Introduction to Artificial Intelligence

COMP307/AIML420

3K & Decision Tree: Tutorial

Dr Fangfang Zhang

fangfang.zhang@ecs.vuw.ac.nz

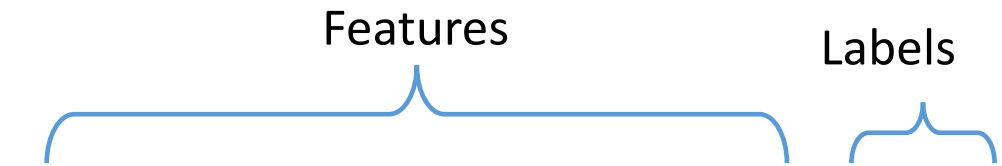
COMP307/AIML420 Week 3 (Tutorial)

1. Announcements
   - Assignment 1 (**15%**)
   - Helpdesk sessions
2. Sets
   - Training and Test sets
   - Validation set
3. Datasets
   - Instances
   - Features and feature vectors
   - Class label
4. 3-K
   - k-Nearest Neighbour
   - k-fold Cross Validation
   - k-Means Clustering
5. Decision Trees
   - DT learning
   - Impurity measure Conditions

Tips

- The goal of this course is to learn how to design/implement algorithms (keys) rather than using tools (except for reading data)
- Doing assignments (high level):
  - understand
  - implement

Datasets and Instances



f_1	f_2	f_3	f_4	f_5	. . .	class
10	2.2	45	3.7	22.1	. . .	A
3.7	7.9	12	2.1	17.5	. . .	A
22.8	27.9	11.4	36	77	. . .	B
90.4	6.34	2.77	15.8	53.7	. . .	A
74.6	4.78	84.9	15.9	103	. . .	B
2.89	14.7	3.11	10	52	. . .	B

Each row is an instance

K-Nearest Neighbour

- **Predict** class label of an instance (in test data) based on the closest k neighbours (from training data)
- Doing assignments (part 1):
  - store training instance --- this is **a KNN model!** (lazy learning)
  - normalise the data. i.e., both training and test (data preprocessing)
  - find the k nearest instances (neighbours) for each test instance
  - set the majority class label of found neighbours (voting) as the predicted class label of an instance
  - predicted label VS true label to calculate classification accuracy

K-Nearest Neighbour

Distance measure (Euclidean distance)

$$d = \sqrt{\sum_{i=1}^{n} \frac{(a_i - b_i)^2}{R_i^2}} = \sqrt{\frac{(a_1 - b_1)^2}{R_1^2} + \frac{(a_2 - b_2)^2}{R_2^2} + \cdots + \frac{(a_n - b_n)^2}{R_n^2}}$$

Range of the *i*th feature

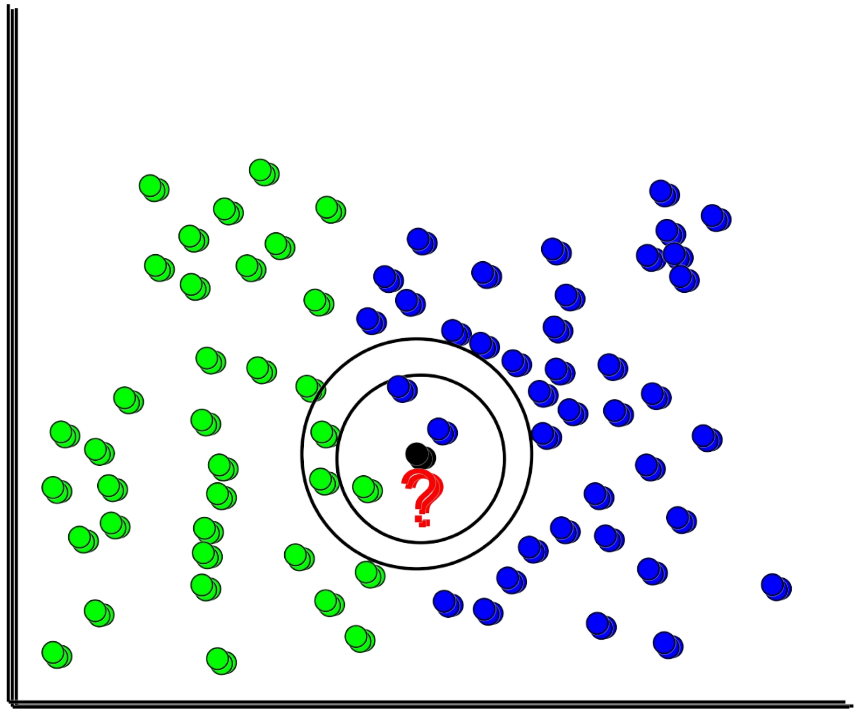
$$d = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

