جامعة الطائف
TAIF UNIVERSITY

DATA MINING

Assoc. Prof. Dr. Salha Alzahrani

College of Computers and Information Technology
Taif University
Saudi Arabia

s.zahrani@tu.edu.sa

Classification | Classification using Nearest Neighbour Algorithm

Recap of Previous Lecture

Content of This Lecture

Summary & Checklist

Recap of Lecture 4

- Introduction
- Nearest instance
- k-Nearest Neighbour Classification
- Example of classification using Nearest Neighbour Algorithm
- Distance Measures
- Distance Measures: Euclidean
- Distance Measures: Manhattan
- Distance Measures: Maximum Dimension
- Nearest Neighbour Algorithm: Step-by-Step

Recap of Previous Lecture
Content of This Lecture
Summary & Checklist

Content of Lecture 5

- Trees
- Decision Rules and Decision Trees
- Decision Trees: The Golf Example
- How to construct a Decision Tree?
- Functions of Decision Trees
- Decision Trees: The degrees Dataset
- The TDIDT Algorithm
- The TDIDT Algorithm: Adequacy Condition
- Summary & Checklist.

Classification | Trees

- Computer Scientists and Mathematicians often use a structure called a ***tree*** to represent data items and the processes applied to them.
- Tree is **a structure used to represent data items and the processes applied to them.**
- Example of a tree: →
  The letters A to M are labels added for ease of reference and are not part of the tree itself.

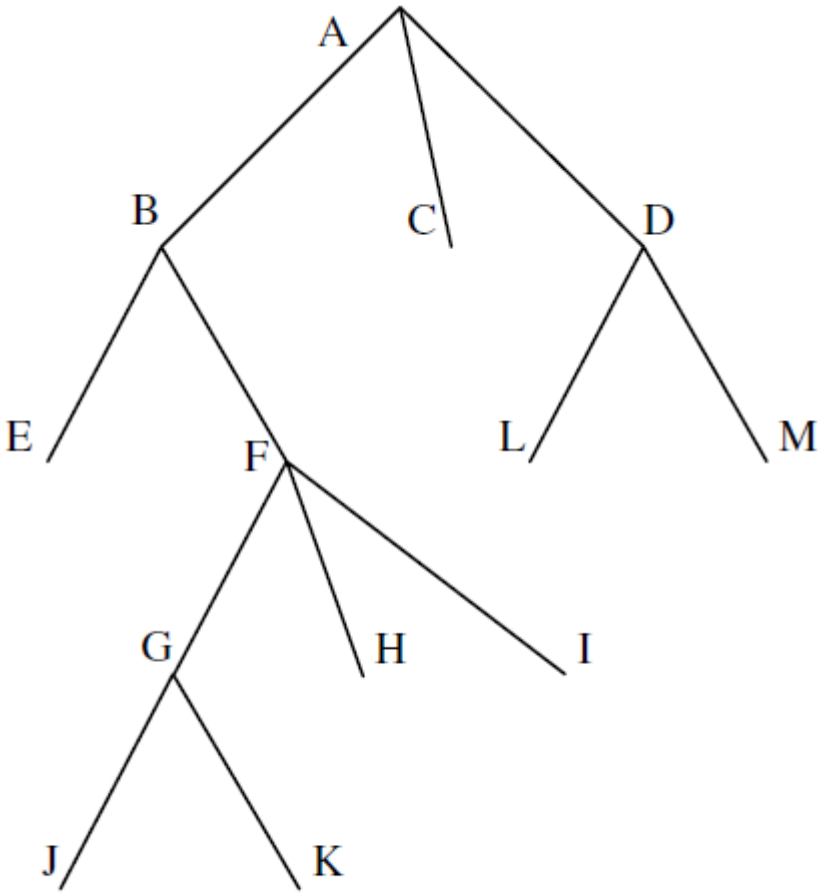


Figure A.1 A Tree with 13 Nodes

Classification | Trees

- A tree consists of a collection of points, called ***nodes***, joined by straight lines, called ***links***.

- The node at the top of the tree is called the ***root***, or the ***root node***.
- Nodes that have no other nodes below them in the tree are called ***leaf nodes*** or just ***leaves***.
- Nodes that are neither the root nor a leaf node are called ***internal nodes***.
- The path from the root node of a tree to any of its leaf nodes is called a ***branch***. A tree has as many branches as it has leaf nodes.

