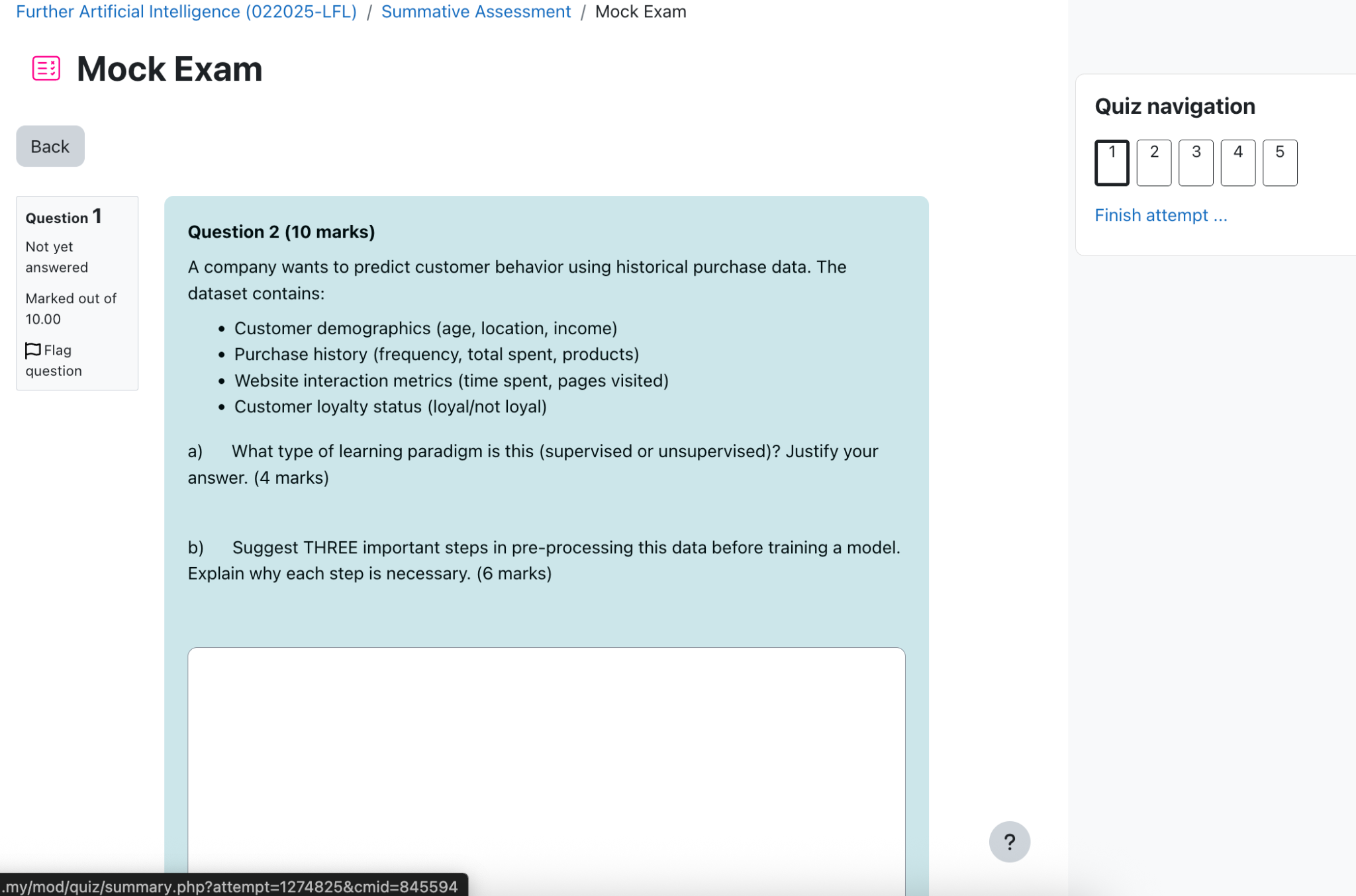
FAI mock test



Base on the information as provided in the question, this is a supervised learning.  
Because the goal is to predict customer behavior by obtaining pattern from historical purchase data. Supervised learning is used when the dataset includes labeled outputs (e.g., loyalty status or purchase outcomes) to train a model to map inputs (demographics, purchase history, website metrics) to these outputs.

