National Olympiad in AI (NOAI) 2026 Preliminary Round Assessment

Welcome to the NOAI 2026 Preliminary Round Assessment.

Date: 20th February 2026, Friday

Duration: 3 hours 5 mins

Total Questions: 300 questions
Start Time: 9:30 am
End Time: 12:35 pm
==========

Assessment Coverage

This assessment evaluates your knowledge across five core AI domains:

1. **Mathematics**
2. **Computing**
3. **AI / Machine Learning / Deep Learning**
4. **Generative AI**
5. **Problem-Solving**

==========

Instructions

Section 1: Personal Information (5 mins)

Please provide accurate information for the following fields:

1. **Full Name** (as per ID)

2. **School Name** (Dropdown options provided)
3. **Teacher In-Charge** (For Polytechnic Students)

4. **Gender** (Male / Female)

5. **Current Education Level and Current Year** (e.g., JC Year 2, Polytechnic Year 3, IP Year 5)

6. **Previous Participation** (Have you participated in NOAI or IOAI before?)

7. **Citizenship** (Singapore Citizen / PR / International Student)

Section 2: Assessment Questions (3 hours)

- Answer all 300 questions to the best of your ability
- Each question is multiple choice with one correct answer
- Questions are distributed across five domains (see coverage above)
- No penalty for wrong answers - attempt every question
- Manage your time: ~36 seconds per question average

Submission

- Review your answers before submitting
- Click **"Submit"** when complete
- ⚠️ **IMPORTANT: Once submitted, you CANNOT change your answers**
- Ensure all questions are answered before clicking Submit

* Indicates required question

1. Email *

Personal Information (1 of 10)

Disclaimer: You consent to the collection, use, and disclosure of your personal data by ***AI Singapore*** for the sole purpose of administering the ***National Olympiad in Artificial Intelligence (NOAI)***. This includes identity verification, communication, and the processing of results and awards.

We are committed to protecting your privacy in accordance with Singapore's Personal Data Protection Act (PDPA). Your information will be stored securely and will not be shared with unauthorised third parties.

2. Full Name as per ID *

Personal Information (2 of 10)

3. School Name * Dropdown

Mark only one oval.

- ANDERSON SERANGOON JUNIOR COLLEGE
- ANGLO-CHINESE JUNIOR COLLEGE
- ANGLO-CHINESE SCHOOL (INDEPENDENT)
- BEATTY SECONDARY SCHOOL
- BEDOK GREEN SECONDARY SCHOOL
- BUKIT BATOK SECONDARY SCHOOL
- CEDAR GIRLS' SECONDARY SCHOOL
- CLEMENTI TOWN SECONDARY SCHOOL
- DUNMAN HIGH SCHOOL
- EUNOIA JUNIOR COLLEGE
- GREENDALE SECONDARY SCHOOL
- HWA CHONG INSTITUTION
- JURONG PIONEER JUNIOR COLLEGE
- METHODIST GIRLS' SCHOOL
- MONTFORT SECONDARY SCHOOL
- NANYANG GIRLS' HIGH SCHOOL
- NATIONAL JUNIOR COLLEGE
- NEW TOWN SECONDARY SCHOOL
- NGEE ANN SECONDARY SCHOOL
- NUS HIGH SCHOOL OF MATHEMATICS AND SCIENCE
- RAFFLES GIRLS' SCHOOL (SECONDARY)
- RAFFLES INSTITUTION
- SCHOOL OF SCIENCE AND TECHNOLOGY, SINGAPORE
- ST. ANDREW'S SCHOOL (SECONDARY)

- [ ] ST. JOSEPH'S INSTITUTION
- [ ] TAMPINES MERIDIAN JUNIOR COLLEGE
- [ ] VICTORIA JUNIOR COLLEGE
- [ ] YISHUN INNOVA JUNIOR COLLEGE
- [ ] ZHONGHUA SECONDARY SCHOOL
- [ ] NGEE ANN POLYTECHNIC
- [ ] NANYANG POLYTECHNIC
- [ ] REPUBLIC POLYTECHNIC
- [ ] SINGAPORE POLYTECHNIC
- [ ] TEMASEK POLYTECHNIC
- [ ] OTHERS

Personal Information (3 of 10)

4. Answer ONLY if you have chosen "OTHERS" in the "School" Section

Personal Information (4 of 10)

5. For Polytechnic Students **ONLY**: Please indicate your Teacher In-Charge

Dropdown

Mark only one oval.

- Mr Jair Zhou (Ngee Ann Poly)
- Dr Brandon Ooi (Nanyang Poly)
- Mr Lim De Yi (Nanyang Poly)
- Mr Frankie Cha (Republic Poly)
- Mr Wu Biao (Singapore Poly)
- Mr Yusuf (Singapore Poly)
- Mr Daryl Sim (Singapore Poly)
- Dr Denver Wong (Temasek Poly)
- OTHERS

Personal Information (5 of 10)

6. Answer ONLY if you have chosen "OTHERS" in the "Teacher In-Charge" Section

Indicate your polytechnic and teacher's name [eg. NYP - Ms Ng]

_____

Personal Information (6 of 10)

7. Gender *

Dropdown

Mark only one oval.

- Male (He/Him)
- Female (She/Her)

Personal Information (7 of 10)

8. Current Education Level * Dropdown

Mark only one oval.

- Secondary
- Integrated Programme
- Junior College
- Polytechnic

Personal Information (8 of 10)

9. Current Year * Dropdown

Mark only one oval.

- Year 1
- Year 2
- Year 3
- Year 4
- Year 5
- Year 6

Personal Information (9 of 10)

You may choose more than one option.

10. Previous Participation *

Tick all that apply.

- [ ] NA
- [ ] IOAI 2024 Pre-Selection Trial
- [ ] IOAI 2024 @ Bulgaria
- [ ] NOAI 2025
- [ ] IOAI 2025 Pre-Selection Trial
- [ ] IOAI 2025 @ Beijing

Personal Information (10 of 10)

11. Citizenship *

Please type either:
- Singapore Citizen
- Singapore PR (+ Country of Origin)
- Foreigner (Nationality)

NOAI 2026 Preliminary Round Assessment

START OF ASSESSMENT QUESTIONS

12. What is the result of multiplying a 3×2 matrix by a 2×4 matrix? *

Mark only one oval.

- ◯ 3×4 matrix
- ◯ 2×2 matrix
- ◯ 3×2 matrix
- ◯ Cannot multiply

13. Given vector v = [3, 4], what is its L2 norm (Euclidean length)? *

Mark only one oval.

- 5
- 7
- 25
- 12

14. Which operation is NOT valid for matrices A (3×3) and B (3×3)? *

Mark only one oval.

- A + B
- A × B
- A ÷ B
- A - B

15. What property does the identity matrix (I) has? *

Mark only one oval.

- All elements are 1
- Diagonal elements are 1, rest are 0
- All elements are 0
- Transpose equals inverse

16. If matrix A is 2×3, what is the shape of A^T (transpose)? *

Mark only one oval.

- 2×3
- 3×2
- 3×3
- 2×2

17. What is the dot product of [1, 2, 3] and [4, 5, 6]? *

Mark only one oval.

- 20
- 28
- 32
- 36

18. Which matrix is symmetric? *

Mark only one oval.

- [[1, 2], [3, 4]]
- [[1, 2], [2, 3]]
- [[1, 0], [1, 0]]
- [[0, 1], [0, 0]]

19. What is the determinant of [[1, 0], [0, 1]]? *

Mark only one oval.

- 0
- 1
- 2
- -1

20. In matrix multiplication AB, which dimension must match? *

Mark only one oval.

- Rows of A = rows of B
- Columns of A = rows of B
- Rows of A = columns of B
- All dimensions must match

21. What is the rank of a zero matrix (all elements = 0)? *

Mark only one oval.

- 0
- 1
- Equal to its rows
- Undefined

22. A square matrix with determinant 0 is called: *

Mark only one oval.

- Orthogonal
- Singular
- Symmetric
- Diagonal

23. In eigenvalue decomposition, what does $Av = \lambda v$ represent? *

Mark only one oval.

- v is an eigenvector with eigenvalue λ
- λ is an eigenvector with eigenvalue v
- A is diagonal
- v must be orthogonal

24. Why is the covariance matrix always symmetric? *

Mark only one oval.

- By definition, Cov(X,Y) = Cov(Y,X)
- It only works for symmetric data
- It's a required property for PCA
- Due to matrix multiplication rules

25. In PCA, what do the eigenvectors of the covariance matrix represent? *

Mark only one oval.