Normalise to [0, 1], based on each column

Training Set

f_1	f_2	f_3	class		
10.3	45.7	2.7	A	$d(\cdot,\cdot) =$	14.84
7.1	80.5	1.1	A	$d(\cdot,\cdot) =$	47.40
22.3	20.4	9.6	B	$d(\cdot,\cdot) =$	24.57
30.5	21.2	17.9	B	$d(\cdot,\cdot) =$	33.65
5.2	67.1	7.7	A	$d(\cdot,\cdot) =$	33.88
15.6	18.6	11.4	B	$d(\cdot,\cdot) =$	21.19
11.9	53.4	6.3	A	$d(\cdot,\cdot) =$	22.24

2.1	33.5	4.7	A



K-means Clustering: Algorithm

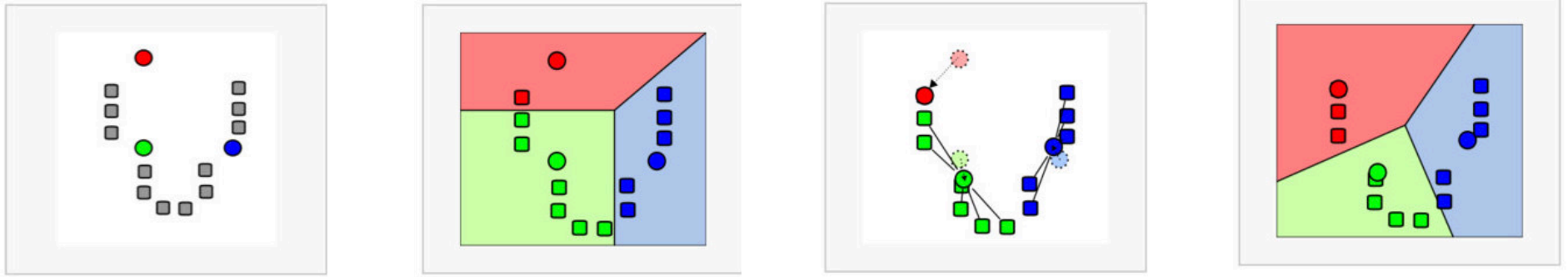
1. Initialise k initial “means” randomly from the data set

2. Create k clusters by assigning every instance to the nearest cluster: based on the nearest mean according to the distance measure

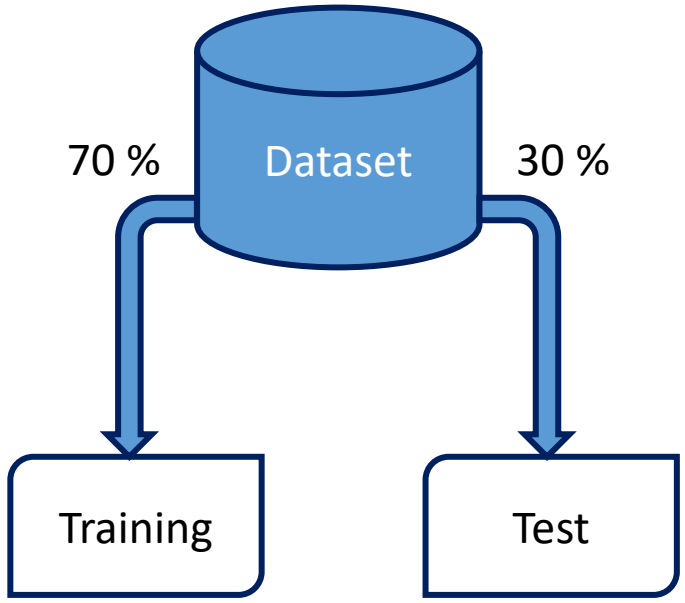
3. Replace the old means with the centroid (mean) of each cluster