b) Three important steps in pre-processing this data before training a model.   
1.To avoid missing values such as NULL value within the dataset. Sometimes the model can't handle NULL value causing a error and stop the model from proceeding next steps.  
2.To avoid extreme outlier. For example, if the customer spending behaviour varies too much, this doesn't bring the benefit of diverse dataset, but imbalance dataset. Things like this might effect the model performance. To overcome this scenario, we should do normalization , standardization to the dataset before training the model.

| Explanation: This is a supervised learning problem because: | |
| --- | --- |
|  | 1. Labeled Data: We have known outcomes (loyal/not loyal) for each customer. Think of this like having answer keys for a test - we know the correct answer for each example. |
|  | 2. Goal-Oriented Prediction: We're trying to predict something specific (loyalty) based on patterns in past data. It's like trying to predict if someone will enjoy a movie based on their past movie ratings |

3. Input-Output Pairing: Each customer example has both attributes (demographics, purchase history) and a known outcome (loyalty). This creates clear X → Y relationship.

4. Performance Measurement: We can objectively measure how good our model is by comparing its predictions to the actual loyalty statuses.

Contrast this with unsupervised learning, where we'd just look for natural groupings in customer data without knowing who's loyal upfront.

Part B: Data Preprocessing Steps

Explanation: Data preprocessing is like preparing ingredients before cooking. Here's why each step matters:

1. Handling Missing Values

Why it matters: Customer datasets often have gaps. Some customers don't provide all information, or tracking systems miss transactions.

How to do it:  
• For numerical data (like age or purchase amount): Use average values

• For categorical data (like location): Use most common value or create a new "unknown" category

Real-world example: If 30% of customers didn't provide their age, replacing those blanks with the average age (35) keeps those customers in our analysis.

2. Feature Scaling/Normalization

Notes:

Common feature scaling methods include:

Min-Max Scaling: Transforms reatures to a specific range (typically [0,1])

Formula: X scaled = (X -X\_min) / (X\_max - X\_min)

Standardization (Z-score normalization): Rescales features to have zero mean and unit variance

Formula: X scaled = (X - u) / o

Robust Scaling: Uses median and interquartile range instead of mean and standard deviation, making it less sensitive to outliers

Feature scaling is essential in machine learning pipelines, particularly when working with distance-based algorithms or when features have significantly different scales or units of measurement.

​​Normalization

Normalization in data processing is a technique that transforms data to fit within a specific range, typically (0,1] or (-1,1]. This process adjusts values measured on different scales to a common scale, making them comparable and suitable for various computational methods.

The primary purposes of normalization include:

1. ﻿﻿﻿Improving algorithm performance: Many machine learning algorithms perform better when features are on a similar scale. This prevents features with larger ranges from dominating the learning process.
2. ﻿﻿﻿Speeding up convergence: Normalized data often leads to faster convergence in gradient-based optimization methods.
3. Preventing numerical instability: Features with very large values can cause numerical issues in computation.

Common normalization techniques include Min-Max scaling (rescaling data to a fixed range), Z-score normalization (standardizing data to have zero mean and unit variance), and Decimal scaling (moving the decimal point of values).

In neural networks specifically, normalization is crucial as it helps prevent issues like vanishing or exploding gradients and allows the network to learn more effectively from diverse input features.

Why it matters: Different measurements use drastically different scales.

Income might be thousands of dollars while website visits might be 1-10.

• How to do it:

Min-Max scaling: Convert all values to a 0-1 scale

Standardization: Convert to values with average=0 and standard

deviation=1

Real-world example: A customer with $100,000 income and 5 website visits would have income dominating any calculations without scaling.