- Principal component directions
- Data variance
- Mean of data
- Number of features

26. If matrix A has eigenvalues [4, 2, 0], what can you conclude? *

Mark only one oval.

- A is invertible
- A is singular (non-invertible)
- A is orthogonal
- A is diagonal

27. What is the trace of a matrix? *

Mark only one oval.

- Sum of all elements
- Sum of diagonal elements
- Product of diagonal elements
- Determinant

28. In Singular Value Decomposition (A = UΣV^T), what does Σ represent? *

Mark only one oval.

- Singular values (diagonal matrix)
- Eigenvectors
- Rotation matrix
- Covariance

29. Why do neural networks prefer orthogonal weight initialisation? *

Mark only one oval.

- Prevents vanishing/exploding gradients
- Reduces memory usage
- Speeds up computation
- Increases model capacity

30. If A is 100×50 matrix, what is the maximum possible rank? *

Mark only one oval.

- 100
- 50
- 150
- 5000

31. In QR decomposition (A = QR), what is Q? *

Mark only one oval.

- Orthogonal matrix
- Upper triangular matrix
- Diagonal matrix
- Symmetric matrix

32. What property makes a matrix positive definite? *

Mark only one oval.

- All elements > 0
- All eigenvalues > 0
- Determinant > 0
- Trace > 0

33. The Frobenius norm of a matrix is: *

Mark only one oval.

- Square root of sum of squared elements
- Maximum absolute value
- Sum of absolute values
- Determinant

34. In the Moore-Penrose pseudoinverse, when is A^+ = (A^T A)^(-1) A^T valid? *

Mark only one oval.

- Always
- When A has full column rank
- When A is square
- When A is symmetric

35. Why is the condition number important for numerical stability? *

Mark only one oval.

- It measures how much output changes with small input changes
- It determines matrix rank
- It calculates determinant
- It counts non-zero elements

36. In spectral clustering, why do we use eigenvectors of the graph Laplacian? *

Mark only one oval.

- They reveal cluster structure in the data
- They speed up computation
- They reduce dimensionality
- They normalise the data

37. What is the derivative of $f(x) = x^2$? *

Mark only one oval.

- x
- 2x
- x^2
- 2

38. What is the derivative of $f(x) = 5x + 3$? *

Mark only one oval.

- 5
- 3
- 5x
- 8

39. The chain rule states that $d/dx[f(g(x))]$ equals: *

Mark only one oval.

- $f'(x) \times g'(x)$
- $f'(g(x)) \times g'(x)$
- $f(x) \times g'(x)$
- $f'(x) + g'(x)$

40. What is the partial derivative $\partial/\partial x$ of $f(x,y) = x^2y + 3x$? *

Mark only one oval.

- 2xy + 3
- $x^2 + 3$
- 2xy
- x^2y

41. What is the gradient of $f(x,y) = x^2 + y^2$? *

Mark only one oval.

- [2x, 2y]
- [x, y]
- $[x^2, y^2]$
- 2x + 2y

42. The derivative of e^x is: *

Mark only one oval.

- e^x
- xe^(x-1)
- e
- 1

43. What is the second derivative of $f(x) = x^3$? *

Mark only one oval.

- $3x^2$
- $6x$
- x^2
- 3

44. In gradient descent, what does the negative gradient point toward? *

Mark only one oval.

- Direction of steepest ascent
- Direction of steepest descent
- Local maximum
- Saddle point

45. What is $\partial/\partial x$ of sigmoid $\sigma(x) = 1/(1+e^{(-x)})$? *

Mark only one oval.

- $\sigma(x)$
- $\sigma(x)(1-\sigma(x))$
- $e^{(-x)}$
- $-\sigma(x)$

46. In backpropagation, if $L = (\hat{y} - y)^2$ and $\hat{y} = wx$, what is $\partial L/\partial w$? *

Mark only one oval.

- $2(\hat{y} - y)$
- $2(\hat{y} - y)x$
- $(\hat{y} - y)x$
- $2w$

47. Why is ReLU's derivative problematic at x=0? *

Mark only one oval.

- It's undefined/discontinuous
- It equals infinity
- It equals zero
- It's always negative

48. The Hessian matrix contains: *

Mark only one oval.

- First-order partial derivatives
- Second-order partial derivatives
- Eigenvalues
- Gradients

49. In optimisation, what does a positive definite Hessian at a critical point indicate? *

Mark only one oval.

- Local minimum
- Local maximum
- Saddle point
- Inflection point

50. What is the Taylor series used for in optimisation? *

Mark only one oval.

- Approximating functions locally
- Computing exact solutions
- Finding global minima
- Normalising inputs

51. In momentum-based optimisation, what does the momentum term do? *

Mark only one oval.

- Accumulates past gradients to smooth updates
- Increases learning rate
- Computes second derivatives
- Regularises weights

52. Why is the learning rate annealed (decreased) during training? *

Mark only one oval.

- To fine-tune near convergence
- To speed up early training
- To prevent underfitting
- To increase batch size

53. What is the gradient of $f(x) = ||x||^2$ (L2 norm squared)? *

Mark only one oval.

- x
- 2x
- ||x||
- x^2

54. In automatic differentiation, what is the computational graph? *

Mark only one oval.

- A graph representing operation dependencies
- A plot of loss over time
- A network architecture diagram
- A visualisation of data

55. Why does the vanishing gradient problem occur with sigmoid activations in deep networks? *

Mark only one oval.

- Sigmoid derivative < 1, causing gradients to exponentially decay
- Sigmoid outputs are too large
- Sigmoid is computationally expensive
- Sigmoid doesn't have a derivative

56. In Newton's method for optimisation, why is it rarely used in deep learning? *

Mark only one oval.

- Requires computing and inverting the Hessian ($O(n^3)$)
- Doesn't find minima
- Only works for convex functions
- Slower than SGD

57. The mean of [2, 4, 6, 8, 10] is: *

Mark only one oval.

- 5
- 6
- 7
- 8

58. The median of [1, 3, 5, 7, 9] is: *

Mark only one oval.

- 3
- 5
- 7
- 9

59. What is the variance of [0, 0, 0, 0]? *

Mark only one oval.

- 0
- 1
- 4
- Undefined

60. A fair 6-sided die is rolled. What is P(rolling ≥ 5)? *

Mark only one oval.

- 1/6
- 1/3
- 1/2
- 2/3

61. In a normal distribution, approximately what % of data is within 1 standard deviation? *

Mark only one oval.

- 50%
- 68%
- 95%
- 99%

62. Two events A and B are independent if: *

Mark only one oval.

- $P(A \mid B) = 0$
- $P(A \cup B) = P(A) + P(B)$
- A and B cannot occur together
- $P(A \cap B) = P(A) \times P(B)$

63. Bayes' theorem is used to compute: *

Mark only one oval.

- $P(A) + P(B)$
- $P(A) \times P(B)$
- $P(A \mid B)$ from $P(B \mid A)$
- Mean and variance

64. The standard deviation is: *

Mark only one oval.

- Square of variance
- Sum of deviations
- Mean absolute deviation
- Square root of variance

65. A dataset has mean=10, std=2. What is the z-score of x=14? *

Mark only one oval.

- 1
- 2
- 4
- 7

66. In a confusion matrix, precision is defined as: *

Mark only one oval.

- TP / (TP + FN)
- TP / (TP + FP)
- (TP + TN) / Total
- TN / (TN + FP)

67. What is recall (sensitivity) in classification? *

Mark only one oval.

- TP / (TP + FP)
- TN / (TN + FN)
- (TP + TN) / Total
- TP / (TP + FN)

68. The F1-score is the: *

Mark only one oval.

- Arithmetic mean of precision and recall
- Product of precision and recall
- Harmonic mean of precision and recall
- Sum of precision and recall

69. In a right-skewed distribution, which is typically largest? *

Mark only one oval.

- Mode > Median > Mean
- Mean > Median > Mode
- Mean = Median = Mode
- Median > Mean > Mode

70. What is the Central Limit Theorem's key insight? *

Mark only one oval.

- All distributions are normal
- Variance decreases with sample size
- Mean equals median
- Sample means are normally distributed for large n

71. In hypothesis testing, what is a Type I error? *

Mark only one oval.

- Accepting false null hypothesis (false negative)
- Correct rejection
- Rejecting true null hypothesis (false positive)
- Correct acceptance

72. Cross-entropy loss for binary classification is: *

Mark only one oval.