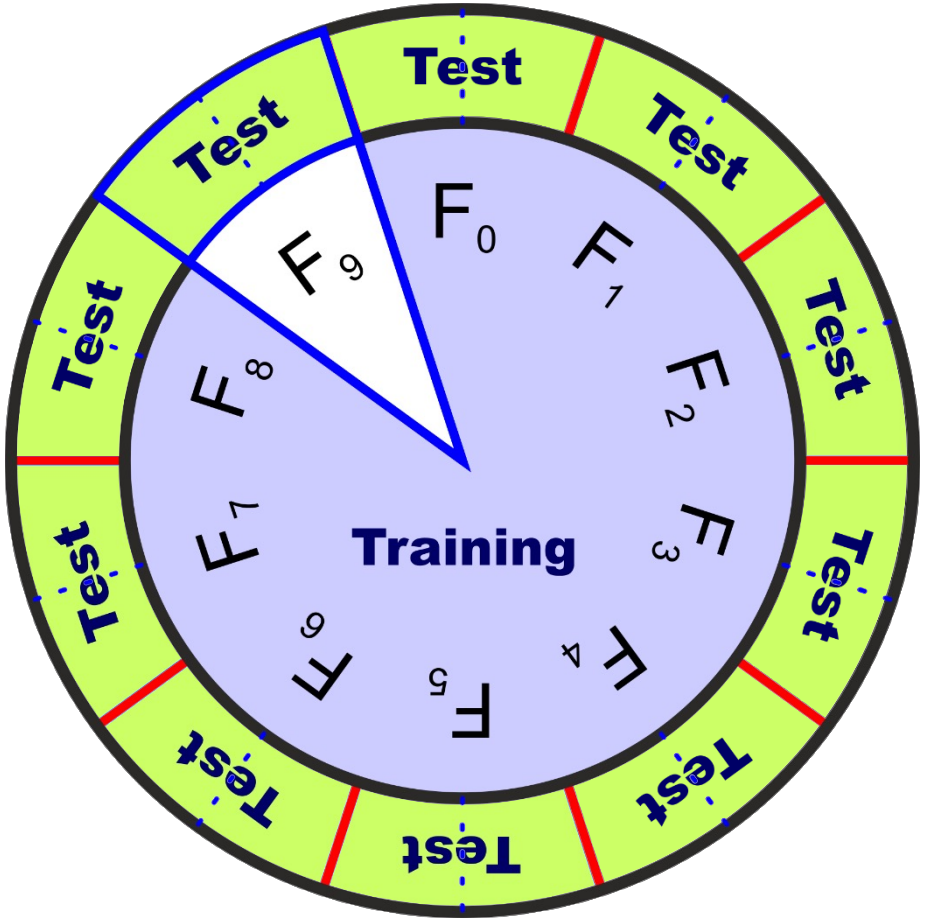
4. Repeat the above two steps until convergence (no change in each cluster centroid).

- *Centroid is not an instance*



k-fold Cross Validation





75 %
92 %
43 %
76 %
30 %
82 %
83 %
47 %
50 %
73 %

65.10 %

- Less number of training instances
- Avoid training instance selection biases

Training Dataset

- Approve/Reject a loan application?

Applicant	Job	Deposit	Family	Class
1	true	low	single	Approve
2	true	low	couple	Approve
3	true	low	single	Approve
4	true	high	single	Approve
5	false	high	couple	Approve
6	false	low	couple	Reject
7	true	low	children	Reject
8	false	low	single	Reject
9	false	high	children	Reject