3. Encoding Categorical Variables

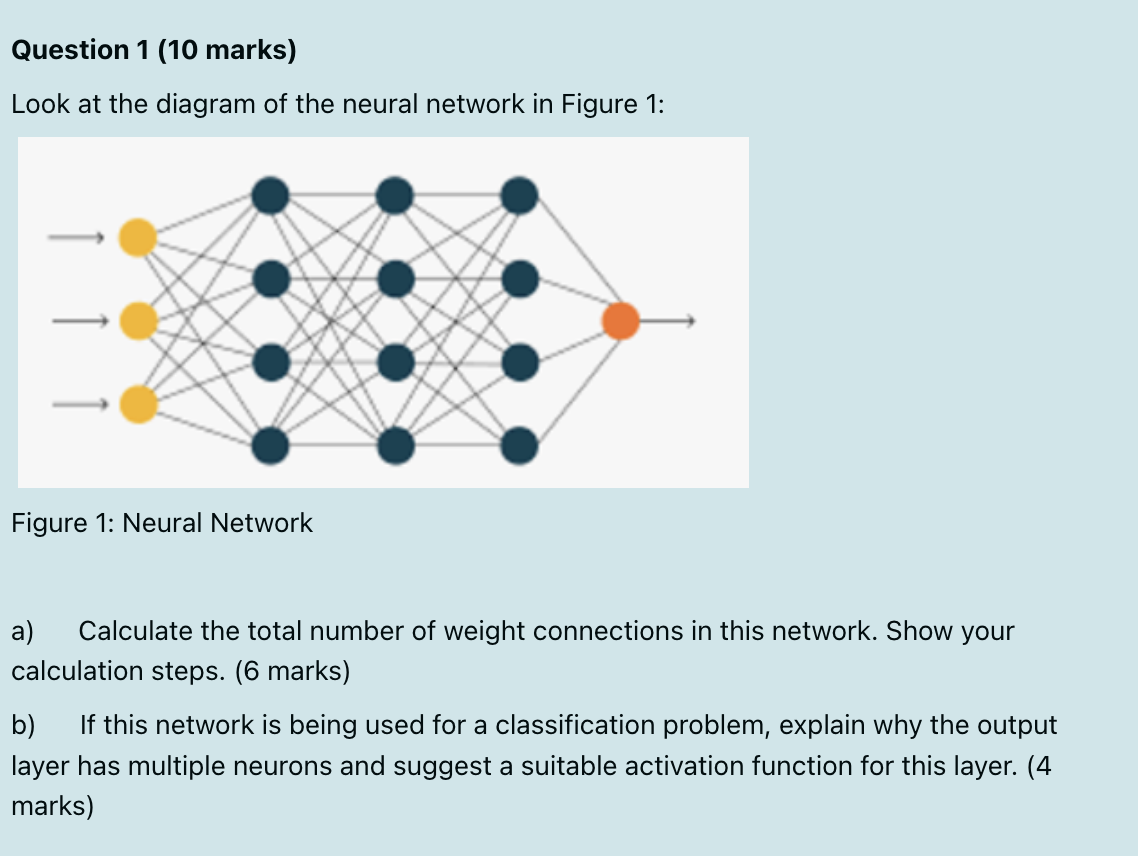
**Why it matters:**

Computers need numbers, not text. Variables like "location" or "product type" need conversion.

**How to do i**t:

* ﻿﻿One-hot encoding: Create separate yes/no columns for each category
* ﻿﻿Label encoding: Assign numbers (1, 2, 3...) to each category

**﻿﻿Real-world example:** Converting "Location" from text values (Urban, Suburban, Rural) to numeric values the model can process.