- $(y - \hat{y})^2$
- $|y - \hat{y}|$
- $y \times \hat{y}$
- $-[y \log(\hat{y}) + (1-y) \log(1-\hat{y})]$

73. What does a p-value of 0.03 mean? *

Mark only one oval.

- 97% confidence in result
- 3% error rate
- 3% probability of observing data if null hypothesis is true
- Effect size is 0.03

74. The covariance between X and X is: *

Mark only one oval.

- 0
- Variance of X
- 1
- Mean of X

75. In maximum likelihood estimation, what are we maximising? *

Mark only one oval.

- Prior probability of parameters
- Posterior probability
- Prediction accuracy
- Probability of observing the data given parameters

76. Why is the Kullback-Leibler (KL) divergence asymmetric? *

Mark only one oval.

- It only works for discrete distributions
- KL(P||Q) ≠ KL(Q||P) because it's based on expectation over P
- It measures distance incorrectly
- It's a programming error

77. In gradient descent, the learning rate controls: *

Mark only one oval.

- Step size toward minimum
- Number of iterations
- Batch size
- Model capacity

78. What is batch gradient descent? *

Mark only one oval.

- Uses one sample per update
- Uses mini-batches
- Uses all training data per update
- Uses validation data

79. Stochastic Gradient Descent (SGD) uses: *

Mark only one oval.

- All data per update
- Fixed mini-batches
- Validation set
- One random sample per update

80. Mini-batch Gradient Descent (GD) is a compromise between: *

Mark only one oval.

- Stochastic Gradient Descent (SGD) and Adam
- Batch GD and SGD
- Momentum and RMSprop
- L1 and L2 regularisation

81. What problem does momentum help solve in optimisation? *

Mark only one oval.

- Overfitting
- Underfitting
- Class imbalance
- Oscillations and slow convergence

82. Adam optimiser combines: *

Mark only one oval.

- Stochastic Gradient Descent (SGD) and batch GD
- L1 and L2 regularisation
- Momentum and RMSprop
- Dropout and batch norm

83. Training a Transformer model (BERT-base, 110M parameters) from scratch with AdamW optimiser. Two initialisation strategies are tested: *

Strategy A: Constant LR = 1e-4 from step 0
- Step 100: Loss = 8.2 (stable descent)
- Step 500: Loss = 5.1
- Step 10K: Loss = 2.3, converges smoothly

Strategy B: Start with LR = 1e-3 from step 0 (10× higher)
- Step 100: Loss = 12.7 (worse than random initialisation at 9.5)
- Step 300: Loss = NaN (training diverged)
- Training failed

Strategy C: Linear warmup 0 → 1e-3 over 5K steps, then constant
- Step 100: Loss = 9.1 (gradual improvement)
- Step 5K: Loss = 3.2 (warmup complete)
- Step 10K: Loss = 1.8 (better than Strategy A)

Why does Strategy B diverge while Strategy C succeeds with the same peak learning rate?

Mark only one oval.

- Warmup gradually increases batch size alongside learning rate, reducing gradient noise that causes divergence
- Warmup allows the model to memorise training data first with small updates, then generalise with larger updates
- Early training with random weights produces large, unstable gradients; warmup keeps learning rate small while Adam's second moment estimates (v_t) stabilise, preventing explosive parameter updates
- High learning rate without warmup causes gradient clipping to activate, which distorts the loss landscape and prevents convergence

84. Training a language model for 100K steps with three candidate schedules, all using same optimiser (AdamW, β_1=0.9, β_2=0.999): *

Schedule A - Constant: LR = 3e-4 throughout
- Step 20K: Loss = 3.2, learning actively
- Step 50K: Loss = 2.5, still decreasing
- Step 100K: Loss = 2.1, validation perplexity = 22.3

Schedule B - Cosine Decay: LR = 3e-4 → 3e-5 (cosine curve over 100K steps)
- Step 20K: LR ≈ 2.8e-4, Loss = 3.1
- Step 50K: LR ≈ 1.8e-4, Loss = 2.3
- Step 100K: LR = 3e-5, Loss = 1.8, validation perplexity = 19.1

Schedule C - Step Decay: LR = 3e-4 until step 60K, then 3e-5, then 3e-6 at step 80K
- Step 20K: LR = 3e-4, Loss = 3.2
- Step 50K: LR = 3e-4, Loss = 2.5
- Step 100K: LR = 3e-6, Loss = 1.7, validation perplexity = 19.8

Why does Schedule B achieve best validation performance despite Schedule C reaching lower training loss?

Mark only one oval.

- Cosine decay provides smooth transition allowing model to progressively refine parameters; Schedule C's abrupt drops cause loss spikes and may overshoot optimal regions; Schedule A's constant LR prevents convergence to sharp minima
- Lower final learning rate (Schedule C's 3e-6) causes overfitting by converging too precisely to training data; Schedule B's 3e-5 maintains slight exploration preventing memorisation
- Cosine schedule's mathematical properties align with Adam optimiser's momentum, creating synergistic optimisation dynamics
- Step decay creates discontinuities in the gradient flow causing the model to forget previously learned patterns

85. You implement Newton's method for a small 3-layer network (10K parameters). * Training is 100× slower than Adam despite better convergence. Profiling shows 95% of time is spent in one operation. What could be this one operation?

Mark only one oval.

- Matrix inversion of the 10K×10K Hessian matrix
- Forward pass through the network
- Computing gradients via backpropagation
- Loading data from disk

86. In convex optimisation, what guarantees does gradient descent have? *

Mark only one oval.

- Converges to global minimum with appropriate learning rate
- Finds solution in polynomial time
- Always faster than other methods
- Requires no hyperparameters

87. What is the output of print(type([1, 2, 3]))? *

Mark only one oval.

- <class 'array'>
- <class 'tuple'>
- <class 'list'>
- [1, 2, 3]

88. Which is mutable in Python? *

Mark only one oval.

- Tuple
- String
- List
- Integer

89. What does len("Hello") return? *

Mark only one oval.

- 4
- 5
- 6
- Error

90. How do you create an empty dictionary? *

Mark only one oval.

- {}
- []
- ()
- set()

91. What is the result of [1, 2] + [3, 4]? *

Mark only one oval.

- [4, 6]
- [[1, 2], [3, 4]]
- Error
- [1, 2, 3, 4]

92. Which loop iterates over indices 0 to 4? *

Mark only one oval.

- for i in range(4):
- for i in range(1, 5):
- for i in range(5):
- for i in range(0, 4):

93. What is list comprehension [x**2 for x in range(3)]? *

Mark only one oval.

- [1, 4, 9]
- [0, 1, 4]
- [0, 1, 2]
- [2, 4, 6]

94. What happens when you modify a list passed to a function? *

Mark only one oval.

- Original list is unchanged
- Creates a copy automatically
- Original list is modified (passed by reference)
- Raises an error

95. What is the difference between is and ==? *

Mark only one oval.

- They are identical
- is checks identity (same object), == checks equality (same value)
- is is faster
- == checks type

96. What does *args allow in a function? *

Mark only one oval.

- Variable number of keyword arguments
- Default values
- Type hints
- Variable number of positional arguments

97. Is this correct for exception handling? *

Python Code:

```
try:
   # code
except ValueError:
   # handle
```

Mark only one oval.

- This is correct
- Should be `catch` not `except`
- Missing `finally`
- `try` should be followed by `else`

98. What is a generator in Python? *

Mark only one oval.

- Class for creating objects
- Loop structure
- Function using yield to produce values lazily
- Random number generator

99. What does zip([1,2], ['a','b']) produce? *

Mark only one oval.

- Iterator of pairs: (1,'a'), (2,'b')
- [1, 2, 'a', 'b']
- [[1, 'a'], [2, 'b']]
- Error

100. What is the GIL (Global Interpreter Lock) in Python? *

Mark only one oval.

- Locks variables during assignment
- Manages GPU memory
- Protects against race conditions
- Prevents true multi-threading for CPU-bound tasks

101. How does Python's garbage collection work? *

Mark only one oval.

- Manual memory management
- Mark-and-sweep only
- Reference counting + cyclic garbage collector
- No garbage collection

102. How do you create a NumPy array from a list? *

Mark only one oval.

- np.list([1, 2, 3])
- array([1, 2, 3])
- np.array([1, 2, 3])
- numpy([1, 2, 3])

103. What is the shape of np.zeros((3, 4))? *

Mark only one oval.

- (4, 3)
- 12
- (3, 4, 0)
- (3, 4)

104. Element-wise multiplication of arrays a * b requires: *

Mark only one oval.

- One-dimensional arrays
- Square arrays
- Same number of elements
- Same shape (or broadcastable)

105. What does np.arange(0, 10, 2) produce? *

Mark only one oval.