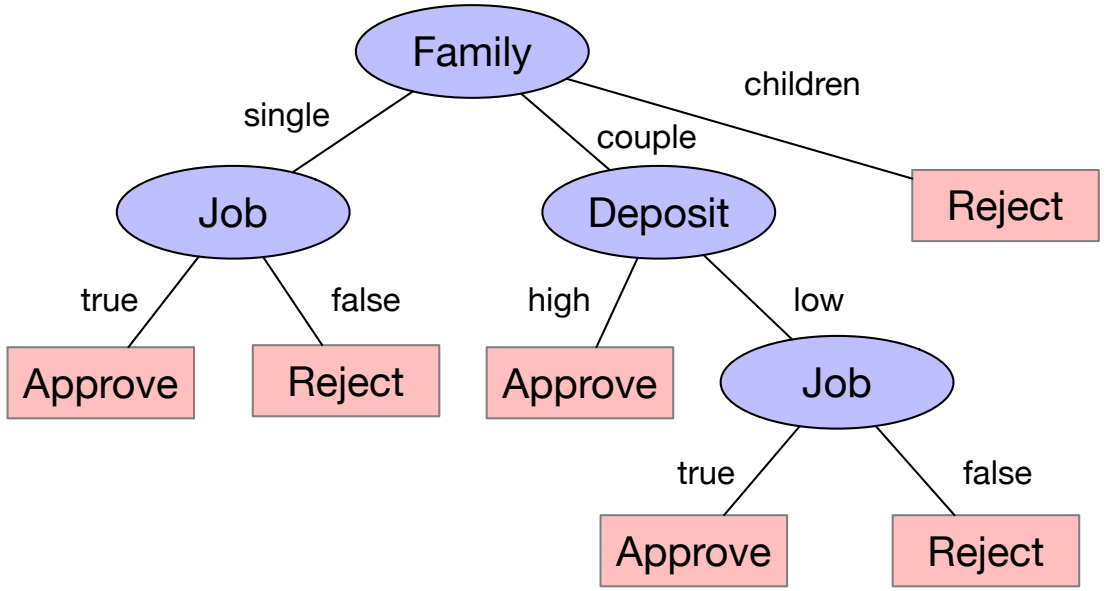
Binary classification task

Class Label: Approve and Reject

Three features

- **Job**: true and false
- **Deposit**: low and high
- **Family**: single
  couple
  children

An Example Decision Tree



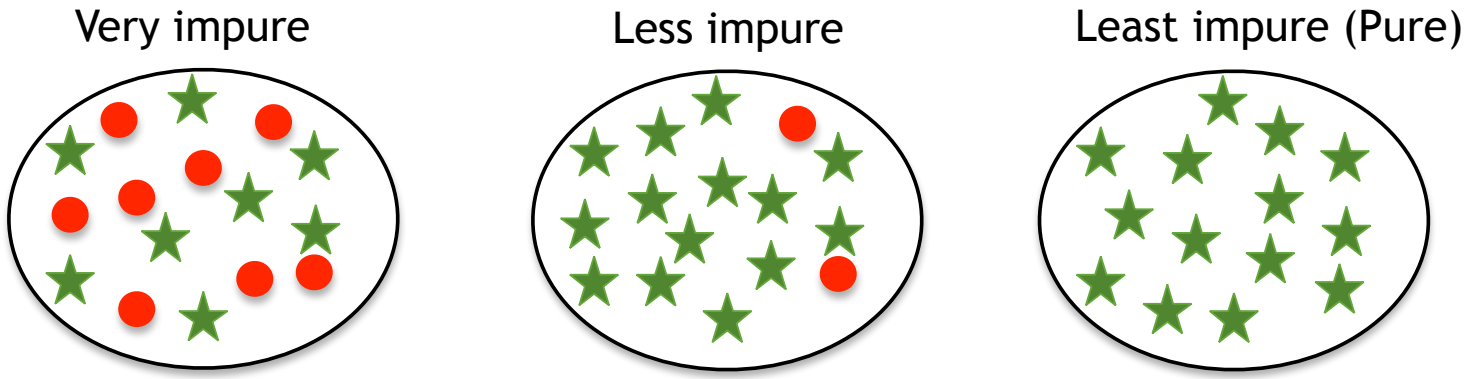
Applicant	Job	Deposit	Family	Class
1	true	low	single	Approve
2	true	low	couple	Approve
3	true	low	single	Approve
4	true	high	single	Approve
5	false	high	couple	Approve
6	false	low	couple	Reject
7	true	low	children	Reject
8	false	low	single	Reject
9	false	high	children	Reject

Node: features (blue), or classes (pink)

