Data Structures and Algorithms
with applications in Machine Learning
- MCQ 4 -

NAME: ____________________ **GROUP:** ____________________

Each Question: 1 Mark **Duration:** 20 Minutes

Completely fill the circles as shown: ○○●○

Answer sheet

Q1. ○ a. ○ b. ○ c. ○ d.

Q2. ○ a. ○ b. ○ c. ○ d.

Q3. ○ a. ○ b. ○ c. ○ d.

Q4. ○ a. ○ b. ○ c. ○ d.

Q5. ○ a. ○ b. ○ c. ○ d.

Q6. ○ a. ○ b. ○ c. ○ d.

Q7. ○ a. ○ b. ○ c. ○ d.

Q8. ○ a. ○ b. ○ c. ○ d.

Q9. ○ a. ○ b. ○ c. ○ d.

Q10. ○ a. ○ b. ○ c. ○ d.

The Quiz

Problem Description

We consider a graph $G = (V, E)$ representing individuals on a social media platform, where each node $i \in V$ corresponds to a person, and each edge $(i, j) \in E$ represents a connection between two individuals. The nodes in this graph are categorized into two groups:

- Nodes labeled 1: People who like the singer Adele.
- Nodes labeled 0: People who do not like Adele (although, let's face it, they must have questionable taste because Adele is absolutely phenomenal!).

The dataset derived from this graph is divided into two subsets:

- **Training set:** Contains n_1 samples and is used to train the predictive model. The training samples are represented in figure 1 by nodes that are **green** (for individuals labeled 1: liking Adele) and **red** (for individuals labeled 0: not liking Adele).
- **Test set:** Contains n_2 samples and is used to evaluate the model's performance. The test samples are represented in figure 1 by nodes that are **gray**.

Notably, the dataset is highly imbalanced, with 99% of the data labeled as 1 (people who like Adele).

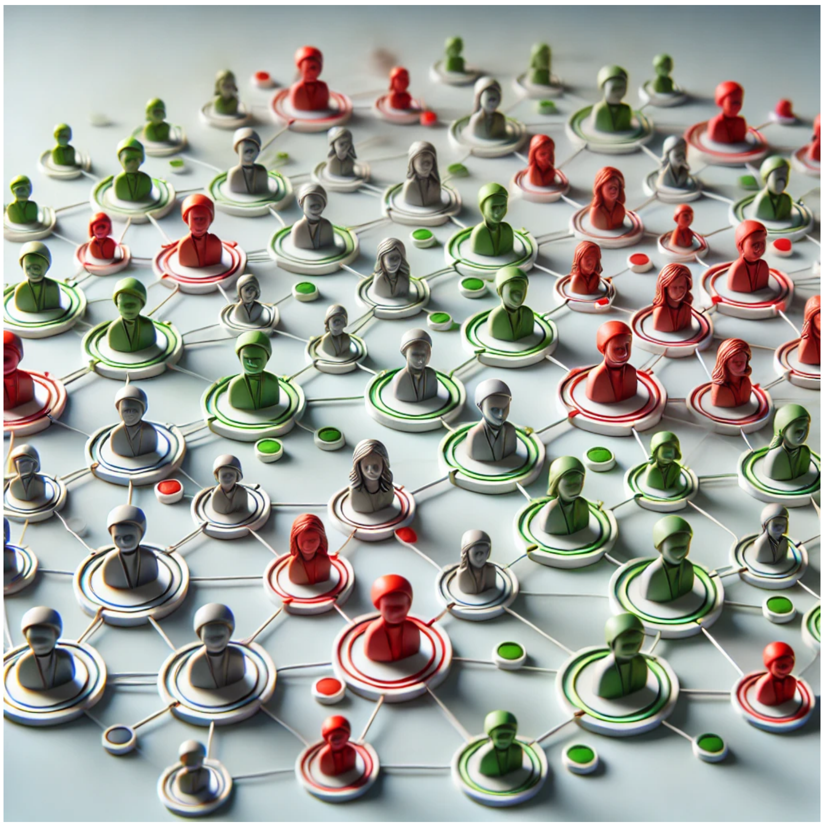


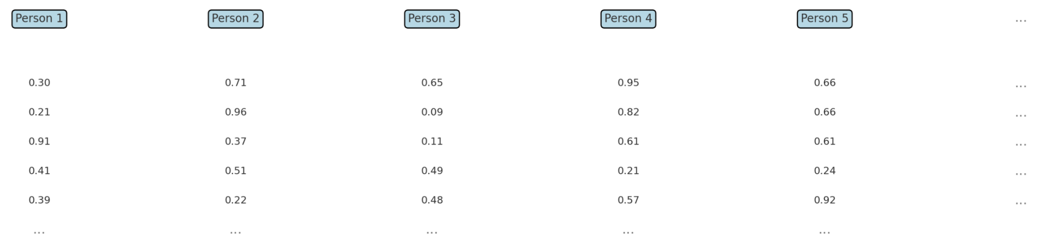
Figure 1: Representation of the social media graph. Nodes in **green** and **red** represent the training samples for individuals labeled 1 (liking Adele) and 0 (not liking Adele), respectively. Nodes in **gray** represent the test samples. Connections between nodes represent social interactions.

