Homework 04

***Submission Notices*:**

- *Conduct your homework by filling answers into the placeholders in this file (in Microsoft Word format). Questions are shown in black color, instructions/hints are shown in italics and blue color, and your content should use any color that is different from those.*
- *After completing your homework, prepare the file for submission by exporting the Word file (filled with answers) to a PDF file, whose filename follows the following format,*
  <StudentID-1>_<StudentID-2>_HW04.pdf (Student IDs are sorted in ascending order)
  *E.g., **2112001_2112002_HW04.pdf***
  and then submit the file to Moodle directly WITHOUT any kinds of compression (.zip, .rar, .tar, etc.).
- *Note that you will get zero credit for any careless mistake, including, but not limited to, the following things.*
  1. *Wrong file/filename format, e.g., not a pdf file, use "-" instead of "_" for separators, etc.*
  2. *Disorder format of problems and answers*
  3. *Conducted not in English*
  4. *Cheating, i.e., copying other students' works or letting other students copy your work.*

Problem 1. (2pts) Aside from the types of learning introduced in the lectures (i.e., supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning), introduce two types of learning. For each type of learning, describe the **learning process** and **two applications**.

Please fill your answer in the table below

Types of learning	Description
1. Transfer Learning	*Learning process: Transfer learning is a machine learning method where a model developed for one task is used as a starting point for a model on a second task. Instead of starting the learning process from scratch, you start from patterns that have been learned when solving a different problem. This is particularly useful when you have limited data for the new task.*
	Application: 1. Natural Language Processing (NLP): Pre-trained models like BERT, GPT, and RoBERTa are examples of transfer learning. These models are initially trained on a large dataset to capture general language patterns and can then be fine-tuned on smaller datasets for specific tasks like sentiment analysis, text classification, or named entity recognition. 2. Image Classification: Models trained on large datasets, like ImageNet, capture general features from images. They can be used as a starting point and fine-tuned for a specific

	classification task, like detecting a particular type of defect on a production line or diagnosing medical images.
2. Meta-learning (or Learning to Learn)	*Learning process: In meta-learning, the goal is to train models on a variety of tasks in such a way that the model can quickly adapt to new, unseen tasks using minimal data. The model essentially learns the structure of learning itself. It doesn't just learn a specific task, but rather how to learn various tasks.*
	Application: 1. Algorithm Selection: For a given problem, meta-learning can be used to recommend a set of machine learning algorithms or hyperparameters that are likely to perform well. By observing the characteristics of the dataset and the performance of various algorithms on similar past tasks, meta-learning can guide the selection process. 2. Rapid Adaptation in Robotics: For robots that need to operate in diverse environments, meta-learning can be beneficial. After being trained across a range of tasks, such a robot can quickly adapt to a new task in a new environment, like manipulating a never-before-seen object, after only a few trials.

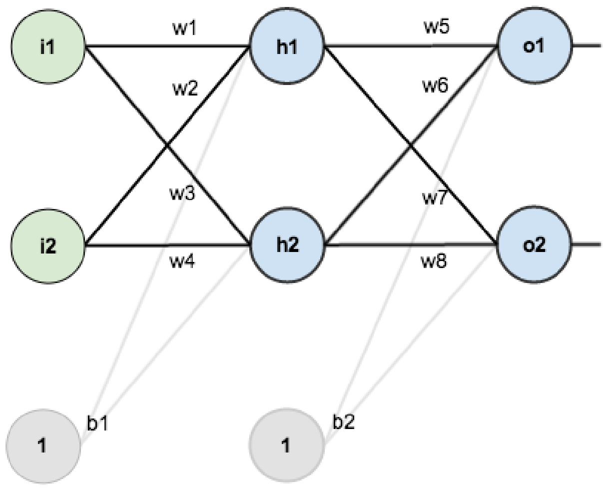
Problem 2. (3pts) Consider the following training dataset, in which **Transportation** is the target attribute. Show calculations to choose an attribute for the **root node** of the ID3 decision tree

Gender	Car Ownership	Travel Cost	Income Level	Transportation
Male	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Female	1	Cheap	Medium	Train
Female	0	Cheap	Low	Bus
Male	1	Cheap	Medium	Bus
Male	0	Standard	Medium	Train
Female	1	Standard	Medium	Train
Female	1	Expensive	High	Car
Male	2	Expensive	Medium	Car
Female	2	Expensive	High	Car

Please fill your answer in the white cells of the following table

		Counts			Metric values		
	Attribute values	**Bus**	**Car**	**Train**	**H**	**AE**	**IG**
Whole		**4**	**3**	**3**	**1.571**		
Gender (0.5pt)	Female	1	2	2	1.522	1.446	0.125
	Male	3	1	1	1.371		
Car Ownership (0.5pt)	0	2	0	1	0.918	1.036	0.535
	1	2	1	2	1.522		
	2	0	2	0	0		
Travel Cost (0.5pt)	Cheap	4	0	1	0.722	0.361	1.21
	Expensive	0	3	0	0		
	Standard	0	0	2	0		
Income Level (0.5pt)	Low	2	0	0	0	0.875	0.696
	Medium	2	1	3	1.459		
	High	0	0	2	0		

Problem 3. (3pts) Given a neural network with two inputs, two hidden neurons, two output neurons, as shown below. Additionally, in the hidden and output layers, each of which will include a bias that has a constant output value of 1.