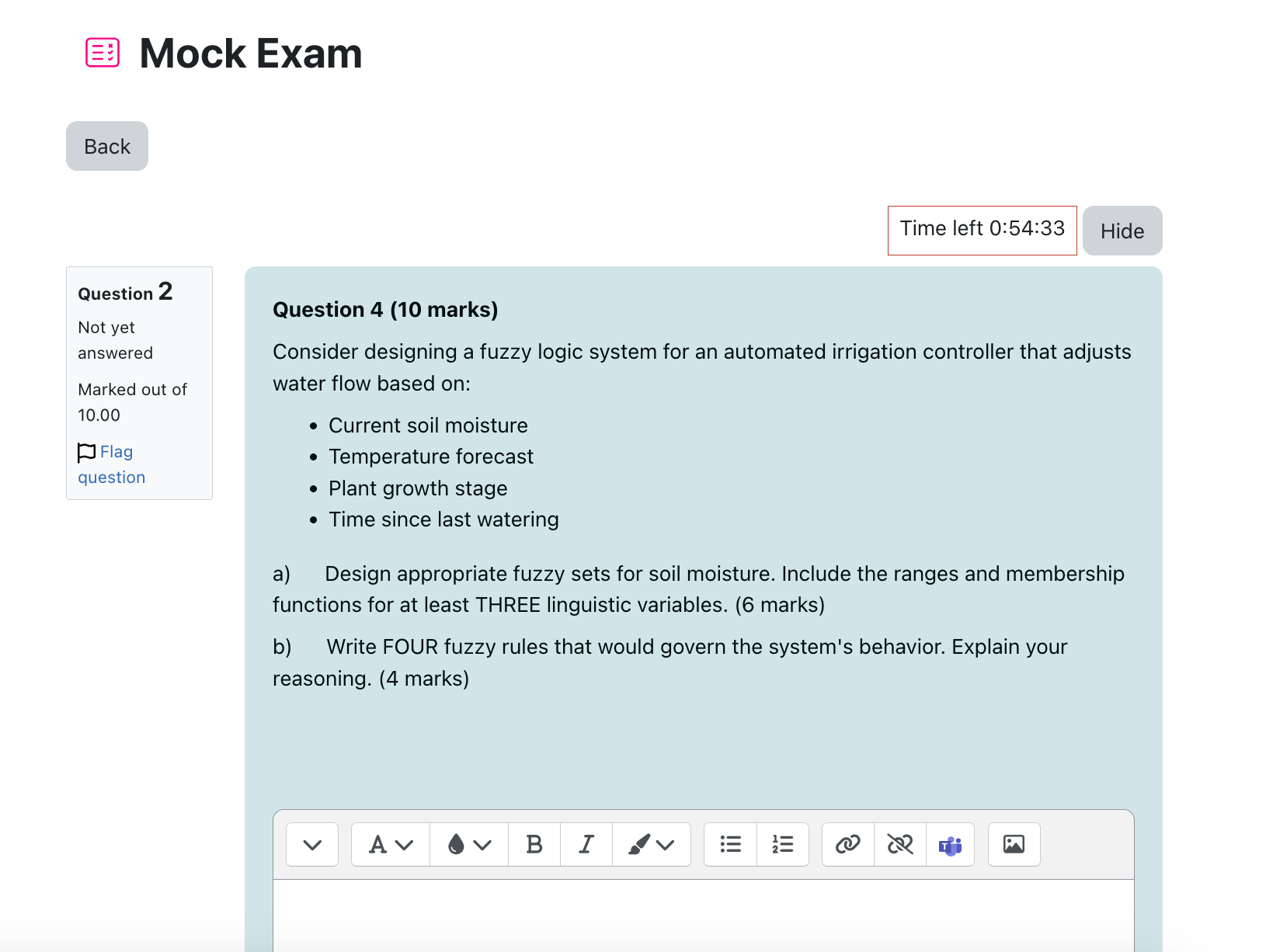


Part A:  
 3 input neuron  
12 hidden neuron(dark  
1 output neuron(ornage)  
  
Total Connection = Total 3x12=36

Total weight connection = 36+12=48

Part B

With a single output neuron the network is likely being used fr binary classification rather then multiclass classification,The network would predict whether an input belongs to a specific class 1 or not 0  
  
For a binary classification problem with a single noutput neurin, a suitable activation function would be sigmoid (logistic function). The sigmoid function is appropriate because:  
1. It mays any input vlue between 0 and 1  
2. The output can be interpreted as a probability. The probability output tells us about the model’s confidence , an OUTPUT OF 0.51 VS 0.99 BITH CLASSIFY AS POSITIVE BUT THE LATTER INDICATES MUCH HIGHER CONFIDENCE.  
[3.It](http://3.it)’s pdocus a smooth gradient which is helpful during training. During backpropagation, the network



Part A: Soil Moisture Fuzzy Sets

Explanation: Fuzzy logic allows us to model the real world's "fuzziness" instead of strict yes/no boundaries. For soil moisture:

Dry (0% to 30% moisture)

• Think of this as "definitely dry" at 10-15% moisture

• From 0-10%, it gradually becomes "more dry"