For each node $i \in V$, we generate a D-dimensional embedding vector, $Z_i \in \mathbb{R}^D$, using the mapping:

$$\xi : i \in V \to Z_i \in \mathbb{R}^D$$

Here, ξ transforms each node i into its corresponding D-dimensional embedding vector Z_i. This embedding captures structural and relational information from the graph.

Figure 2 illustrates the D-dimensional embedding vector Z_i for each individual node $i \in V$.



Person 1	Person 2	Person 3	Person 4	Person 5	...
0.30	0.71	0.65	0.95	0.66	...
0.21	0.96	0.09	0.82	0.66	...
0.91	0.37	0.11	0.61	0.61	...
0.41	0.51	0.49	0.21	0.24	...
0.39	0.22	0.48	0.57	0.92	...
...	...	...	...	...	

Figure 2: Embedding Vectors Representing Individuals in the Graph

The objective is to predict the label $y_i \in \{0, 1\}$ for each node based on its embedding vector Z_i. To achieve this, we define the mapping:

$$\phi : Z_i \in \mathbb{R}^D \rightarrow y_i \in \{0, 1\}$$

Specifically, we search for ϕ within the space of **Random Forest Classifiers** to map the embedding Z_i to its corresponding label y_i.

Understanding the problem

Q. 1 Recall the transformation process:

$$\xi : i \rightarrow Z_i, \quad \phi : Z_i \rightarrow y_i$$

In this process: ξ maps a node i in the graph to its D-dimensional embedding vector $Z_i \in \mathbb{R}^D$, which captures the structural and relational information of the graph.

ϕ maps the embedding vector Z_i to the label $y_i \in \{0, 1\}$.

Based on this description, which of the following correctly identifies the type of learning associated with ξ and ϕ?

○ a. ξ represents "Unsupervised Learning", and ϕ represents "Supervised Learning".

○ b. Both ξ and ϕ represent "Unsupervised Learning".

○ c. Both ξ and ϕ represent "Supervised Learning".

○ d. ξ represents "Supervised Learning", and ϕ represents "Unsupervised Learning".

Q. 2 The mapping ξ is responsible for transforming each node in the graph into a D-dimensional embedding vector. This embedding vector, $Z_i \in \mathbb{R}^D$, captures the structural and relational information of the graph. The role of ξ is conceptually similar to the GloVe approach in Natural Language Processing, which learns vector representations of words by leveraging co-occurrence patterns.

In the context of the GloVe algorithm, what is the equivalent to the nodes i in a graph?

○ a. Co-occurrence matrices themselves, as they represent pairwise relationships.

○ b. Words in a corpus, where their co-occurrence relationships define their embeddings.

○ c. Context windows, where embeddings are derived directly from fixed-size contexts.

○ d. Frequency counts, where embeddings are assigned based purely on occurrence frequencies.

Q. 3 Recall that the Random Forest Classifier is trained to predict the label $y_i \in \{0, 1\}$ for each node i based on its D-dimensional embedding vector $Z_i \in \mathbb{R}^D$. The training dataset consists of n_1 samples, where each sample is a pair (Z_i, y_i).

What is the shape of the training data used to train the Random Forest Classifier?

○ a. A matrix of shape (n_1, n_1) for features and a vector of length D for labels.

○ b. A matrix of shape (D, n_1) for features and a vector of length D for labels.

○ c. A matrix of shape (n_1, D) for features and a vector of length n_1 for labels.

○ d. A single matrix of shape $(n_1, D + 1)$, combining features and labels together.

The Random Forest Classifier

Q. 4 Decision Trees are known to have a tendency to overfit the training data, especially when the tree is deep. Random Forest addresses this issue by:

○ a. Combining multiple Decision Trees through bagging, where each tree is trained on a random subset of the training data.

○ b. Training a single shallow Decision Tree with reduced depth to minimize overfitting.

○ c. Using a single Decision Tree but regularizing it by pruning unnecessary branches.

○ d. Combining multiple Decision Trees through boosting, where each tree corrects the errors of the previous one.

Q. 5 The following pseudo-code implements the training process for a Random Forest Classifier.

```
Algorithm 1 Random Forest Training
Require: Training data $X$, labels $y$, number of trees $T$, number of features $F$
Ensure: Trained Random Forest model with $T$ Decision Trees
1: Initialize an empty list of trees: forest ← []
2: for $t = 1$ to $T$ do
3:     bootstrap_sample ← random_subset($X, y$)        ▷ Sample data with replacement
4:     selected_features ← ...                          ▷ Fill in the blank
5:     tree ← train_decision_tree(bootstrap_sample, selected_features)
6:     forest.append(tree)
7: end for
8: return forest
```

What should replace the blank to ensure the Random Forest Classifier selects a random subset of features for each tree?