- Learning rate 0.5
- Input values: i1=0.05 i2=0.10
- Target values: t1=0.01 t2=0.99
- Bias values: b1=0.35 b2=0.60
- Initial weight: w1=0.15 w2=0.20 w3=0.25 w4=0.30
  w5=0.40 w6=0.45 w7=0.50 w8=0.55

Present all calculations required to perform the backpropagation once (i.e. one forward pass and one backward pass) on the given neural network in the following cases

a) Ignore all biases (1.5pts)

Forward pass

	h1	h2	o1	o2
Sum	0.0275	0.0425	0.432530357	0.534280154
Sigmoid	0.506874567	0.510623401	0.606477732	0.630480835

Backward pass

w1	w2	w3	w4	w5	w6	w7	w8
0.14990587	0.19981174	0.24988759	0.29977519	0.36392146	0.41365463	0.52122763	0.57138462

b) Take into account all biases (1.5pts)

Forward pass

	h1	h2	o1	o2
Sum	0.3775	0.3925	1.10590597	1.2249214
Sigmoid	0.59326999	0.59688438	0.75136507	0.77292847

Backward pass

w1	w2	w3	w4	w5	w6	w7	w8	b1	b2
0.1497807	0.1995614	0.2497511	0.2995022	0.3589164	0.4086661	0.5113012	0.5613701	0.340637	0.5498

Problem 4. (2pts) You are given the following tables, which represent the outcomes of some functions. The functions take two values x and y and output the outcomes of the operations. Please identify **at least two models** for each of the functions that are perfectly represent the functions for some choice of parameters. Justify your answer. Note: there are no constraints on the architecture (e.g, the number of neurons, activation function, or the best splitting criterion), and the depth of decision tree is 0-index.

a) (1pt) $\boldsymbol{f(x, y) = x \oplus y}$.

$\boldsymbol{x}$	$\boldsymbol{y}$	$\boldsymbol{x \oplus y}$
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

☐ A neural network with no hidden layer

☐ A neural network with a single hidden layer

☐ A decision tree of depth one

☐ A decision tree of depth two

Explanation: A perceptron (no hidden layer neural network) $g(x, y) = \alpha(\beta(x, y))$ where $\beta(x, y) = ax + by + c$ and α is a (monotonic) activation function cannot correctly classify the XOR function. Note $\beta(1, 1) = a + b + c$ and $\beta(0, 0) = c$. WLOG, suppose $a + b + c \geq c$ (we can always flip the signs of all coefficients and reflect g to make this true). Suppose for contradiction that $(0, 0)$ and $(1, 1)$ are both correctly classified, then $\alpha(a + b + c) = 0 = \alpha(a)$. Since activation function α must be monotonic, it follows that all points $\alpha(x)$ for $x \in [c, a + b + c]$ must also be 0. Note that one of $\beta(1, 0) = a + c$ or $\beta(0, 1) = b + c$ must lie between $[c, a + b + c]$. Thus at least one of $(1, 0)$ or $(0, 1)$ is misclassified, giving a contradiction.

By contrast, a neural network with a hidden layer has sufficient capacity to represent XOR. Indeed, the universal approximation theorem shows that a single hidden layer network can represent any function given enough neurons. For XOR, a small network suffices: $g(x, y) = sgn(B \cdot sgn(A \begin{pmatrix} x \\ y \end{pmatrix}))$ where $sgn(x)$ is -1 when $x < 0$, 0 when $x = 0$ and $+1$ when $x > 0$ and $A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$, $B = (1 \quad 1)$.

A decision tree of depth one can only split on a single variable. Since XOR depends on the values of both variables, no tree of depth one can represent it.

A decision tree of depth two can represent any two-variable boolean function, including XOR

(1pt) $\boldsymbol{f(x, y) = \neg(x \vee y)}$

$\boldsymbol{x}$	$\boldsymbol{y}$	$\boldsymbol{\neg(x \vee y)}$
0	0	**1**

0	1	**0**
1	0	**0**
1	1	**0**

- ☐ A neural network with no hidden layer
- ☐ A neural network with a single hidden layer
- ☐ A decision tree of depth one
- ☐ A decision tree of depth two

Explanation: A single layer neural network $g(x, y) = sgn(-x - y + 1)$ classifies $f(x, y)$ correctly. A two layer neural network can also classify it (we can always make the second layer an identity). A decision tree of depth one cannot represent f since it depends on two variables. A decision tree of depth two can represent it (and any other two-variable boolean function)