Classification | Trees

There are a number of conditions that must be satisfied to consider a structure as a tree:

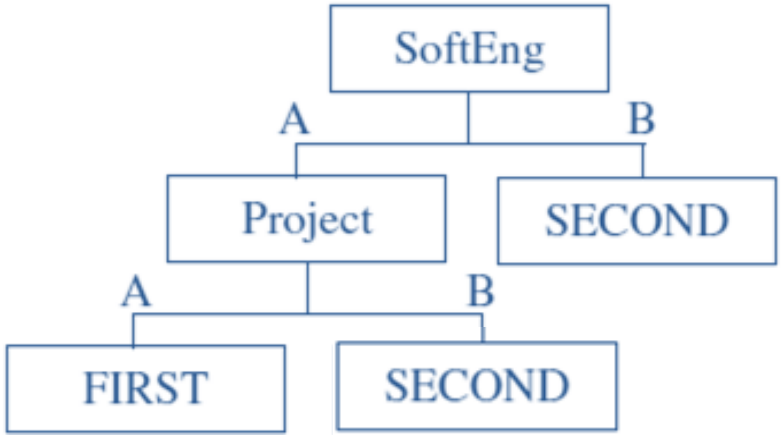
1. There must be a single node, the root, with no links 'flowing into' it from above.
2. There must be a path from the root node to every other node in the tree (so the structure is connected).
3. There must be only *one path* from the root to each of the other nodes.

Classification | Decision Rules and Decision Trees

Decision rules: also called **classification rules**, are a set of rules that can be used to predict the classification of an unseen (unclassified) instance.

IF SoftEng = A AND Project = A THEN Class = FIRST
Else Class = SECOND

Decision trees: also called classification trees, are a way of representing a set of classification rules using trees.



Classification | Decision Trees: The Golf Example

- A golfer who decides whether to **play** or **not play** each day on the basis of the weather.
- The figure shows the results of two weeks (14 days) of observations of weather conditions and the decision on whether or not to play.

- **If tomorrow the values of *Outlook*, *Temperature*, *Humidity* and *Windy* were sunny, 74 °F, 77% and false respectively, what would the decision be?**

Outlook	Temp (°F)	Humidity (%)	Windy	Class
sunny	75	70	true	play
sunny	80	90	true	don't play
sunny	85	85	false	don't play
sunny	72	95	false	don't play
sunny	69	70	false	play
overcast	72	90	true	play
overcast	83	78	false	play
overcast	64	65	true	play
overcast	81	75	false	play
rain	71	80	true	don't play
rain	65	70	true	don't play
rain	75	80	false	play
rain	68	80	false	play
rain	70	96	false	play

Classes
play, don't play
Outlook
sunny, overcast, rain
Temperature
numerical value
Humidity
numerical value
Windy
true, false

Figure 3.1 Data for the Golf Example

Classification | Decision Trees: The Golf Example

In order to determine the decision (classification) for a given set of weather conditions from the decision tree, first look at the value of *Outlook*.

There are three possibilities.

1. If the value of ***Outlook* is sunny**, next consider the value of ***Humidity***. If the value is less than or equal to 75 the decision is *play*. Otherwise the decision is *don't play*.
2. If the value of ***Outlook* is overcast**, the decision is *play*.
3. If the value of ***Outlook* is rain**, next consider the value of ***Windy***. If the value is true the decision is *don't play*, otherwise the decision is *play*.

*Note that the value of *Temperature* is never used.

