1st Iran AI Olympiad — Problems and Solutions

2nd Round — Part I (Multiple Choice Section)
Academic Year 1403–1404 / 2024–2025

Shayan Shahrabi
shayanshahrab@gmail.com

Last updated: 19 July 2025

This is a compilation containing problems and solutions from the Multiple Choice questions of the **1st IOAI (Iranian Olympiad in Artificial Intelligence)**. The ideas of the solution are a mix of my own work and solutions found by the community. However, all the writing and translation of the questions to English is maintained by me. I've tried to answer in a neat, easy to understand way, while mentioning some notes and/or resources for future reading for some of the questions.

Corrections and comments are welcome!

"If I have seen further, it is by standing on the shoulders of giants."
— Sir Isaac Newton

Introduction

The second round of the 1st Iran AI Olympiad took place on April 17, 2025. This round consisted of two parts. The first part was a multiple-choice exam with 15 questions and a duration of 120 minutes. The second part, held two hours after the first, was a written section featuring 4 descriptive questions, with a time limit of 150 minutes.

Contents

Question 1

Figure 1 shows the data points for a KNN classification to 2 black and white classes task. The dotted data points are the ones not classified yet. If the KNN algorithm is used with $k = 1$ and $k = 3$, which classes the data points 1 and 2 will be allocated to?

(A) For $k = 1$, Data 1 is black and Data 2 is black. For $k = 3$, Data 1 is black and Data 2 is white.

(B) For $k = 1$, Data 1 is black and Data 2 is black. For $k = 3$, Data 1 is black and Data 2 is black.

(C) For $k = 1$, Data 1 is black and Data 2 is black. For $k = 3$, Data 1 is white and Data 2 is white.

(D) For $k = 1$, Data 1 is white and Data 2 is white. For $k = 3$, Data 1 is black and Data 2 is white.

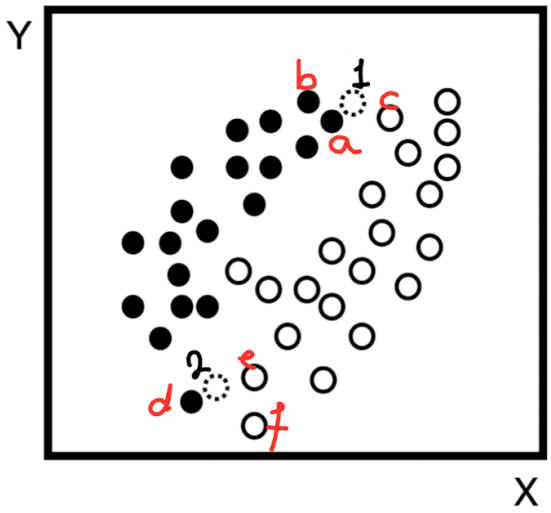


Figure 1: Data Points for KNN Classification Problem

Correct Answer: A For $k = 1$, data point 1 will choose it's closest neighbor (point a) and point 2 will choose point d. Thus they will be classified both as **black**.

For $k = 3$, point 1 will choose among the points a, b and c with the majoriy being black (and thus classified as **black**) while point 2 will select among points d, e and f and being classified as **white**.

Question 2

Imagine the 2D space (x_1, x_2). For solving a classification problem in this space, a decision tree is used. Each node in this tree checks a condition of the form $x_i < C$ where x_i is one of the features x_1 or x_2 and C is a constant. Define the complexity of the tree to be the number of its nodes and suppose that the current tree has the least complexity among all the possible trees.
Now consider the following two transformations of the input data:

1. Normalize the input data, i.e., transform them to have mean 0 and variance 1.
2. Rotate the input data by 45° around the origin.

Suppose after applying each of the above transformations, we train a new decision tree on the transformed data. Which of the following statements about the new trained tree is correct?

(A) After applying the first transformation, we can achieve the performance of the original tree with a tree of lower complexity.

(B) After applying the second transformation, we cannot determine whether the new tree will be more complex or simpler than the original one.

(C) After applying the second transformation, we can achieve the performance of the original tree with a tree of lower complexity.

(D) After applying the first transformation, we cannot determine whether the new tree will be more complex or simpler than the original one.

Correct Answer: B

Question 3

An online streaming company wants to build a machine learning model to predict customer churn (likelihood of subscription cancellation). The goal is to identify which customers are more likely to cancel so the company can offer personalized promotions to prevent churn.
Two models (Model 1 and Model 2) have been trained and their calibration curves are shown in Fig. 2. The company wants to decide which model is better to use. Which one should they prefer, and why?
Notes:

1. First, customers are ranked based on their predicted probability of churn, from highest to lowest.
2. Then, they are divided into K equal groups.
3. For each group, the proportion of customers who actually churned is plotted on the y-axis:

$$x_i = \frac{\text{Number of customers in group } i \text{ who churned}}{\text{Total number of customers in group } i}, \quad \forall i \in [1, K]$$

4. For each group, the average predicted churn probability is plotted on the x-axis:

$$y_i = \frac{\sum_{c \in \text{group } i} p_c}{\text{Total number of customers in group } i}, \quad \forall i \in [1, K]$$

5. Finally, the points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_K, y_K)\}$ are plotted to form the calibration curve.