- [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
- [2, 4, 6, 8, 10]
- [0, 2, 4, 6, 8]
- [0, 2, 4, 6, 8, 10]

106. How do you access the first row of a 2D array arr? *

Mark only one oval.

- arr[0] or arr[0, :]
- arr[:, 0]
- arr[0, 0]
- arr.row(0)

107. What is broadcasting in NumPy? *

Mark only one oval.

- Sending data to GPU
- Automatic shape adjustment for operations
- Parallel computation
- Data shuffling

108. Which is the correct NumPy syntax to compute the mean of array `arr`? *

Mark only one oval.

- np.mean(arr)
- arr.average()
- mean(arr)
- np.avg(arr)

109. What is vectorisation in NumPy? *

Mark only one oval.

- Converting scalars to vectors
- GPU acceleration
- Using array operations instead of loops
- Data type conversion

110. You have a Pandas DataFrame `df` with a column named 'total price' (contains a space). Which is the ONLY valid way to select this column? *

Mark only one oval.

- df.total price
- df('total price')
- df.loc['total price']
- df['total price']

111. What does df.groupby('category').mean() do? *

Mark only one oval.

- Groups by mean values
- Computes mean for each category group
- Creates new category column
- Sorts by category

112. How do you handle missing values in Pandas? *

Mark only one oval.

- Manually replace with loops
- Use try-except
- Cannot handle missing data
- df.fillna(), df.dropna(), or df.interpolate()

113. What is the difference between np.dot() and * for arrays? *

Mark only one oval.

- They are identical
- * is matrix multiplication
- np.dot() is matrix multiplication, * is element-wise
- np.dot() is element-wise

114. How do you concatenate two arrays along axis=0? *

Mark only one oval.

- np.concatenate([arr1, arr2], axis=0)
- arr1 + arr2
- np.stack([arr1, arr2])
- arr1.concat(arr2)

115. Why is NumPy faster than Python lists for numerical operations? *

Mark only one oval.

- Contiguous memory, C implementations, vectorisation
- Uses GPU automatically
- Compiled to machine code
- Parallel processing only

116. What is stride in NumPy arrays? *

Mark only one oval.

- Step size in arange()
- Bytes to jump to next element in each dimension
- Array shape
- Data type size

117. How do you create a PyTorch tensor from a list? *

Mark only one oval.

- torch.array([1, 2, 3])
- Tensor([1, 2, 3])
- torch.tensor([1, 2, 3])
- tensor([1, 2, 3])

118. What is the purpose of requires_grad=True? *

Mark only one oval.

- Freezes the tensor
- Moves tensor to GPU
- Sets random seed
- Enables gradient computation for the tensor

119. How do you move a tensor to GPU? *

Mark only one oval.

- tensor.gpu()
- tensor.to('cuda') or tensor.cuda()
- tensor.device('cuda')
- Automatically handled

120. What does loss.backward() do? *

Mark only one oval.

- Reverses the forward pass
- Updates weights
- Computes gradients via backpropagation
- Resets the model

121. Why do you need optimiser.zero_grad()? *

Mark only one oval.

- Initialises optimiser
- Clears loss
- PyTorch accumulates gradients; must reset
- Resets model weights

122. What is nn.Module in PyTorch? *

Mark only one oval.

- Base class for all neural network modules
- Activation function
- Loss function
- Optimiser

123. How do you define a forward pass in PyTorch? *

Mark only one oval.

- Call model.backward()
- Implement forward() method in nn.Module
- Use predict() method
- Automatic from __init__

124. What is the purpose of model.train() and model.eval()? *

Mark only one oval.

- Start/stop training
- Load/save models
- Toggle GPU usage
- Switch between training/evaluation modes (affects dropout, batch norm)

125. How do you create a simple linear layer? *

Mark only one oval.

- nn.Layer(input_size, output_size)
- nn.Linear(input_size, output_size)
- torch.Linear(input_size, output_size)
- Linear(input_size, output_size)

126. What happens if you call loss.backward() twice without zero_grad()? *

Mark only one oval.

- Gradients accumulate (add up)
- Error is raised
- Second call overwrites first
- Only second call's gradients are kept

127. What is torch.no_grad() used for? *

Mark only one oval.

- Resets gradients
- Stops training
- Disables gradient computation (saves memory during inference)
- Freezes model

128. How do you access model parameters? *

Mark only one oval.

- model.weights()
- model.get_params()
- model.parameters() or model.named_parameters()
- model.layers()

129. What is the difference between torch.Tensor and torch.tensor? *

Mark only one oval.

- They are identical
- tensor infers dtype, Tensor uses default float32
- Tensor is deprecated
- tensor cannot compute gradients

130. How do you implement a custom loss function? *

Mark only one oval.

- Use built-in losses only
- Modify existing loss
- Subclass nn.Module and define forward()
- Write regular Python function (no gradients)

131. What is torch.nn.Sequential used for? *

Mark only one oval.

- Process sequences (RNN)
- Parallel processing
- Stack layers in order for simple models
- Data augmentation

132. How do you freeze a layer's parameters? *

Mark only one oval.

- Remove layer from model
- Use model.freeze()
- Set param.requires_grad = False
- Set learning rate to 0

133. What is a DataLoader in PyTorch? *

Mark only one oval.

- Loads pretrained models
- Reads image files
- Stores training data
- Iterates over dataset in batches with shuffling

134. How do you save a trained model? *

Mark only one oval.

- model.save('path')
- torch.save(model.state_dict(), 'path')
- save(model, 'path')
- model.export('path')

135. What is mixed precision training? *

Mark only one oval.

- Training on multiple GPUs
- Combining different optimisers
- Using FP16 + FP32 to speed up training with less memory
- Using multiple loss functions

136. Why use torch.jit.script() or torch.jit.trace()? *

Mark only one oval.

- Optimise model for deployment via TorchScript
- Debug model
- Visualise architecture
- Convert to TensorFlow

137. Your code raises IndexError: list index out of range. What's the likely cause? *

Mark only one oval.

- Wrong data type
- Memory overflow
- GPU error
- Accessing index ≥ list length

138. What does print() debugging help with? *

Mark only one oval.

- Inspecting variable values during execution
- Fixing syntax errors
- Optimising speed
- Reducing memory

139. A RuntimeError: CUDA out of memory suggests *

Mark only one oval.

- Batch size too large or model too big for GPU
- Code syntax error
- Wrong optimiser
- Missing data

140. How do you check if a tensor is on GPU? *

Mark only one oval.

- tensor.device or tensor.is_cuda
- tensor.gpu()
- tensor.location
- torch.device(tensor)

141. Your training loss is NaN. What should you check first? *

Mark only one oval.

- Model architecture only
- Learning rate (too high), loss function (division by zero), or exploding gradients
- Batch size
- Number of epochs

142. How do you profile PyTorch code to find bottlenecks? *

Mark only one oval.

- Print execution time for each line
- Guess which parts are slow
- Use torch.profiler or cProfile
- Use time.sleep()

143. What is gradient clipping used for? *

Mark only one oval.

- Speeding up training
- Reducing model size
- Preventing exploding gradients
- Improving accuracy

144. Why might training be slow on GPU compared to CPU? *

Mark only one oval.

- GPU is always slower
- Wrong optimiser
- Too many parameters
- Data transfer overhead, small batch size, or simple model

145. What is the difference between .item() and .detach() in PyTorch? *

Mark only one oval.

- They are identical
- .item() converts to Python scalar, .detach() removes from comp graph
- .item() moves to CPU
- .detach() deletes tensor

146. How do you detect memory leaks in PyTorch? *

Mark only one oval.

- Check disk space
- Count parameters
- Run on CPU
- Monitor torch.cuda.memory_allocated() or use profilers

147. Supervised learning requires: *

Mark only one oval.

- Unlabeled data only
- Reinforcement signals
- No data
- Labeled training data

148. Which is an unsupervised learning algorithm? *

Mark only one oval.

- K-means clustering
- Logistic regression
- Decision tree
- Support Vector Machine

149. In linear regression, what are we predicting? *

Mark only one oval.

- Class label
- Cluster assignment
- Continuous value
- Probability distribution

150. Logistic regression is used for: *

Mark only one oval.

- Continuous regression
- Clustering
- Dimensionality reduction
- Binary classification

151. In KNN (K-Nearest Neighbors), predictions are based on: *

Mark only one oval.

- K clusters
- K features
- K decision trees
- K nearest training examples

152. A decision tree splits data based on: *

Mark only one oval.