Classification | Decision Trees: The Golf Example

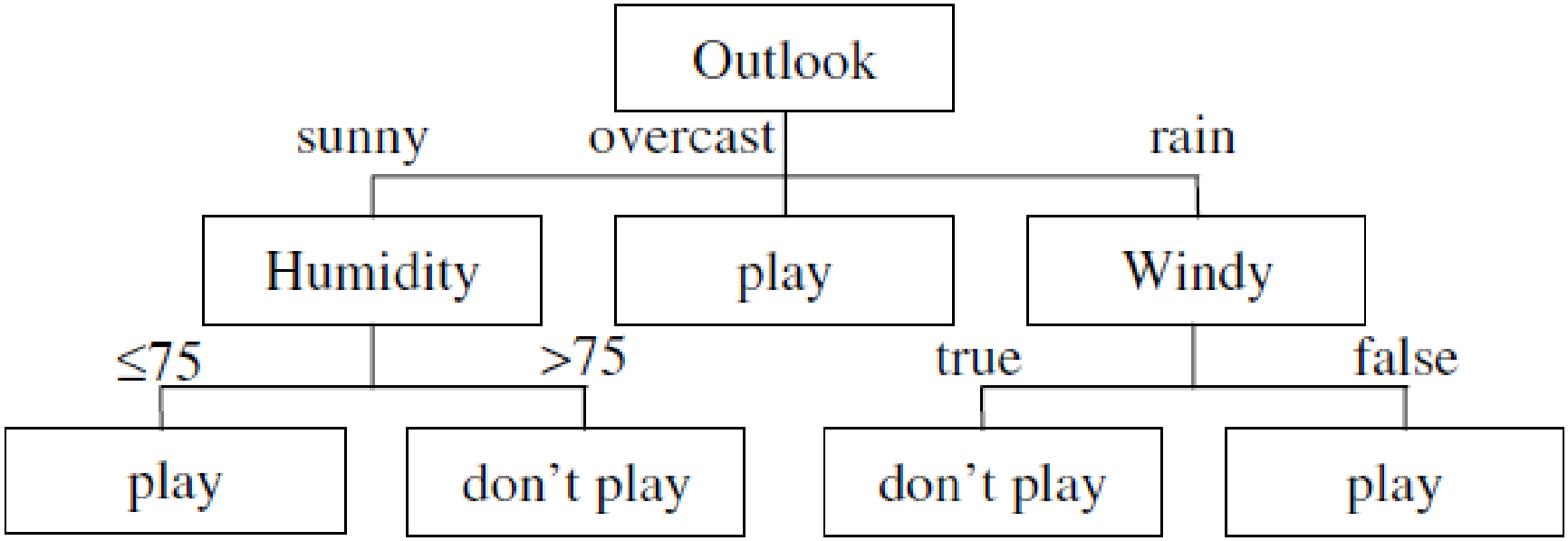


Figure 3.2 Decision Tree for the Golf Example

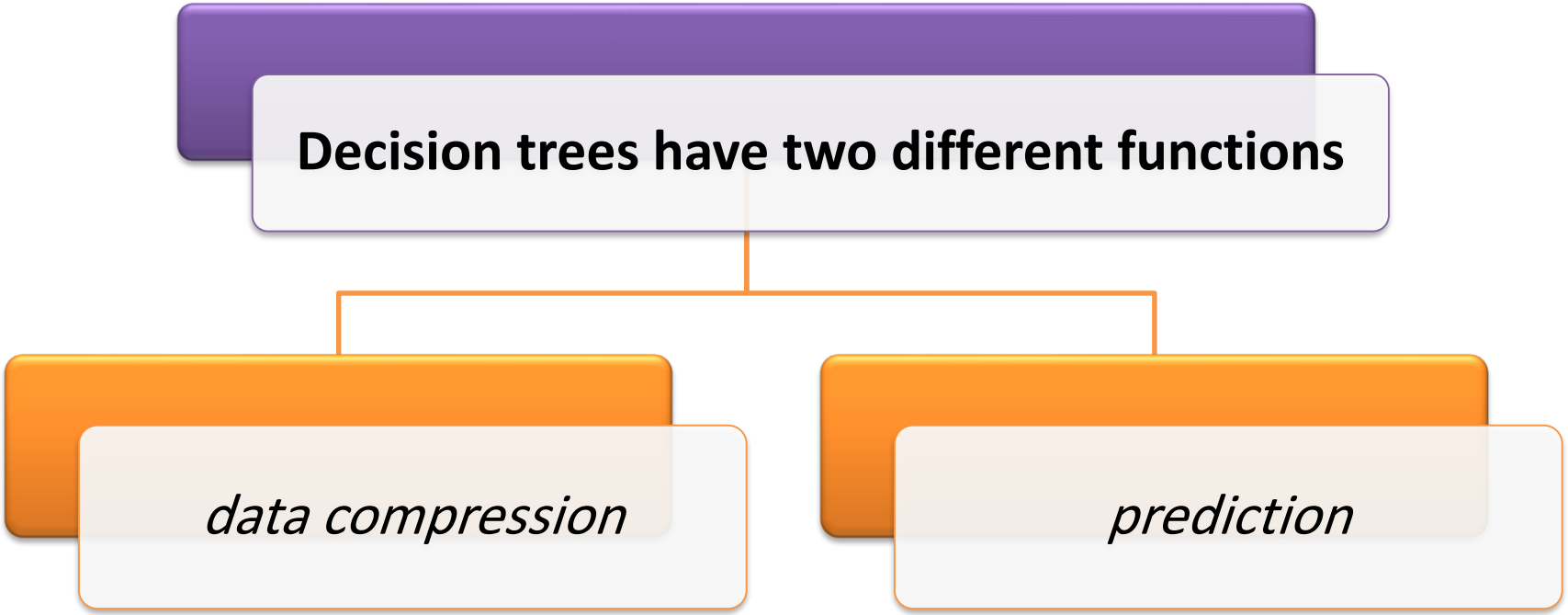
Classification | How to construct a Decision Tree?