(A) Model 1

(B) Model 2

(C) Choosing the best model depends on the condition of the company

(D) Both models are good. Unless they both have similar performance, the calibration curves do not play an important role

Correct Answer: B

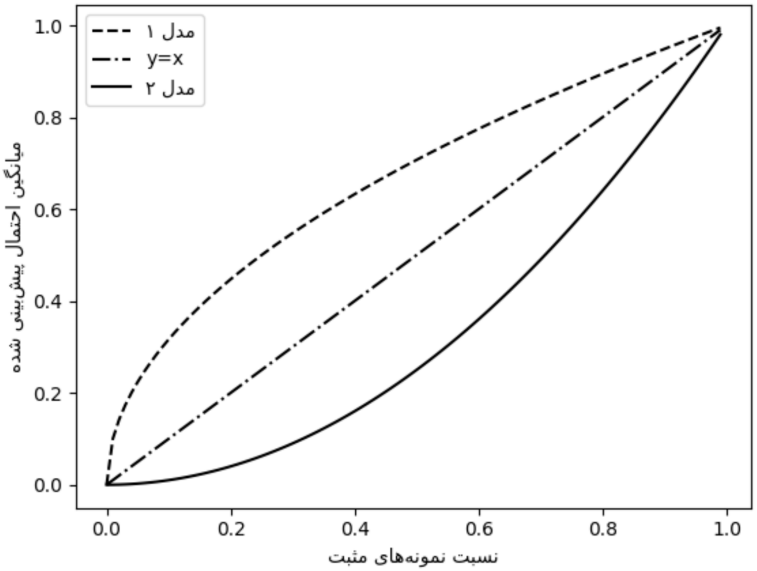


Figure 2: Calibration curves comparing predicted churn probabilities with observed frequencies for different models.

Question 5

Suppose you want to solve a 5-class classification problem. You are told that the different classes are linearly separable from each other, and you are only allowed to use binary classifiers to solve the problem.
Is it possible, for all types of training data with the given properties, to determine the minimum number of binary classifiers required to solve this problem? If yes, what is this number?

(A) Yes, 3

(B) Yes, 4

(C) Yes, 5

(D) No, the minimum number of classifiers may vary depending on the training data.

Solution

Since the five classes are linearly separable from each other, it means that for any pair of classes, there exists a linear boundary that separates the two. Given this property, we can use standard strategies for solving multi-class classification using only binary classifiers.

There are two common approaches:

- **One-vs-Rest (OvR)**: Train k classifiers (one for each class), where each classifier

distinguishes one class from the rest. For $k = 5$ classes, this approach requires exactly 5 classifiers.

- **One-vs-One (OvO)**: Train a classifier for each pair of classes. The number of required classifiers is:

$$\binom{k}{2} = \binom{5}{2} = 10$$

However, the question is asking for the *minimum* number of binary classifiers required in the best-case scenario, assuming that the classes are linearly separable.

This leads us to another interpretation: under ideal conditions (linearly separable classes), what is the fewest number of binary decisions we need to uniquely identify all 5 classes?

This is equivalent to encoding the classes using binary labels. To distinguish between k distinct classes using binary decisions, we need at least $\lceil \log_2 k \rceil$ binary classifiers.

$$\lceil \log_2 5 \rceil = \lceil 2.32 \rceil = 3$$

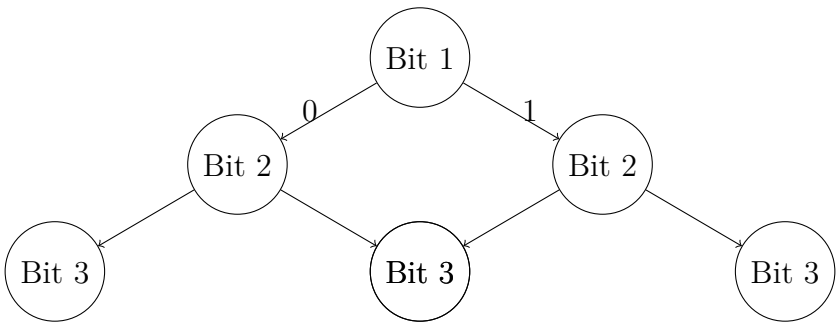
So, in theory, we can construct a decision tree or binary code scheme using only **3 binary classifiers** to uniquely identify all 5 classes.