- Random selection
- Alphabetical order
- Feature values that best separate classes
- Feature names

153. What is overfitting? *

Mark only one oval.

- Model is too simple
- Training loss is high
- Model trains too fast
- Model memorises training data, poor generalisation

154. What is the purpose of a validation set? *

Mark only one oval.

- Train the model
- Tune hyperparameters without using test data
- Final performance evaluation
- Data augmentation

155. Cross-validation is used to: *

Mark only one oval.

- Clean data
- Select features
- Normalise inputs
- Estimate model performance on unseen data

156. Principal Component Analysis (PCA) is used for: *

Mark only one oval.

- Classification
- Clustering
- Dimensionality reduction
- Regression

157. In K-means, how do you choose K? *

Mark only one oval.

- Always use K=2
- K = number of features
- Domain knowledge or elbow method
- K = number of samples

158. What is the bias-variance tradeoff? *

Mark only one oval.

- Bias and variance always increase together
- Bias equals variance
- Simpler models have high bias, complex models have high variance
- No relationship between them

159. L1 regularisation (Lasso) tends to: *

Mark only one oval.

- Produce dense models
- Have no effect
- Only work for classification
- Produce sparse models (many zero weights)

160. L2 regularisation (Ridge) penalises: *

Mark only one oval.

- Sum of absolute weights
- Number of parameters
- Sum of squared weights
- Training loss only

161. In SVM (Support Vector Machine), what is the kernel trick? *

Mark only one oval.

- Reduces dimensionality
- Speeds up training
- Implicitly maps data to higher dimensions
- Selects features

162. Random Forest improves on decision trees by: *

Mark only one oval.

- Using one very deep tree
- Removing all splits
- Using KNN instead
- Ensembling multiple trees with bagging

163. Gradient Boosting builds trees: *

Mark only one oval.

- In parallel independently
- All at once
- Randomly
- Sequentially, each correcting previous errors

164. What is the curse of dimensionality? *

Mark only one oval.

- More features always improve accuracy
- Computations become faster
- High dimensions cause data sparsity and overfitting
- Models become simpler

165. In ensemble methods, bagging reduces: *

Mark only one oval.

- Bias
- Training time
- Model complexity
- Variance

166. Boosting reduces: *

Mark only one oval.

- Variance only
- Both equally
- Neither
- Bias

167. What is feature scaling and why is it important? *

Mark only one oval.

- Removes features
- Normalises features so they have similar ranges (helps many algorithms)
- Creates new features
- Visualises data

168. The Receiver Operating Characteristic (ROC) curve plots: *

Mark only one oval.

- Precision vs Recall
- Accuracy vs Loss
- Training vs Validation error
- True Positive Rate vs False Positive Rate

169. Why might Naive Bayes work well despite its "naive" assumption? *

Mark only one oval.

- It's always accurate
- It uses deep learning
- Feature independence assumption simplifies computation; often works in practice
- It requires no training

170. In anomaly detection, why is one-class Support Vector Machine (SVM) useful? *

Mark only one oval.

- Requires both normal and anomaly examples
- Only works for binary classification
- Fastest algorithm
- Learns from normal data only, detects outliers

171. What is the difference between hard and soft margin Support Vector Machine (SVM)? *

Mark only one oval.

- Soft margin allows some misclassification (C parameter controls tradeoff)
- Hard margin is always better
- Soft margin doesn't use kernel
- No practical difference

172. A perceptron is: *

Mark only one oval.

- Deep neural network
- Convolutional layer
- Recurrent unit
- Single-layer linear classifier

173. Activation functions introduce: *

Mark only one oval.

- Regularisation
- Dropout
- Batch normalisation
- Non-linearity to the network

174. ReLU activation function is: *

Mark only one oval.

- max(0, x)
- 1 / (1 + e^(-x))
- tanh(x)
- x^2

175. Sigmoid function outputs values in range: *

Mark only one oval.

- (-1, 1)
- (-∞, ∞)
- (0, 1)
- [0, ∞)

176. Backpropagation is used to: *

Mark only one oval.

- Initialise weights
- Make predictions
- Compute gradients via chain rule
- Evaluate test accuracy

177. A fully connected layer connects: *

Mark only one oval.

- Only adjacent neurons
- Randomly selected neurons
- First and last neurons only
- Every input to every output

178. What is an epoch in training? *

Mark only one oval.

- One forward pass
- One batch
- One weight update
- One complete pass through training dataset

179. Batch size determines: *

Mark only one oval.

- Number of epochs
- Model architecture
- Learning rate
- Number of samples per gradient update

180. Dropout randomly: *

Mark only one oval.

- Removes layers
- Changes weights
- Shuffles data
- Sets neurons to zero during training

181. Batch normalisation: *

Mark only one oval.

- Normalises weights
- Normalises loss
- Normalises predictions
- Normalises layer inputs to stabilise training

182. Mean Squared Error (MSE) loss is used for: *

Mark only one oval.

- Binary classification
- Multi-class classification
- Clustering
- Regression problems

183. Why is ReLU preferred over sigmoid in hidden layers? *

Mark only one oval.

- Always outputs between 0-1
- More complex
- Requires less memory
- Avoids vanishing gradients, faster computation

184. What is the vanishing gradient problem? *

Mark only one oval.

- Gradients become too large
- Gradients are zero everywhere
- No gradients are computed
- Gradients become very small in deep networks, slowing learning

185. Residual connections (skip connections) help: *

Mark only one oval.

- Reduce model size
- Increase training speed only
- Train deeper networks by enabling gradient flow
- Remove need for activation functions

186. Why is weight initialisation important? *

Mark only one oval.

- Not important, random is fine
- Only affects speed
- Prevents symmetry breaking and improves convergence
- Only for small networks

187. Xavier/Glorot initialisation is designed for: *

Mark only one oval.

- ReLU activations
- No activations
- Tanh and sigmoid activations
- Output layers only

188. You're training a 10-layer ReLU network. After initialising all weights to small random values from N(0, 0.01^2), the network fails to learn (loss stays constant). Switching to He initialization fixes the problem. Why? *

Mark only one oval.

- Small weights make ReLU output all zeros, blocking gradient flow
- Small weights cause numerical underflow in float32 precision
- ReLU requires positive weights to function
- Small weight initialisation causes activations to shrink exponentially through layers, leading to vanishing gradients

189. Early stopping prevents: *

Mark only one oval.

- Underfitting
- Training from starting
- Gradient computation
- Overfitting by stopping when validation error increases

190. Learning rate scheduling: *

Mark only one oval.

- Sets initial weights
- Chooses optimiser
- Selects batch size
- Adjusts learning rate during training (often decreasing)

191. What is the dying ReLU problem? *

Mark only one oval.

- ReLU is too slow
- ReLU causes overfitting
- ReLU has no gradient
- Neurons output zero for all inputs (dead neurons)

192. Leaky ReLU addresses dying ReLU by: *

Mark only one oval.

- Using exponential function
- Adding dropout
- Increasing learning rate
- Allowing small negative slope for x < 0

193. Softmax activation in output layer is used for: *

Mark only one oval.

- Regression
- Binary classification only
- Hidden layers
- Multi-class classification (converts logits to probabilities)

194. Cross-entropy loss for classification measures: *

Mark only one oval.

- Squared error
- Absolute error
- Difference between predicted and true probability distributions
- Cosine similarity

195. Why do batch norm and dropout sometimes conflict? *

Mark only one oval.

- They can't be used together
- Both are redundant
- BatchNorm assumes consistent statistics; dropout adds noise
- Dropout disables batch norm

196. What is layer normalisation and when is it preferred over batch norm? *

Mark only one oval.

- Same as batch norm
- Only for CNNs
- Normalises across features per sample; better for small batches/RNNs
- Slower than batch norm

197. CNNs (Convolutional Neural Networks) are primarily used for: *

Mark only one oval.

- Text processing
- Time series
- Image/visual data processing
- Tabular data

198. A convolutional layer applies: *

Mark only one oval.

- Fully connected operations
- Dropout
- Filters/kernels to detect features
- Softmax

199. Pooling layers: *

Mark only one oval.

- Increase dimensions
- Add parameters
- Reduce spatial dimensions (downsampling)
- Normalise values

200. Max pooling selects: *

Mark only one oval.

- Average value
- Minimum value
- Random value
- Maximum value in each window

201. Stride in convolution controls: *

Mark only one oval.

- Number of filters
- Filter size
- Step size of filter movement
- Learning rate

202. Padding in CNNs: *

Mark only one oval.