Edge: values of the parent nodes

1. How to choose feature

Greedy design: should make the nodes as pure as possible



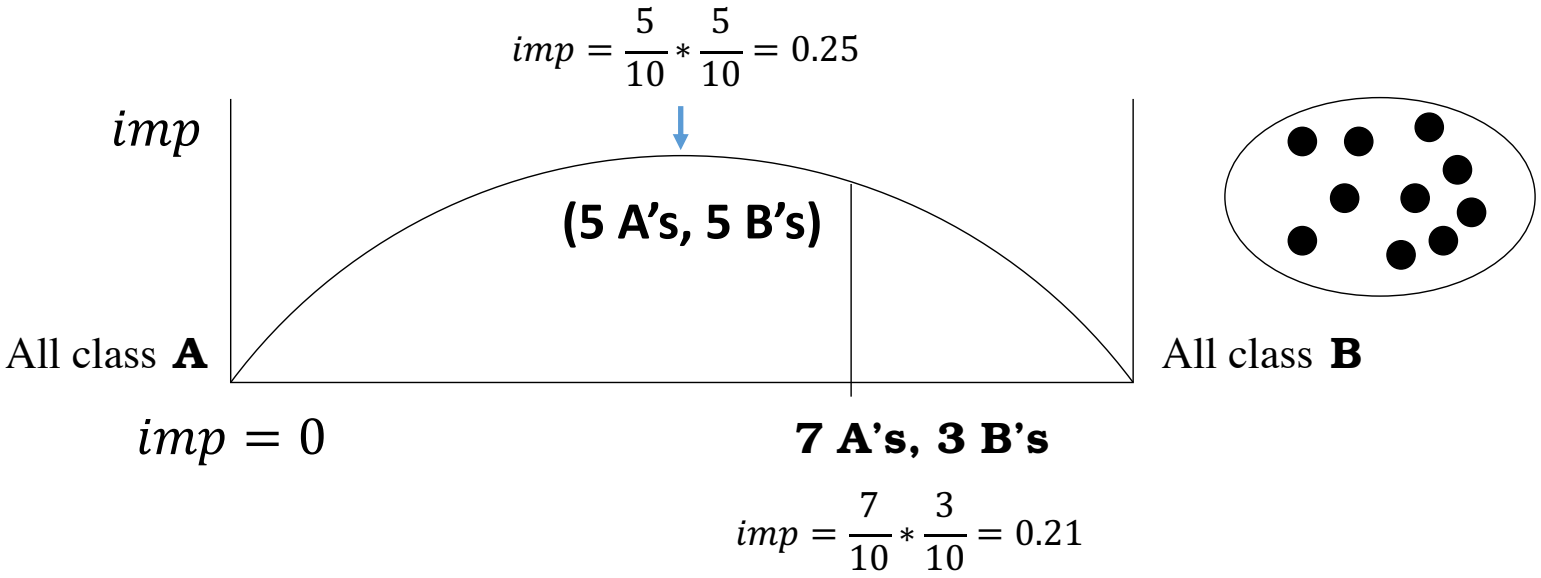
- Node (im)purity: can be defined in different ways
  - **Probability based**

Node Impurity Measure

- Assume there are two classes A and B
- At a node: *m* instances — class A, *n* instances — class B
- Impurity: $imp(node) = P(A)P(B) = \frac{m}{m+n} \times \frac{n}{m+n} = \frac{mn}{(m+n)^2}$

The smaller the better

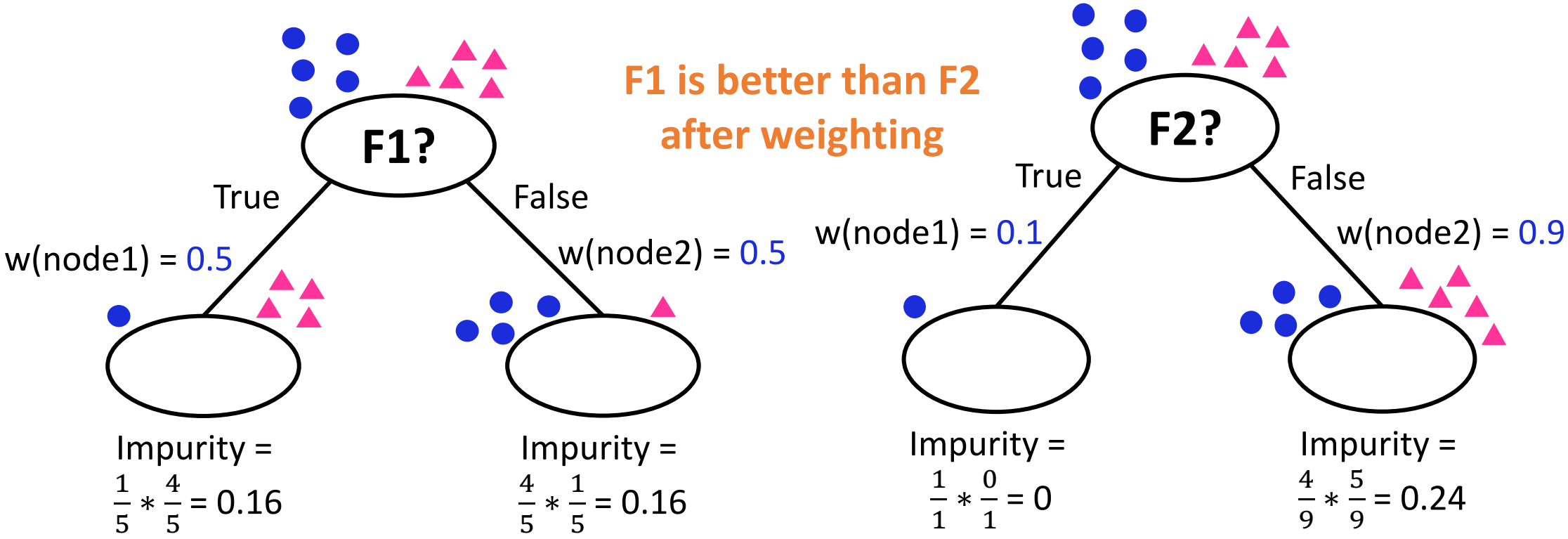
  - If pure (m = 0 or n = 0): $imp = 0$
  - If m = n, imp is maximum
  - Smooth