○ a. random.sample(all_features, F)   ▷ Select F random features

○ b. all_features ▷ Use all features without selecting a subset

○ c. random.shuffle(all_features) ▷ Shuffle features but use all of them

○ d. top_features(F) ▷ Select the top F features by importance

Q. 6 Which of the following is **not** a hyperparameter of a Random Forest Classifier?

○ a. `n_estimators`

○ b. `max_features`

○ c. `min_samples_leaf`

○ d. `learning_rate`

Choosing the Evaluation Metric

Selecting an appropriate evaluation metric is crucial for assessing the performance of a machine learning model. By carefully selecting the right metric, we can ensure that the model's performance aligns with the real-world objectives and constraints of the problem.

Q. 7 Given that the dataset is highly imbalanced, with 99% of the labels corresponding to individuals who like Adele, which of the following evaluation metrics is **not suitable** for assessing the model's performance?

○ a. Precision

○ b. Recall

○ c. Accuracy

○ d. F1-score

Q. 8 Which of the following statements about the AUC (Area Under the Curve) is **not true**?

○ a. AUC evaluates the model's performance across all possible decision thresholds.

○ b. AUC quantifies the likelihood that a randomly chosen positive sample is assigned a higher score than a randomly chosen negative sample.

○ c. AUC is the area under the ROC curve, where the ROC plots the True Positive Rate (TPR) against the False Positive Rate (FPR).

○ d. AUC is sensitive to class imbalance and may significantly degrade in performance when one class dominates.

Tuning Hyperparameters using Cross Validation

Hyperparameter tuning is a crucial step in optimizing the performance of a Random Forest Classifier. In this section, we aim to tune two key hyperparameters: the number of trees (`n_estimators`) and the maximum depth of each tree (`max_depth`). To achieve this, we use Cross Validation, a technique that splits the training data into several folds to evaluate the performance of different hyperparameter combinations. By systematically searching through these combinations, we can identify the configuration that maximizes the model's performance while minimizing overfitting or underfitting.

Q. 9 In the Cross Validation process, we aim to evaluate the model's performance for different hyperparameter combinations and select the best one. Below is the pseudo-code for hyperparameter tuning using Cross Validation:

Algorithm 2 Hyperparameter Tuning with Cross Validation

```
Require: Training data $X$, labels $y$, hyperparameter grid $H$, number of folds $k$
Ensure: Best hyperparameter combination
 1: Initialize best_score ← 0, best_params ← None
 2: for each ($n\_estimators, max\_depth$) in $H$ do
 3:     scores ← []
 4:     for fold = 1 to $k$ do
 5:         Split $X$ and $y$ into training and validation sets: $(X_{\text{train}}, y_{\text{train}}, X_{\text{val}}, y_{\text{val}})$
 6:         Train a Random Forest with $n\_estimators$ and $max\_depth$ on $(X_{\text{train}}, y_{\text{train}})$
 7:         Compute validation score and store: scores.append(evaluate(model, X_val, y_val))
 8:     end for
 9:     avg_score ← mean(scores)
10:     if avg_score > best_score then
11:         best_score ← avg_score
12:         ...                                                        ▷ Fill in the blank
13:     end if
14: end for
15: return best_params
```

Which of the following best replaces the blank to correctly update the best hyperparameters?

○ a. `best_params = (n_estimators, max_depth)`

○ b. `best_params = (X_train, X_val)`

○ c. `best_params = avg_score`

○ d. `best_params = None`

Adjusting the Threshold of the Random Forest Classifier

In this scenario, we use Node Classification to predict whether an individual in a social media graph likes Adele or not. The classification task serves as the foundation for a recommendation system. By identifying individuals who like Adele, we can target them with personalized recommendations, such as suggesting Adele's music, concert tickets, or related content. However, the effectiveness of these recommendations depends on the accuracy of the classifier and the threshold used for making predictions.

The threshold in a Random Forest Classifier determines the balance between False Positives (predicting someone likes Adele when they do not) and False Negatives (predicting someone does not like Adele when they actually do). Adjusting this threshold is crucial because these two types of errors have different business implications:

- **False Positives (FP): Predicting someone likes Adele when they do not.** In this case, the recommendation is irrelevant to the individual, and they are likely to ignore it. The cost of this error is low since it does not negatively affect the user experience or business outcomes significantly.
- **False Negatives (FN): Predicting someone does not like Adele when they actually do.** This error results in missed opportunities, as a potential fan or customer is not reached

with relevant recommendations. The business impact is higher because it reduces the system's effectiveness in engaging users and growing Adele's audience or related revenue streams.

To optimize the recommendation system, it is essential to find a threshold that aligns with the business goal of maximizing engagement. For example, setting a threshold to maintain a minimum recall ensures that the majority of true fans are identified and targeted, even if it allows for a slightly higher false positive rate.

Q. 10 Given that the cost of false negatives is higher in this case, we aim to maintain a minimum recall of 80% to ensure we capture most of the positive cases (people who like Adele). Based on the table below, which threshold (τ) is the best choice ?

Threshold (τ)	True Positive Rate (TPR / Recall)	False Positive Rate (FPR)
τ_1	0.75	0.01
τ_2	0.78	0.07
τ_3	0.80	0.10
τ_4	0.80	0.15

Table 1: Thresholds and their corresponding TPR and FPR

○ a. τ_1

○ b. τ_2

○ c. τ_3

○ d. τ_4