* ﻿﻿From 15-30%, it gradually becomes "less dry"
* ﻿﻿Visual representation: Trapezoidal shape with flat top at 10-15%
* ﻿﻿Real-world meaning: Plants are likely experiencing water stress

Moderate (20% to 60% moisture)

• "Perfect moisture" is at 40%

* ﻿﻿From 20-40%, it gradually becomes "more moderate"
* ﻿﻿From 40-60%, it gradually becomes "less moderate"
* ﻿﻿Visual representation: Triangle with peak at 40%

• Real-world meaning: Optimal growing conditions for most plants

Wet 50%-100%

* "Definitely wet" above 75% moisture
* ﻿﻿From 50-75%, it gradually becomes "more wet"
* ﻿﻿Visual representation: Trapezoidal shape with flat top from 75-100%
* ﻿﻿Real-world meaning: Soil may be too wet, risking root rot

Important concept: Notice the overlaps! A 25% moisture reading is both "somewhat dry" and "somewhat moderate" simultaneously. This is the key insight of fuzzy logic - things can partially belong to multiple

Part B: Fuzzy Rules

Explanation: Fuzzy rules connect inputs to outputs using natural language-like rules that mimic human expert thinking.

Rule 1: IF soil moisture IS dry AND temperature forecast IS high THEN water flow IS high

Plain language: "When soil is dry and hot weather is coming, water heavily"

• Reasoning: Plants face double stress from both current dry conditions and increased

future evaporation

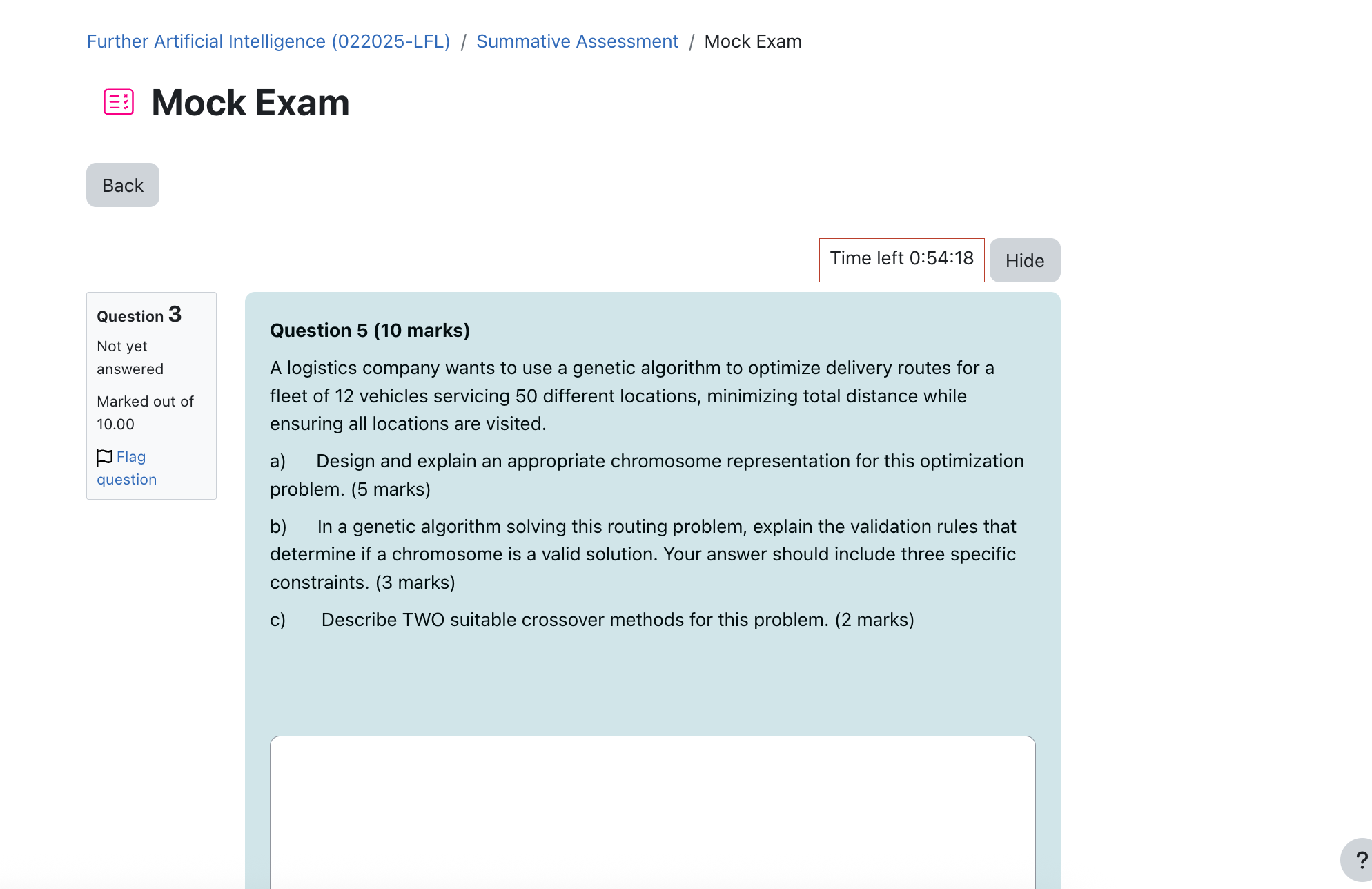
• Real-world parallel: Just like vou'd drink extra water before exercising in hot weather