- Removes edges
- Increases channels
- Adds borders to preserve spatial dimensions
- Normalises inputs

203. Image classification aims to: *

Mark only one oval.

- Label each pixel
- Detect object locations
- Assign one label to entire image
- Generate new images

204. What are feature maps in CNNs? *

Mark only one oval.

- Input images
- Weight matrices
- Loss values
- Outputs of convolutional layers

205. Transfer learning in vision involves: *

Mark only one oval.

- Training from scratch
- Using pretrained model weights as starting point
- Transferring data
- Changing architecture

206. Why are deeper CNNs generally better for complex vision tasks? *

Mark only one oval.

- Always faster
- Learn hierarchical features (edges → textures → objects)
- Fewer parameters
- Simpler to train

207. Object detection differs from classification by: *

Mark only one oval.

- Using CNNs
- Detecting multiple objects and their locations
- Outputting single label
- Not using bounding boxes

208. YOLO (You Only Look Once) is: *

Mark only one oval.

- Classification model
- Segmentation model
- Single-stage real-time object detector
- Two-stage detector

209. R-CNN (Region-based CNN) first: *

Mark only one oval.

- Classifies entire image
- Segments pixels
- Proposes regions, then classifies each
- Generates images

210. Semantic segmentation assigns: *

Mark only one oval.

- One label to entire image
- Bounding boxes
- Class label to each pixel
- Object instances

211. Instance segmentation: *

Mark only one oval.

- Same as semantic segmentation
- Only detects objects
- Classifies images
- Distinguishes individual object instances at pixel level

212. Data augmentation in vision includes: *

Mark only one oval.

- Adding more layers
- Changing loss function
- Increasing batch size
- Rotation, flipping, cropping, color jittering

213. Why is ImageNet pretraining so effective? *

Mark only one oval.

- Large diverse dataset teaches general visual features
- It's the only available dataset
- Always gives best results
- Only works for classification

214. What is the receptive field in CNNs? *

Mark only one oval.

- Region of input that affects a neuron's activation
- Size of filter
- Number of layers
- Output dimension

215. Why do modern CNNs use 3×3 convolutions instead of larger filters? *

Mark only one oval.

- 3×3 is faster only
- Required by hardware
- Stacking 3×3 achieves same receptive field with fewer parameters
- No particular reason

216. What innovation did ResNet introduce? *

Mark only one oval.

- Larger filters
- More pooling
- Skip connections enabling training of very deep networks (100+ layers)
- Batch normalisation

217. Word embeddings represent: *

Mark only one oval.

- Words as one-hot vectors
- Words as strings
- Words as images
- Words as dense vectors

218. Word2Vec learns embeddings by: *

Mark only one oval.

- Random initialisation
- Rule-based mapping
- Predicting context words or target from context
- Supervised classification

219. RNNs (Recurrent Neural Networks) are designed for: *

Mark only one oval.

- Images
- Sequential data like text
- Tabular data
- Audio only

220. LSTMs solve what RNN problem? *

Mark only one oval.

- Slow training
- Large memory
- Long-term dependency/vanishing gradient
- No problem

221. Attention mechanism allows models to: *

Mark only one oval.

- Speed up training
- Focus on relevant parts of input
- Reduce parameters
- Remove recurrence

222. Transformers primarily use: *

Mark only one oval.

- Recurrence (RNN)
- Convolution (CNN)
- Self-attention mechanisms
- Fully connected layers only

223. BERT (Bidirectional Encoder Representations from Transformers) is: *

Mark only one oval.

- Generative model
- CNN-based
- Pretrained language model for understanding
- RNN-based

224. What is the key difference between BERT and GPT? *

Mark only one oval.

- They are identical
- BERT is for vision
- BERT = bidirectional encoder, GPT = autoregressive decoder
- GPT is older

225. Positional encoding in transformers: *

Mark only one oval.

- Removes position
- Only for RNNs
- Adds position information since attention has no inherent order
- Speeds up computation

226. Masked language modeling (BERT's training): *

Mark only one oval.

- Predicts next token
- Generates text
- Classifies sentiment
- Predicts masked tokens from context

227. Tokenisation breaks text into: *

Mark only one oval.

- Characters only
- Sentences only
- Subword units or words for model input
- Random pieces

228. Fine-tuning a pretrained LLM means: *

Mark only one oval.

- Training from scratch
- Further training on task-specific data
- Testing only
- Freezing all weights

229. Why is the CLS token used in BERT? *

Mark only one oval.

- Marks end of sentence
- Indicates error
- Represents pooled sentence/sequence representation for classification
- Random placeholder

230. What is the computational complexity of self-attention in transformers? *

Mark only one oval.

- O(n) linear
- O(log n)
- $O(n^2)$ where n = sequence length (quadratic)
- O(1) constant

231. Why do large language models use causal masking during training? *

Mark only one oval.

- Speeds up training
- No particular reason
- Reduces memory
- Prevents attending to future tokens (autoregressive property)

232. In random forests, what does "out-of-bag (OOB) error" estimate? *

Mark only one oval.

- Error on training data
- Generalisation error using samples not in bootstrap for each tree
- Error on validation set
- Number of trees needed

233. What is the purpose of gradient clipping? *

Mark only one oval.

- Speeds up training
- Reduces model size
- Prevents exploding gradients by limiting gradient magnitude
- Improves accuracy

234. In U-Net architecture, what is the purpose of skip connections? *

Mark only one oval.

- Speed up training
- Reduce parameters
- Preserve spatial information from encoder for precise localisation in decoder
- Add non-linearity

235. What is the main advantage of subword tokenisation (e.g., BPE, WordPiece)? *

Mark only one oval.

- Faster processing
- Simpler implementation
- Works only for English
- Handles unknown words and reduces vocabulary size while capturing morphology

236. What distinguishes generative models from discriminative models? *

Mark only one oval.

- Generative models are always better
- Generative models learn P(X,Y) and can generate new data; discriminative learn P(Y|X) for classification
- Discriminative models cannot classify
- No practical difference

237. GPT stands for: *

Mark only one oval.

- General Purpose Text
- Gradient Propagation Technique
- Generative Pre-trained Transformer
- Graph Processing Tool

238. Large Language Models (LLMs) are trained on: *

Mark only one oval.

- Small labeled datasets
- Images
- Massive text corpora from internet
- Audio files

239. Autoregressive generation means: *

Mark only one oval.

- Predicting next token based on previous tokens
- Generating all tokens simultaneously
- Randomly generating text
- Copying input

240. Temperature in LLM sampling controls: *

Mark only one oval.

- Randomness/diversity of outputs (higher = more random)
- Model speed
- Model size
- Training rate

241. Top-k sampling: *

Mark only one oval.

- Samples from all tokens
- Always picks most likely
- Samples from k most likely next tokens
- Random sampling

242. Prompt engineering is: *

Mark only one oval.

- Model architecture design
- Designing input text to guide LLM outputs
- Training method
- Data preprocessing

243. Zero-shot learning means: *

Mark only one oval.

- No training at all
- Model performs task with one task-specific training example
- Model performs task without task-specific training examples
- Model performs task with many task-specific training examples

244. Few-shot learning provides: *

Mark only one oval.

- No examples
- Full dataset
- Only instructions
- Small number of examples in prompt

245. Fine-tuning vs prompting: fine-tuning involves: *

Mark only one oval.

- Only changing prompts
- No training
- Updating model weights on task data
- Freezing all layers

246. LoRA (Low-Rank Adaptation) enables: *

Mark only one oval.

- Faster inference
- Efficient fine-tuning by training low-rank weight adapters
- Larger models
- Data augmentation

247. What is the main challenge with scaling transformers to long contexts? *

Mark only one oval.

- Memory for weights
- Quadratic attention complexity limits sequence length
- Training data availability
- Vocabulary size

248. Retrieval-Augmented Generation (RAG) enhances LLMs by: *

Mark only one oval.

- Increasing model size
- Training longer
- Retrieving relevant documents to provide context
- Using multiple GPUs

249. Diffusion models generate images by: *

Mark only one oval.

- One-step generation
- Classification
- Gradually denoising from random noise
- Image compression

250. Stable Diffusion is a: *

Mark only one oval.

- Classification model
- Object detection model
- Text-to-image generation model
- Video model only

251. Latent diffusion operates in: *

Mark only one oval.

- Compressed latent space
- Pixel space
- Text space
- Label space

252. The forward diffusion process: *

Mark only one oval.

- Removes noise
- Adds noise gradually to image until pure noise
- Generates images
- Compresses images

253. The reverse diffusion process learns to: *

Mark only one oval.