Node Impurity Measure

$$imp(node) = P(A)P(B) = \frac{m}{m+n} \times \frac{n}{m+n} = \frac{mn}{(m+n)^2}$$

- **Weighted impurity =** $\sum_{i=1}^{2} w(node_i) \times impurity(node_i)$



Weighted Impurity
= 0.5 ∗ 0.16 + 0.5 ∗ 0. 16 = 0.16

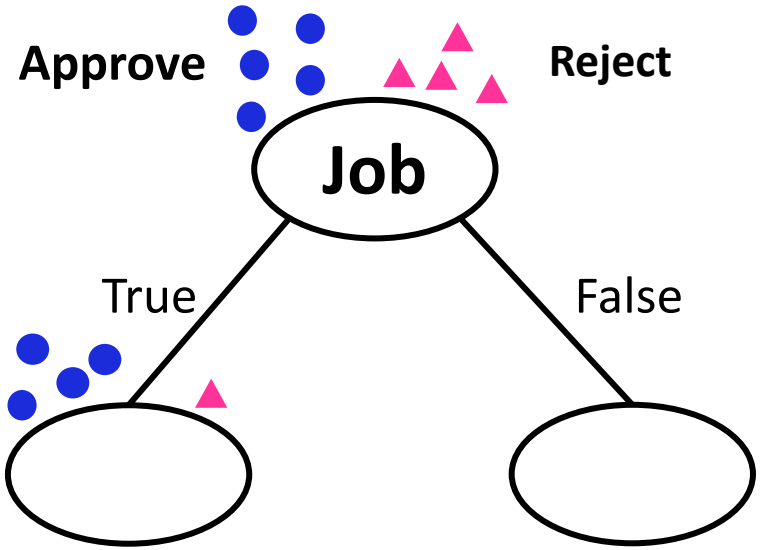
Weighted Impurity
= 0.1 ∗ 0 + 0.9 ∗ 0.24 = 0.22

Learning a Decision Tree

Applicant	Job	Deposit	Family	Class
1	true	low	single	Approve
2	true	low	couple	Approve
3	true	low	single	Approve
4	true	high	single	Approve
5	false	high	couple	Approve
6	false	low	couple	Reject
7	true	low	children	Reject
8	false	low	single	Reject
9	false	high	children	Reject

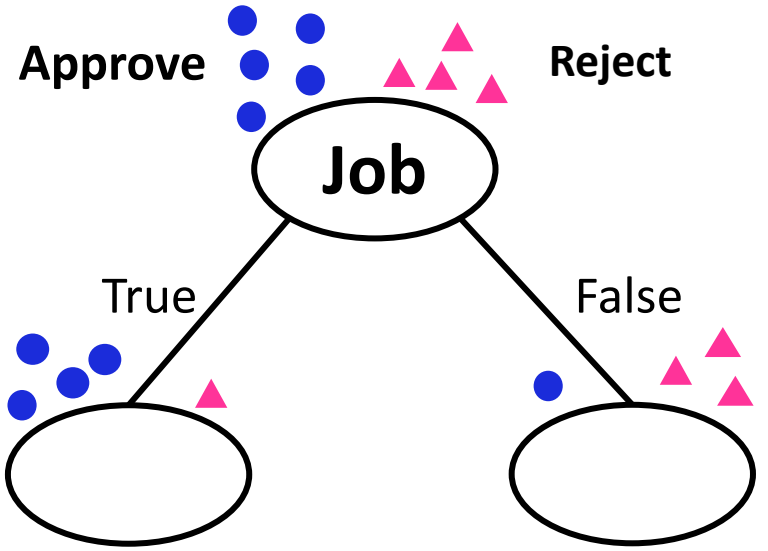
Learning a Decision Tree

Applicant	Job	Deposit	Family	Class
1	true	low	single	Approve
2	true	low	couple	Approve
3	true	low	single	Approve
4	true	high	single	Approve
5	false	high	couple	Approve
6	false	low	couple	Reject
7	true	low	children	Reject
8	false	low	single	Reject
9	false	high	children	Reject