Rule 2: IF soil\_ moisture IS moderate AND plant\_growth stage IS flowering THEN

water flow IS medium

• Plain language: "When soil has some moisture and plants are flowering, water moderately"

• Reasoning: Flowering is a critical stage with specific water needs - too little stresses the plant, too much damages flowers



Part A: Chromosome Representation (5 marks)

A suitable chromosome representation for this problem would be a permutation-based encoding with vehicle delimiters:

1. The chromosome would be an array of integers where:

• Each integer from 1 to 50 represents a location

Special delimiter values (e.g., 0 or -1) separate routes for different vehicles

For example, a chromosome might look like:

[12, 45, 3, 0, 22, 17, 9, 0, 31, 42, 11, 27, 0, ...]

Where 0 serves as a delimiter between vehicle routes, meaning:

* ﻿﻿Vehicle 1 visits locations 12, 45, 3
* ﻿﻿Vehicle 2 visits locations 22, 17, 9
* ﻿﻿Vehicle 3 visits locations 31. 42. 11. 27

• And so on...

This representation directly encodes both the assignment of locations to vehicles and the

sequence of visits for each

This representation directly encodes both the assignment of locations to vehicles and the sequence of visits for each vehicle.

Part B: Validation Rules/Constraints (3 marks)

Three specific constraints that determine if a chromosome is valid:

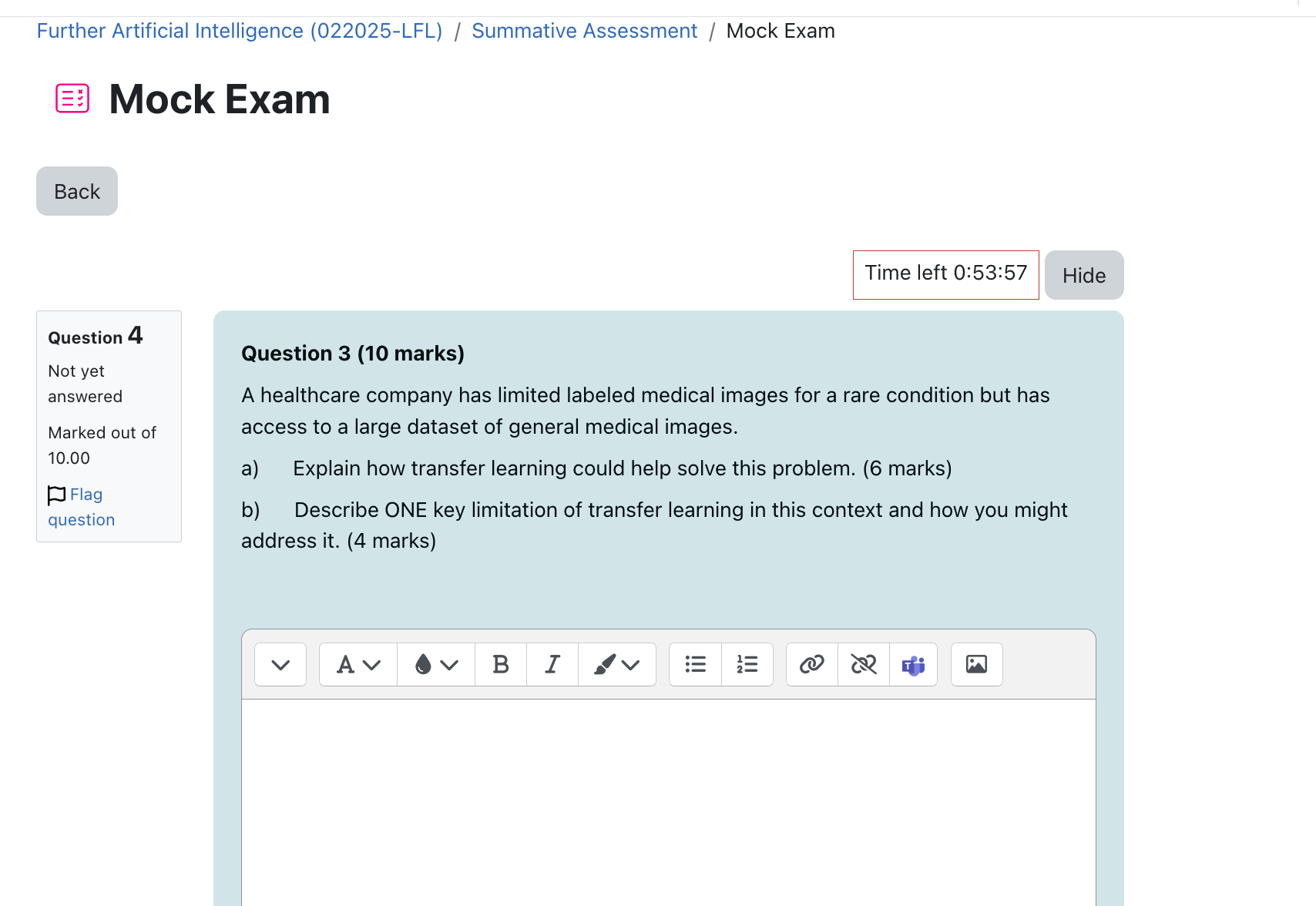
|  | 1. Completeness constraint: Every location (1-50) must appear exactly once in the romosome, ensuring all locations are visited. |
| --- | --- |
|  | 2. Vehicle capacity constraint: Each vehicle has a maximum number of locations it can service based on factors like time, fuel, or cargo capacity. The number of locations between delimiters cannot exceed this maximum. |
|  | 3. Route feasibility constraint: Each route must be physically possible to complete. This could include time window constraints (certain locations must be visited during specific hours) or logical sequence constraints (e.g., pickup must occur before delivery).  Iwo sullable crossover metnoas lor this proviem.  1. Route-Based Crossover:  Select random vehicles/routes from each parent  Exchange these routes between parents  Repair the resulting offspring by removing duplicates and adding missing  locations  2. Partially Mapped Crossover (PMX):  Select a segment of the chromosome from one parent  Map the locations in this segment to the corresponding positions in the second parent  Resolve conflicts to maintain the validity of the permutation  • Preserve vehicle delimiters during the crossover process  Both methods preserve the integrity of good route segments while exploring new combinations of location assignments and visit sequences. |

a.)Each chromosome is a sequence of integers representing the order of visiting the 50 locations. For 12 vehicles, the chromosome can be divided into 12 segments and each representing a vehicle’s route,or maybe node, with a special delimiter (example: 0) to separate vehicles.We can use each number (1 to 50) represents a unique location. “0” separates routes for different vehicles (alternatively, fixed-length segments can be used). For 12 vehicles, the chromosome ensures all 50 locations appear exactly once (permutation) and are split among vehicles.

(Miss this one not sure need discuss)

b) All Locations Visited Exactly Once: The chromosome must include all 50 location IDs (1 to 50) exactly once to ensure every location is serviced without duplication. Valid Vehicle Assignment: The chromosome must be divisible into 12 non-empty sub-routes (one per vehicle) The total distance of each sub-route must be calculable based on location coordinates.

