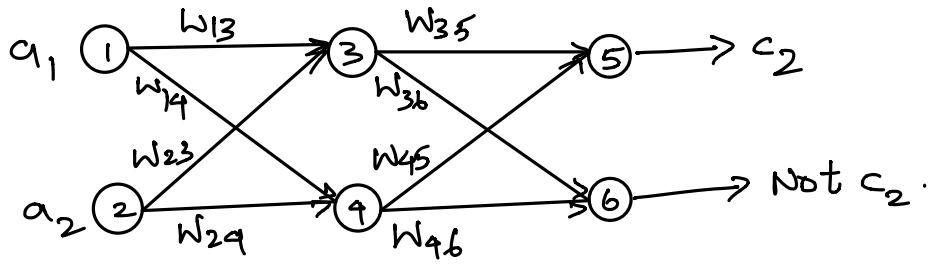
units in the output layer. And, we require atleast 1 hidden layer, we had one with 2 hidden nodes (3, 4). Hence the topology is valid.

(2) Contrast classification w.r.t c_1 :



We again used a hidden layer with 2 internal nodes (3,4) as there atleast need to be 1. As there are 2 attributes, there are 2 input nodes (1,2) & since there are only 2 classes to classify as this being contrast classification (c_1 & not c_1). We decided to go with 2 output nodes.

(3) Contrast classification w.r.t C_2:



We again used a hidden layer with 2 internal nodes (3,4) as there atleast need to be 1. As there are 2 attributes, there are 2 input nodes (1,2) & since there are only 2 classes to classify as this being contrast classifications (C_2 & not C_2). We decided to go with 2 output nodes.

PART-2

PROBLEM-3

PART-2

(1) General description of the following steps of the Backpropagation algorithm.

Step 1 : Initialize :

The weights of all the edges present in the Neural Network are initialized to some small random numbers & also each node is also initialized with a bias. These bias values are also initialized to some small random numbers.

Step 2 : Feed :

Here the training tuple is fed into the input layer where each input node gets the value of the corresponding attribute of the input training tuple.

Step 3 : Propogate :

After feeding the training tuple to the input layer, we need to propogate the inputs forward. The inputs to any unit/node are based on the outputs of the previous layer. It's basically an output of a non-linear activation function applied on the weighted sum of outputs along with the bias of the current node/unit. So basically each unit in the hidden & output layers takes in the net input & applies an activation function on it. Sigmoid or Logistic functions are the most commonly used activation functions.

Step 4 : Back propagate :

In this step, we back propagate the error. After propogating the inputs till the output layer we calculate the error w.r.t to the true output & update the weights & biases by propogating the error backwards.

Step 5 : Back propagate :

This is part of the above back propagation step where we are updating the weights & biases. This back propagation method learns using gradient descent to search for the optimum weights & biases. Our main agenda behind this updation is to decrease the error between the class prediction & assigned true class values. This gradient descent occurs based on a learning rate & this updation of weights & biases is called as case updation. There's another method where these changes are accumulated until all the training data is feeded into the Neural Network. This is called epoch updating, where one iteration through the training data is called an epoch.

Step 6 : Repeat :

We repeat the steps 1-5 again till we terminate to reach a global minimum. Many of the times this process might lead to a local minimum. so the learning rate, number of epochs are an important criteria for learning.

step 7 : Terminate :

We terminate this learning process in three conditions.

(1) When the updation value of the weights in the previous epoch were below some threshold

(2) No. of misclassified tuples are below some threshold.

(3) The prespecified number of epochs are completed.