- Add noise
- Classify images
- Denoise step-by-step to generate images
- Compress images

254. Classifier-free guidance in diffusion: *

Mark only one oval.

- Removes need for classifier
- Speeds up sampling
- Improves generation quality by steering toward text condition
- Reduces model size

255. Why are diffusion models slower than Generative Adversarial Network (GAN) at inference? *

Mark only one oval.

- Larger model size
- More parameters
- Require many iterative denoising steps
- GPU limitations

256. DDPM (Denoising Diffusion Probabilistic Model) defines diffusion as: *

Mark only one oval.

- Single-step generation
- Deterministic process
- Markov Chain with learnt reverse process
- Classification task

257. Multimodal models process: *

Mark only one oval.

- One data type only
- Only text
- Multiple data types
- Only images

258. CLIP (Contrastive Language-Image Pre-training) learns: *

Mark only one oval.

- Image classification only
- Text generation
- Aligned image and text embeddings
- Audio processing

259. Contrastive learning in CLIP: *

Mark only one oval.

- Standard classification
- Matches positive pairs, separates negative pairs
- Regression
- Clustering

260. Vision Transformer (ViT) treats images as: *

Mark only one oval.

- Convolutional features
- Single vector
- Sequence of patches
- Individual pixels

261. Zero-shot image classification with CLIP: *

Mark only one oval.

- Classify without training on specific classes
- Requires labeled data
- Only works for ImageNet
- Needs fine-tuning

262. Why is CLIP useful for downstream tasks? *

Mark only one oval.

- It's the fastest model
- Pretrained representations transfer to various vision-language tasks
- Smallest model size
- Only for generation

263. Flamingo model combines: *

Mark only one oval.

- Text only
- Audio and video
- Generative Adversarial Networks and Diffusion
- Vision and language for few-shot multimodal learning

264. GPT-4 Vision (GPT-4V) can: *

Mark only one oval.

- Process images and text together
- Only process text
- Only generate images
- Only classify images

265. Cross-attention in multimodal models: *

Mark only one oval.

- Speeds up training
- Only within same modality
- Allows one modality to attend to another
- Removes attention

266. Why is vision-language pretraining challenging compared to language-only? *

Mark only one oval.

- Less data available
- Simpler models needed
- No challenges
- Aligning different modalities, scaling data collection

267. You're building a model to predict house prices using a dataset with 20 features (size, bedrooms, location, age, etc.) and 5,000 samples. Which algorithm choice is most appropriate and why? *

Mark only one oval.

- Deep neural network with 5 hidden layers - always performs better than classical ML on any dataset
- Gradient boosted decision trees (XGBoost/LightGBM) - tabular data with moderate feature count and interpretability needs favor tree-based methods
- Linear regression - simplest model is always best
- K-Nearest Neighbors - works well for all tabular data

268. For real-time object detection, which is better? *

Mark only one oval.

- R-CNN
- ResNet
- LSTM
- YOLO

269. A medical diagnosis dataset contains 50 patient samples with 100 biomarker measurements per patient (blood protein levels, gene expressions, etc.). A data scientist trains a logistic regression model achieving: *
- Training accuracy: 98% (49/50 correct)
- Validation accuracy: 62% (31/50 correct)
- Most feature weights are non-zero

What problem is the model experiencing and what is its root cause?

Mark only one oval.

- Overfitting due to high dimensionality (p=100) relative to sample size (n=50) - the model has more parameters (100 weights) than effective training samples, allowing it to memorise training data rather than learn generalisable patterns
- Underfitting because 50 samples are insufficient to train any model - need minimum 1000 samples for machine learning to work
- Class imbalance in the dataset causing the model to predict only the majority class
- Feature scaling issues - biomarkers measured in different units (ng/mL, copies, etc.) preventing proper convergence

270. A customer churn prediction dataset has 10,000 samples with 20 features. After data collection, 15% of values are missing (randomly distributed across samples and features). Four ML practitioners propose different approaches: *
- Approach A: "Use decision trees - they naturally handle missing values through surrogate splits during training."
- Approach B: "Use K-NN with complete-case analysis - just remove any sample with missing values before training."
- Approach C: "Use linear regression with mean imputation - replace missing values with feature means."
- Approach D: "Use SVM after forward-filling missing values with the previous non-missing value in the dataset."

Which practitioner's reasoning is most accurate?

Mark only one oval.

- Practitioner A is correct - tree algorithms (CART, XGBoost, LightGBM) can handle missing values natively by learning optimal directions for missing data during split evaluation, without requiring imputation
- Practitioner B is correct - K-NN requires complete feature vectors for distance calculation, so removing incomplete samples is the only valid approach
- Practitioner C is correct - linear regression requires complete data, and mean imputation is the statistically optimal preprocessing method
- Practitioner D is correct - SVM requires complete vectors for kernel computation, and forward-fill is a standard time-series imputation method that preserves temporal patterns

271. For clustering variable-density regions, which is better? *

Mark only one oval.

- K-means
- Linear regression
- DBSCAN
- Logistic regression

272. A sentiment analysis model (classifying movie reviews as positive/negative) shows the following performance after training for 50 epochs: *

Performance Metrics:
- Training accuracy: 98.5%
- Training loss: 0.08
- Validation accuracy: 72.3%
- Validation loss: 1.45
- Test accuracy: 71.8%
- Model complexity: 3-layer LSTM with 512 hidden units, 2.4M parameters
- Training set size: 5,000 reviews

What is the most likely problem and what evidence supports this diagnosis?

Mark only one oval.

- Data leakage - validation and test performance are similar (72.3% vs 71.8%), suggesting information leaked from training to validation set
- Underfitting - 72% validation accuracy is too low for sentiment analysis, indicating the model failed to learn the underlying patterns in the data
- Overfitting - large train-validation gap (98.5% vs 72.3%), loss divergence (0.08 vs 1.45), and model complexity (2.4M parameters) relative to dataset size (5K samples) indicates the model memorised training data rather than learning generalisable patterns
- Distribution shift - model performs well on training data (98.5%) but test data comes from different distribution, causing performance drop

273. A binary classification task (email spam detection) requires selecting a loss function. The model outputs raw logits (unbounded real values) from final layer, which are then passed through sigmoid activation (σ(z) = 1/(1+e^(-z))) to produce probabilities. *

Four loss functions are proposed:

Loss A - Binary Cross-Entropy (BCE):

L = -[y log(p) + (1-y) log(1-p)], where p = σ(z)
- Penalises confident wrong predictions heavily
- Gradient: ∂L/∂z = p - y (clean gradient for sigmoid)

Loss B - Mean Squared Error (MSE):

L = (y - p)², where p = σ(z)
- Measures squared distance between prediction and target
- Gradient: ∂L/∂z = 2(p - y) × p(1-p) (vanishing gradient when p near 0 or 1)

Loss C - Hinge Loss (SVM-style):

L = max(0, 1 - y × z), where y ∈ {-1, +1}
- Maximises margin between classes
- Not differentiable at margin boundary

Loss D - Mean Absolute Error (MAE):

L = |y - p|, where p = σ(z)
- Measures absolute distance between prediction and target
- Gradient: ∂L/∂z = sign(p - y) × p(1-p) (constant, doesn't scale with error magnitude)

Which loss function is most appropriate for training this neural network, and why?

Mark only one oval.

- Binary Cross-Entropy (Loss A) - designed for probabilistic binary classification, provides strong gradients throughout sigmoid range, penalizes confident mistakes heavily (essential for calibrated spam detection), and gradient scales proportionally with prediction error
- Mean Squared Error (Loss B) - simpler to understand and implement, treats classification as regression problem which is mathematically equivalent
- Hinge Loss (Loss C) - maximises decision margin which improves robustness to adversarial spam examples

Mean Absolute Error (Loss D) - more robust to outliers than MSE, preventing extreme spam examples from dominating training

274. You have imbalanced classes (1% positive). Which metric is misleading? *

Mark only one oval.