Learning a Decision Tree

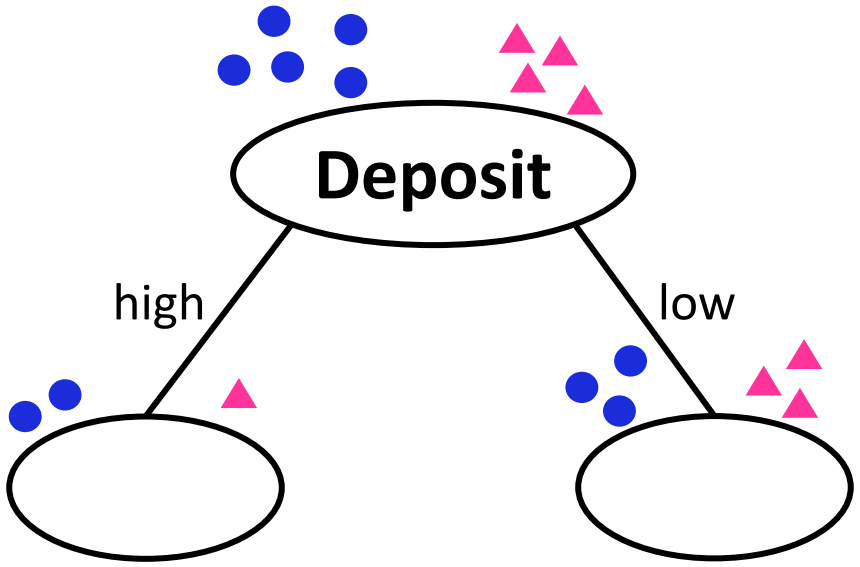
Applicant	Job	Deposit	Family	Class
1	true	low	single	Approve
2	true	low	couple	Approve
3	true	low	single	Approve
4	true	high	single	Approve
5	false	high	couple	Approve
6	false	low	couple	Reject
7	true	low	children	Reject
8	false	low	single	Reject
9	false	high	children	Reject



$imp(\text{node1}) = \frac{4}{5} * \frac{1}{5}$ $imp(\text{node2}) = \frac{1}{4} * \frac{3}{4}$

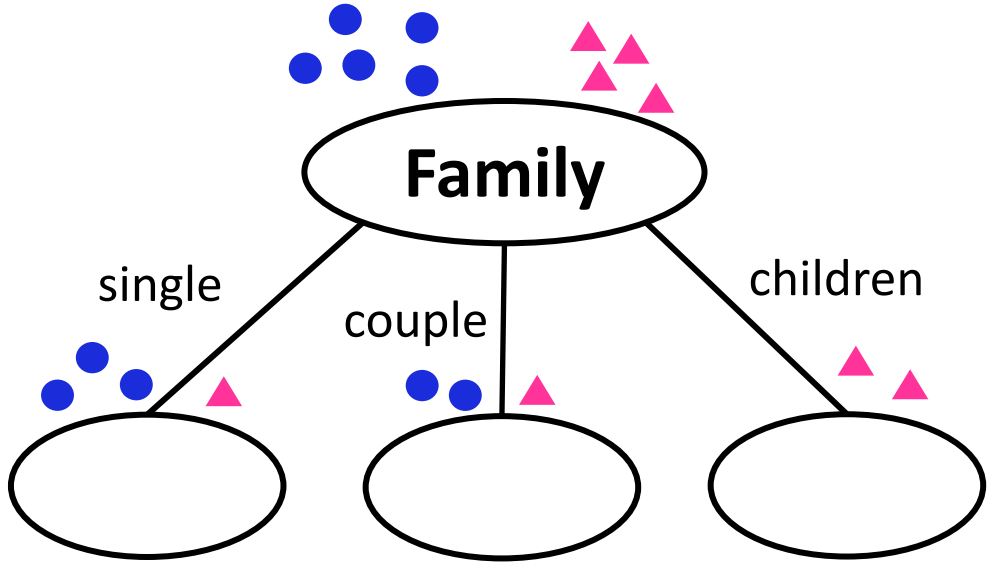
$$imp(\text{Job}) = \frac{5}{9} * \frac{4}{5} * \frac{1}{5} + \frac{4}{9} * \frac{1}{4} * \frac{3}{4} = 0.17$$