- A decision tree is created by a process known as ***splitting on the value of attributes*** (or just ***splitting on attributes***), i.e. testing the value of an attribute such as *Outlook* and then creating a branch for each of its possible values.

- In the case of **continuous attributes** the test is normally whether the value is 'less than or equal to' or 'greater than' a given value known as the ***split value***.

- The splitting process continues until ***each branch can be labelled with just one classification***.

Classification | Functions of Decision Trees



- **A more compact way of representing the dataset.**

- **It can be used to predict the values of other instances not in the training set**
- **Using the decision tree, what is the class for the following instance**

Sunny	**74**	**77**	**False**	**???**

Classification

Decision Trees: The degrees Dataset

- **What determines which instance is classified as FIRST or SECOND?**

SoftEng	ARIN	HCI	CSA	Project	Class
A	B	A	B	B	SECOND
A	B	B	B	A	FIRST
A	A	A	B	B	SECOND
B	A	A	B	B	SECOND
A	A	B	B	A	FIRST
B	A	A	B	B	SECOND
A	B	B	B	B	SECOND
A	B	B	B	B	SECOND
A	A	A	A	A	FIRST
B	A	A	B	B	SECOND
B	A	A	B	B	SECOND
A	B	B	A	B	SECOND
B	B	B	B	A	SECOND
A	A	B	A	B	FIRST
B	B	B	B	A	SECOND
A	A	B	B	B	SECOND
B	B	B	B	B	SECOND
A	A	B	A	A	FIRST
B	B	B	A	A	SECOND
B	B	A	A	B	SECOND
B	B	B	B	A	SECOND
B	A	B	A	B	SECOND
A	B	B	B	A	FIRST
A	B	A	B	B	SECOND
B	A	B	B	B	SECOND
A	B	B	B	B	SECOND

Classes
FIRST, SECOND
SoftEng
A,B
ARIN
A,B
HCI
A,B
CSA
A,B
Project
A,B

Figure 3.3 The *degrees* Dataset

Classification | Decision Trees: The degrees Dataset

- The figure shows a possible **decision tree** corresponding to this training set.
- It consists of a number of ***branches***, each ending with a ***leaf node*** labelled with one of the valid classifications, i.e. FIRST or SECOND.