Order Crossover (OX): Select two random crossover points in the parent chromosomes. Copy the segment between these points from Parent 1 to Child 1, and fill the remaining positions with the missing locations in the order they appear in Partially Matched Crossover (PMX): Both emphasize that you swap a piece while making sure the final list or route is still valid (maintaining the original items without repeats or omissions).



a.) Usually transfer learning use pre-trained model which they trained on large scale of dataset. While the example above shoes a limited labeled medical images but has access to a large dataset.In this case we doesn't need a RAG or retrained the whole model. What we need to do just train on particular layer of the pre-trained model. For example, we can use the last layer of the pre-trained model, feed with the small scale medical images , train the last layer to make it adapt with the newly feeded dataset. So in this approach we didn't spend hours of time just to retrain a pre-trianed model with limited dataset. We can maximize their function from both side.

b.) Key limitation of transfer learning Limitation:

Even though we leverage the benefit of transfer learning, they still have some limitation. As pre-trained model are usually trained on large scale of data, it performs well on general response. Even if we train its last layer , this might be insufficient for the pre-trained model to process the niche info.

To adress this issue, we can perform additional pre-training or fine-tuning on the large dataset of general medical images before fine-tuning on the labeled dataset for the rare condition.

Part A: Transfer Learning Approach

Explanation: Transfer learning is like learning to play a new sport after already mastering another sport - your general athletic skills transfer over. Here's how it works for medical

images:

1. Pre-training Phase: First, train a neural network on a large dataset of general medical images.

This builds a foundation of understanding - the network learns to identify

Iges, shapes, textures, and patterns common in medical imagery.

Think of this as learning "medical visual vocabulary."

2. Feature Extraction: The early layers of the network learn generalized features.

* These layers recognize basic elements like edges and shapes that appear in all

medical images.

* Visual example: Both brain scans and lung X-rays contain edges, contrasts between tissues, and curved anatomical structures.

3. Layer Freezing Strategy: Keep the weights of early layers fixed.

• This preserves the general knowledge while preventing the limited rare condition data from causing overfitting.

Analogy: Keep your basic athletic skills while learning specific techniques for a

new sport.

4. Fine-tuning Process: Only update the weights of later layers.

These layers learn to recognize specific patterns unique to the rare condition.

Using less data works because we're only teaching the "specialization"

5.Practial benefit:

Need significantly less data for the specific task

Training converges faster

Results in better performance than training from scratch

A visual timeline might help students understand: Start with general model → Freeze early

layers → Train later layers on specific data.

Part B: Limitation and Solution

Explanation: The main limitation of transfer learning is Domain Shift. Domain shift refers to a phenomenon in machine learning where the statistical properties of the data change between the training (source) and deployment (target) environments. This mismatch between training and real-world data can significantly degrade model performance.

The Problem (Domain Shift): Even though we're working with medical images in both cases, there are important differences between the source (general medical images) and target domains (rare condition images):

* Different imaging equipment produces different image characteristics
* Different body parts have unique visual properties
* Different preprocessing techniques alter image appearance
* Different contrast levels and resolution

It's like trying to apply skills from road cycling to mountain biking - the basics are similar, but the terrains are different enough to cause problems.

Solution: Domain Adaptation Techniques

1. Gradient Reversal Layers:

These special layers actually work to make features "domain-blind"

They penalize the network for learning features that are specific to just one

domain

Real-world example: Force the network to focus on tissue abnormalities ather than image brightness differences

2. Data Augmentation for Domain Alignment:

Votes: Domain alignment is a set of techniques used in machine learning to addres. he problem of domain shift by making the feature representations of source anc target domains more similar. It aims to improve model generalization across different domains by minimizing the distributional differences between them.

* Create variations of the rare condition images that more closely resemble the

general images

* Apply specific transformations that address the differences
* Example: If source images are brighter, darken the target images during training

3. Progressive Fine-tuning:

Start by only training the final layer

• Gradually unfreeze and train more layers as you collect more data

Think of it as slowly allowing more fundamental changes to the model