Example Encoding Scheme

We could assign each class a 3-bit code, such as:

$$\begin{aligned} C_1 &\to 000 \\ C_2 &\to 001 \\ C_3 &\to 010 \\ C_4 &\to 011 \\ C_5 &\to 100 \end{aligned}$$

Then, we train 3 binary classifiers, one for each bit position.



Each classifier splits the data based on one bit in the code.

Conclusion

It is possible to determine the minimum number of binary classifiers required to solve a 5-class linearly separable problem. The minimum number is:

$$\boxed{3}$$

using binary encoding or decision trees under ideal conditions.

Correct Answer: A

Additional Notes

Multi-class classification strategies: OvA and OvO

- **One-vs-All (OvA) / One-vs-Rest:** In this approach, for a k-class problem, k binary classifiers are trained. Each classifier learns to distinguish one class from all the others combined. The final prediction is made by choosing the class whose classifier outputs the highest confidence score.
- **One-vs-One (OvO):** Here, a binary classifier is trained for every pair of classes, leading to $\binom{k}{2}$ classifiers. For prediction, a voting scheme is usually used where each classifier votes for one class, and the class with the most votes wins.

Use in models like SVM: Support Vector Machines (SVMs) are inherently binary classifiers. To apply SVMs to multi-class problems, OvA or OvO strategies are commonly employed. OvO tends to be more computationally expensive but can sometimes yield better performance when classes are well separated pairwise.
Additional notes:

- The choice between OvA and OvO depends on the dataset size, number of classes, and computational resources.
- Another strategy, *Error-Correcting Output Codes (ECOC)*, uses binary encoding schemes to reduce the number of classifiers, similar to the minimum number derived in the main solution.
- The theoretical minimum number of binary classifiers required to uniquely identify k classes is $\lceil \log_2 k \rceil$, but practical implementations often require more due to data complexity.

Question 8

A company is developing a neural network for K-class classification. Because of data confidentiality, it cannot access the input dataset directly. At each step, only the raw output values a_i (logits) of the network are available, before applying the softmax function.
They want to use these values with the cross-entropy loss function. However, computing e^{a_i} for large a_i can cause numerical instability and overflow. Which option below can fix this issue without affecting the training process?

(A) Use $b_i = \dfrac{a_i}{\max_{j\in[1,K]} a_j}$ instead of a_i

(B) No effective step can be taken; inputs must be normalized by shifting them to zero mean and unit variance

(C) Use $b_i = a_i - \max_{j\in[1,K]} a_j$ instead of a_i

(D) No effective step can be taken; the only way is to reduce network layers or use different weight initializations

Correct Answer: C

Explanation:

A: Wrong. Dividing by the maximum logit changes the relative scale of the values and alters the softmax distribution.

B: Wrong. Normalizing input data does not solve the exponential overflow problem in the softmax computation.

C: Correct. Subtracting the maximum logit from all logits does not change the softmax probabilities (since the shift cancels out in numerator and denominator), but it prevents numerical overflow.

D: Wrong. Reducing layers or changing initialization does not address the numerical instability in the softmax step.

Question 11

We want to minimize the function below using the Gradient Descent algorithm:

$$f : \mathbb{R} \to \mathbb{R}, \quad f(x) = x^2 - 6x + 11x - 6$$

The update rule in gradient descent is:

$$x_{t+1} = x_t - \eta(2x_t - 12x_t + 11)$$

Assume the learning rate is $\eta = 1$. Consider the following two initializations:

1. Starting from $x = 4$
2. Starting from $x = 7$

Which statement about the algorithm's behavior is correct?

(A) In the first case, the algorithm converges to a local minimum.

(B) In the second case, the algorithm converges and the values of x_t decrease.

(C) In the first case, the algorithm diverges.

(D) In the second case, the algorithm may converge to a local optimum.

Correct Answer: C

Explanation:

A: Wrong. Starting from $x = 4$ does not lead to convergence to a local minimum.

B: Wrong. From $x = 7$, the updates do not guarantee convergence with $\eta = 1$.

C: Correct. With $x = 4$ and $\eta = 1$, the updates diverge instead of converging.

D: Wrong. In the second case, the updates are unstable and cannot ensure convergence to a local optimum.

Question 12

Which option is **wrong** about Neural Networks?

(A) Starting with the same initial values for weights and biases, if the gradient descent uses all data (Full Batch), the algorithm always converges to the same set of weights and biases.

(B) A single optimization step in stochastic gradient descent (even with a small enough learning rate) may increase the overall loss function, since it is only based on a subset of the training data.

(C) The learning step in stochastic gradient descent is usually smaller than the learning step when using all the training data.

(D) Studying the plot of the cost function over time on the training data can help diagnose overfitting.