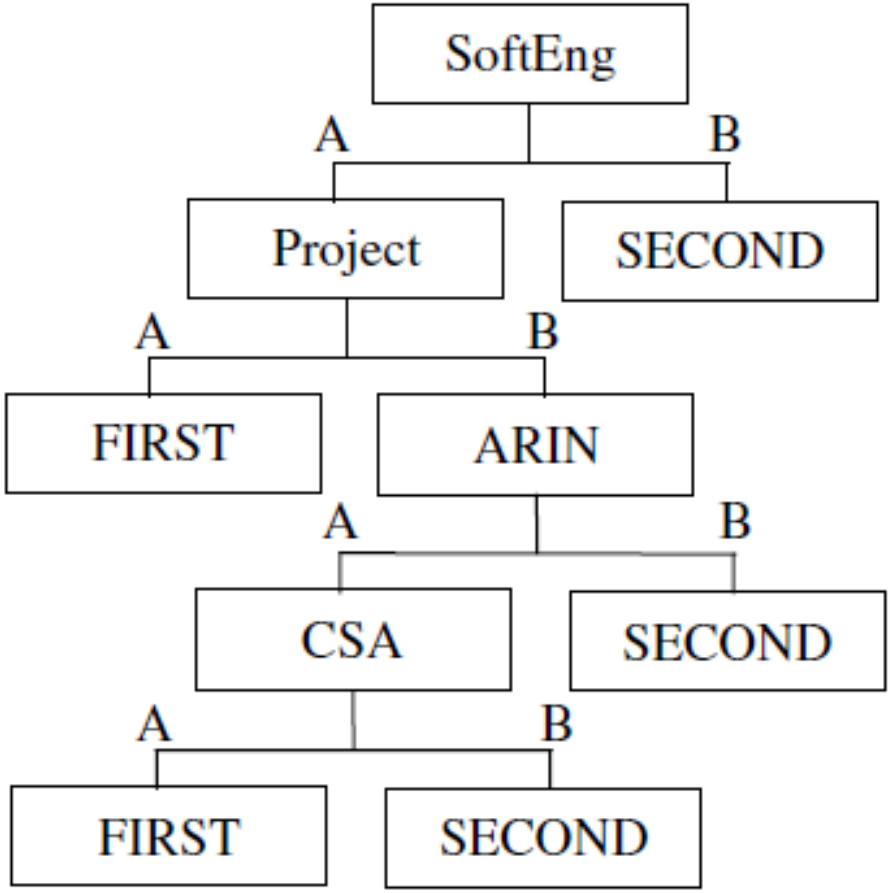


Figure 3.4 Decision Tree for the *degrees* Dataset

Classification | Decision Trees: The degrees Dataset

- For practical use, the rules can easily be simplified to an equivalent nested set of **IF . . . THEN . . . ELSE rules**, with even more compression

```
if (SoftEng = A) {
   if (Project = A) Class = FIRST
   else {
      if (ARIN = A) {
         if (CSA = A) Class = FIRST
         else Class = SECOND
      }
      else Class = SECOND
   }
}
else Class = SECOND
```

Classification | The TDIDT Algorithm

- Decision trees are widely used as a means of generating classification rules because of the existence of a simple but very powerful algorithm called **TDIDT**, which stands for ***Top-Down Induction of Decision Trees***.

- This has been known since the mid-1960s and has formed the basis for many classification systems, two of the best-known being ID3 [4] and C4.5 [3], as well as being used in many commercial data mining packages.

Classification | The TDIDT Algorithm

TDIDT: BASIC ALGORITHM

IF all the instances in the training set belong to the same class
THEN return the value of the class
ELSE (a) Select an attribute A to split on$^+$
(b) Sort the instances in the training set into subsets, one for each value of attribute A
(c) Return a tree with one branch for each *non-empty* subset, each branch having a descendant subtree or a class value produced by applying the algorithm recursively

$^+$ Never select an attribute twice in the same branch

Figure 3.5 The TDIDT Algorithm

Classification | The TDIDT Algorithm: Adequacy Condition

- There is one important condition which must hold before the TDIDT algorithm can be applied. This is the ***Adequacy Condition***: no two instances with the same values of all the attributes may belong to different classes.

- A major problem with the TDIDT algorithm, which is not apparent at first sight, is that it is ***underspecified***. The algorithm specifies 'Select an attribute *A* to split on' but no method is given for doing this.

- Provided the adequacy condition is satisfied the algorithm is guaranteed to terminate and any selection of attributes will produce a decision tree, provided that an attribute is never selected twice in the same branch.

Recap of Previous Lecture
Content of This Lecture
Summary & Checklist

Summary & Checklist

- ☑ Trees
- ☑ Decision Rules and Decision Trees
- ☑ Decision Trees: The Golf Example
- ☑ How to construct a Decision Tree?
- ☑ Functions of Decision Trees
- ☑ Decision Trees: The degrees Dataset
- ☑ The TDIDT Algorithm
- ☑ The TDIDT Algorithm: Adequacy Condition.

Reminder | Next Lecture !

Mid-Term Exam

Covers everything: Lecture 1 - Lecture 5

Thank You !

✉ s.zahrani@tu.edu.sa