Learning a Decision Tree



$imp(\text{node1}) = \frac{2}{3} * \frac{1}{3}$

$imp(\text{node2}) = \frac{3}{6} * \frac{3}{6}$



$imp(\text{node1}) = \frac{3}{4} * \frac{1}{4}$

$imp(\text{node2}) = \frac{2}{3} * \frac{1}{3}$

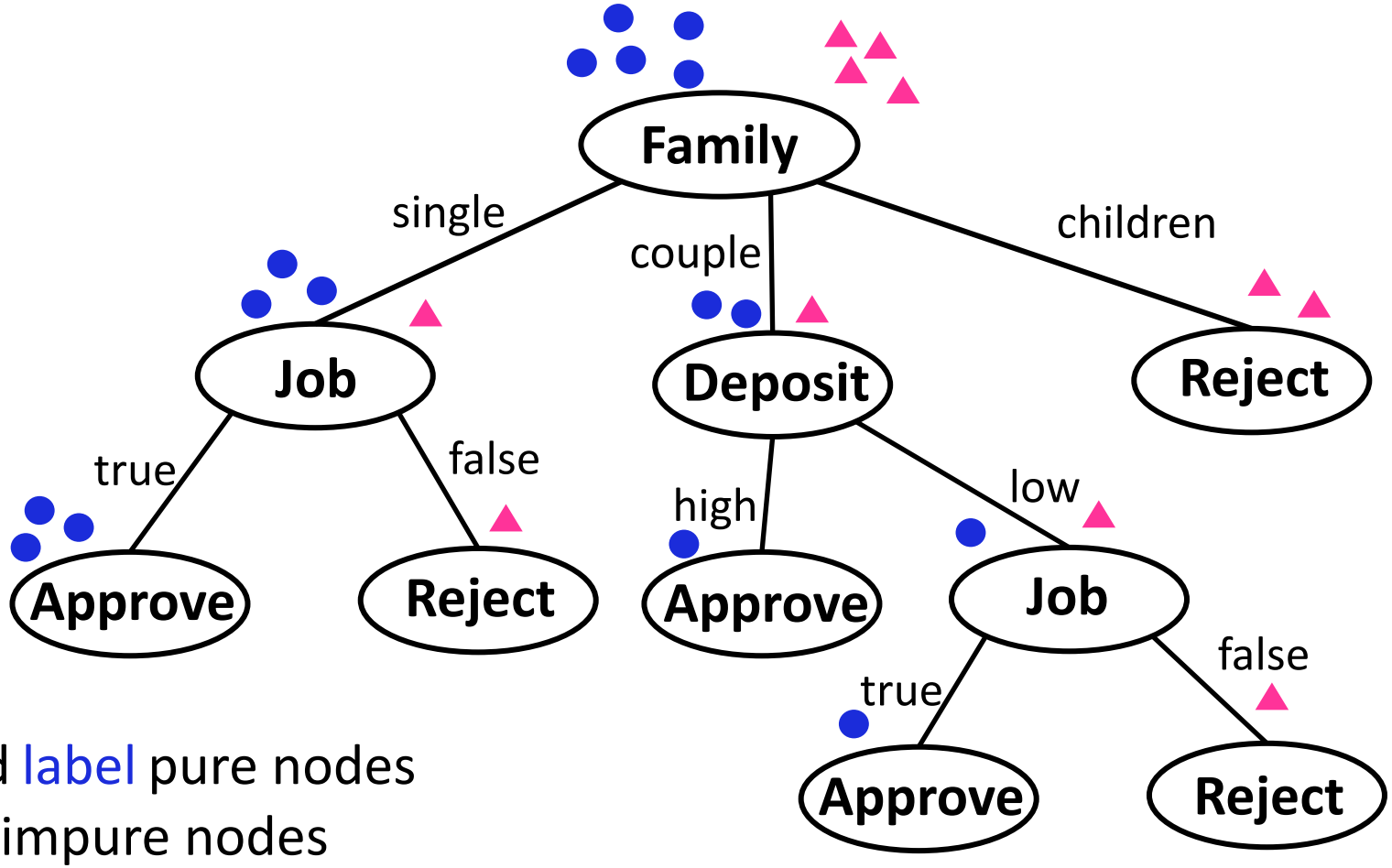
$imp(\text{node3}) = \frac{0}{2} * \frac{2}{2}$

$$imp(\text{Deposit}) = \frac{3}{9} * \frac{2}{3} * \frac{1}{3} + \frac{6}{9} * \frac{3}{6} * \frac{3}{6} = 0.24$$

$$imp(\text{Family}) = \frac{4}{9} * \frac{3}{4} * \frac{1}{4} + \frac{3}{9} * \frac{2}{3} * \frac{1}{3} + \frac{2}{9} * \frac{0}{2} * \frac{2}{2} = 0.15$$

imp(Family) < imp(Job) < imp(Deposit)

Learning a Decision Tree



- Identify and label pure nodes
- Recurse on impure nodes

  ---> Consider attributes “Job” and “Deposit”

Summary

- Part 1 and part 2 of assignment 1
- Part 3: Neural Networks:
  --- Perceptron learning (Monday)
  --- Back Propagation (Tuesday)