Correct Answer: A

Explanation:

A: This statement is wrong. Neural network loss functions are non-convex, so even with full batch gradient descent, convergence may occur at different local minima. The result is not guaranteed to be the same every time.

B: Correct. Since stochastic gradient descent (SGD) updates parameters using a subset of the data, the loss may temporarily increase after an update.

C: Correct. In practice, the learning step (step size) for SGD is usually chosen smaller to avoid instability caused by noisy updates.

D: Correct. By analyzing the cost function curve on training data, one can detect overfitting (e.g., if training loss keeps decreasing while validation loss increases).

Question 13

Mr. Javaheryan is responsible for essay evaluation at the Ministry of Education. He wants to use an intelligent model to distinguish between authentic and non-authentic student essays. For this, he divides the original dataset into training, validation, and test sets. During training, hyperparameters such as the number of hidden layers, number of neurons per layer, and regularization are tuned using the validation set. Although the model performs very well on the training and validation sets, its performance drops significantly on the test set. Which of the following options **cannot** help fix this problem?

(A) Add new data to the validation set, then repeat the training process and tune hyperparameters again.

(B) Use cross-validation (K-fold Cross Validation).

(C) Shuffle the data and repeat the train/validation/test split.

(D) Use more combinations of hyperparameters for training and validation.

Correct Answer: A

Explanation:

A: Wrong choice. Adding new data only to the validation set does not solve the problem, because the model will still not generalize well to unseen test data. The issue is overfitting, not lack of validation data.

B: Correct. Cross-validation uses multiple splits, which provides a better estimate of generalization and helps tune hyperparameters more reliably.

C: Correct. Shuffling and re-splitting the data can ensure that training, validation, and test sets are representative and not biased.

D: Correct. Trying more hyperparameter combinations increases the chance of finding a model that generalizes better.

Question 14

When training a Neural Network, what might happen if the learning rate is set too high?

(A) Faster convergence and reaching the global optimum

(B) Large fluctuations in the weights, which can prevent convergence

(C) Less overfitting because of larger updates

(D) Higher accuracy without further adjustments

Correct Answer: B

If the learning rate is too high, the model may overshoot the optimal solution during training. This causes the weights to change drastically from one update to the next, making it difficult for the model to converge.

For example, instead of gradually approaching the minimum of the loss function, the updates may jump back and forth across it (see Fig. 3). In some cases, the loss might even increase instead of decreasing, leading to training failure.

On the other hand, a properly chosen learning rate allows smooth convergence, while a very low learning rate makes training stable but extremely slow.

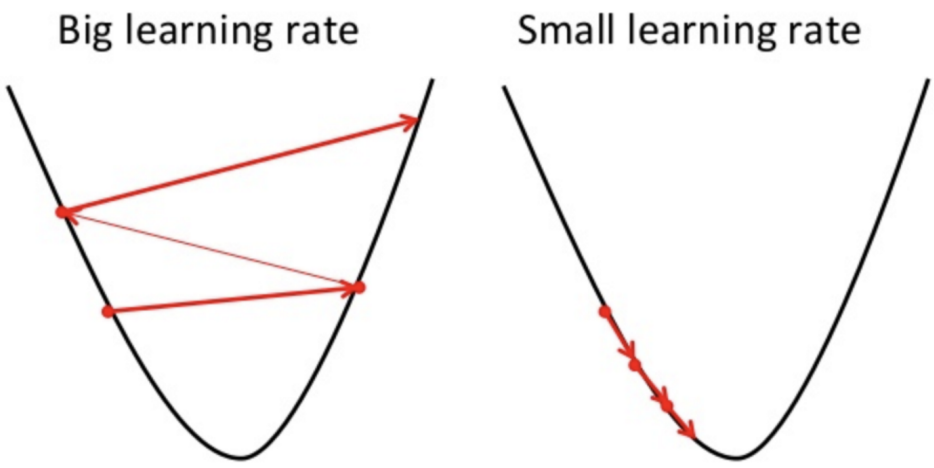


Figure 3: Comparison of two learning rates. A smaller learning rate leads to gradual, stable convergence toward the optimum, while a larger learning rate causes unstable oscillations and prevents convergence.

Question 15

What effect does using a small batch size have on gradient updates and model convergence?

(A) Updates are very smooth and free of noise

(B) The model's performance improves without affecting the speed of convergence

(C) The noise in the updates may help the model avoid local optima but can also make convergence unstable

(D) Memory usage increases and training speed decreases

Correct Answer: C

In general, smaller batch sizes allow faster learning and reduce training time. However, they also introduce noise, which can make convergence unstable and increase the risk of getting stuck in local optima. Larger batch sizes slow down learning but provide more stable updates, which increases the likelihood of reaching the global optimum.