F1-score

Precision

Accuracy

Recall

275. A fraud detection dataset contains 10,000 transactions: 9,800 legitimate (98%) and 200 fraudulent (2%). A data scientist trains a logistic regression classifier with default settings and reports: *

Model Performance:
- Overall accuracy: 98.0%
- Precision (fraud): 0.15 (15%)
- Recall (fraud): 0.10 (10%)
- F1-score (fraud): 0.12

Confusion Matrix:
- True Negatives: 9,800 (all legitimate correctly classified)
- False Positives: 0
- False Negatives: 180 (missed frauds)
- True Positives: 20 (caught frauds)

The model predicts "legitimate" for almost every transaction. Four approaches are proposed:

Approach A - Class Weights:
- Modify loss function: $L = w_1 \times L(\text{fraud}) + w_0 \times L(\text{legit})$, where $w_1=49$, $w_0=1$
- Penalises fraud misclassification 49× more (inverse of class frequency ratio)

Approach B - Random Oversampling:
- Duplicate fraud samples randomly until balanced: 9,800 fraud + 9,800 legit
- Train on 19,600 total samples (98% duplicates)

Approach C - SMOTE (Synthetic Minority Over-sampling):
- Generate synthetic fraud samples by interpolating between k-nearest fraud neighbors
- Create 9,600 synthetic frauds + 200 real frauds = 9,800 fraud samples

Approach D - Random Undersampling:
- Randomly remove legitimate samples to balance: 200 fraud + 200 legit
- Train on only 400 samples (discard 9,600 legit transactions)

Which approach best balances practicality and performance?

Mark only one oval.

- Class Weights (Approach A) - preserves all training data, computationally efficient, directly encodes cost of misclassification into optimisation, and works with any algorithm that supports sample/class weighting
- Random Oversampling (Approach B) - simplest implementation, guaranteed to balance classes, and allows model to see minority examples more frequently

- [ ] SMOTE (Approach C) - creates synthetic samples that interpolate feature space, adds diversity beyond simple duplication, and avoids exact memorisation of minority class

- [ ] Random Undersampling (Approach D) - fastest training time with smallest dataset, forces model to focus on minority class patterns

276. After applying class weights to the fraud detection model, three practitioners evaluate performance differently: *

Dataset: 10,000 transactions (9,800 legit, 200 fraud)

Model Results:
- True Positives (TP): 180 (frauds caught)
- False Positives (FP): 294 (legit flagged as fraud)
- True Negatives (TN): 9,506 (legit classified correctly)
- False Negatives (FN): 20 (frauds missed)

Practitioner A reports: "98.7% accuracy - excellent model!"
- Calculation: (TP + TN) / Total = (180 + 9,506) / 10,000 = 98.7%

Practitioner B reports: "38% precision - terrible model, too many false alarms!"
- Calculation: Precision = TP / (TP + FP) = 180 / (180 + 294) = 38%

Practitioner C reports: "90% recall - good model, catches most fraud!"
- Calculation: Recall = TP / (TP + FN) = 180 / (180 + 20) = 90%

Which practitioner's evaluation is most appropriate for this imbalanced classification problem?

Mark only one oval.

- Practitioner A using accuracy - 98.7% correctly classified transactions is objectively the best overall performance metric
- Practitioner B using precision - 38% precision means majority of fraud alerts are false alarms, creating operational burden and user friction
- Practitioner C focusing on recall - catching 90% of frauds (180/200) is the primary objective; false positives (294 legit transactions flagged) can be manually reviewed by fraud team
- None individually sufficient - must evaluate F1-score (harmonic mean of precision and recall) or use ROC-AUC which aggregates performance across all thresholds

277. You're training models on a text classification dataset (5,000 samples, 10 categories). Three models show different performance patterns: *

Model A - Logistic Regression (Linear):
- Training accuracy: 68%
- Validation accuracy: 67%
- Training loss: 1.85
- Validation loss: 1.88
- Behaviour: Similar errors on both train and validation sets

Model B - Deep Neural Network (5 layers, 2M parameters):
- Training accuracy: 99%
- Validation accuracy: 71%
- Training loss: 0.05
- Validation loss: 2.15
- Behaviour: Memorises training data, fails on new examples

Model C - Neural Network with Regularisation (3 layers, dropout=0.5, L2=0.01):
- Training accuracy: 85%
- Validation accuracy: 83%
- Training loss: 0.52
- Validation loss: 0.58
- Behaviour: Balanced performance

Which diagnosis and solution is correct for each model?

Mark only one oval.

- Model A: High bias (underfitting) → increase capacity or add features; Model B: High variance (overfitting) → add regularisation or get more data; Model C: Well-balanced (target this configuration)
- Model A: Correct fit (train-val gap minimal) → no changes needed; Model B: High bias (needs more training) → train longer; Model C: High variance (dropout too strong) → reduce regularisation
- Model A: High variance (too simple) → collect more data; Model B: Correct fit (high accuracy achieved) → deploy; Model C: High bias (accuracy too low) → increase capacity
- All models have data quality issues → focus on data cleaning before model tuning

278. Which architecture is best suited for sequential data with long-term dependencies? *

Mark only one oval.

- Feedforward NN
- Logistic regression
- K-means
- Transformer or LSTM

279. When to use cross-validation vs single train/validation split? *

Mark only one oval.

- Always use CV
- Always use split
- No difference
- Cross-validation for small datasets, split for large

280. Your training loss oscillates wildly. What's likely wrong? *

Mark only one oval.

- Learning rate too low
- Too much data
- Learning rate too high
- Perfect training

281. When might you prefer a generative model over discriminative? *

Mark only one oval.

- Never use generative
- Only for classification
- Always prefer discriminative
- When you need to sample new data or model data distribution

282. Your training loss oscillates wildly without convergence after 50 epochs. Systematic debugging should prioritise which diagnostic sequence? *

Mark only one oval.

- Verify gradient flow with torch.autograd.grad, inspect loss surface curvature, then reduce learning rate with warm restarts
- Check data pipeline integrity (normalisation, augmentation consistency, label encoding), then validate loss function implementation for numerical stability
- Monitor gradient norms across layers, test learning rate with LR range test, then inspect optimiser state for momentum accumulation issues
- Increase batch size for gradient stability, add gradient clipping, then switch to adaptive optimisers (Adam → AdamW)

283. Training abruptly produces NaN losses at epoch 23 after stable convergence. Which hypothesis-driven debugging approach is most effective? *

Mark only one oval.

- Enable anomaly detection (torch.autograd.set_detect_anomaly(True)), log pre-activation statistics, then inspect batch samples causing numerical overflow
- Reduce learning rate by 10x, add gradient clipping at norm=1.0, switch to mixed precision training with loss scaling
- Check for division by zero in loss function, add epsilon (1e-8) to denominators, validate input data range
- Replace batch normalisation with layer normalisation, use gradient checkpointing, decrease model capacity

284. Validation accuracy remains at 10% (random baseline for 10-class problem) while training loss decreases smoothly to 0.01. Which failure mode is most likely? *

Mark only one oval.

- Label mismatch between training and validation sets (different encoding schemes or class mappings)
- Extreme overfitting due to insufficient regularisation and model memorisation of training set
- Vanishing gradients preventing feature learning, causing model to predict uniform distribution
- Data leakage in training set artificially inflating training performance without validation generalisation

285. Training a ResNet-50 with batch size 128 causes "CUDA out of memory" on a 16GB GPU. Which memory optimisation strategy provides best throughput-memory tradeoff? *

Mark only one oval.

- Gradient accumulation (effective batch 128 = 4 micro-batches × 32) with gradient checkpointing on residual blocks
- Mixed precision training (FP16 forward/backward, FP32 optimiser) with dynamic loss scaling
- Reduce batch size to 32 without other changes, accepting 4x longer training time
- Enable torch.cuda.empty_cache() after each backward pass and use pin_memory for data loading

286. Production model shows train accuracy 99.8%, validation 92%, but test accuracy 64%. Training and validation sets are from 2023 data; test set is from 2025 production data. What's the primary issue? *

Mark only one oval.

- Temporal distribution shift - training distribution no longer represents production data characteristics
- Overfitting masked by validation set from same temporal distribution as training set
- Test set contamination or labeling errors causing artificially low accuracy measurement
- Model capacity insufficient to generalise despite high training accuracy

287. Custom model runs on GPU (CUDA) but CPU inference fails with "RuntimeError: mat1 and mat2 shapes cannot be multiplied". Identical input shape (batch_size=1) used for both. What's the root cause? *

Mark only one oval.

- Batch normalisation statistics (running_mean, running_var) stored with GPU-specific batch dimensions causing shape inference errors on CPU
- GPU uses NHWC tensor layout while CPU defaults to NCHW, causing dimension misalignment in convolution operations
- Implicit .cuda() calls within forward() method not wrapped in device-agnostic code (to(device))
- Dynamic control flow (if-statements based on input.is_cuda) creates different computation graphs for GPU vs CPU execution

288. All gradients report zero for final layer during backpropagation, but earlier layers show non-zero gradients. Loss function is custom cross-entropy with label smoothing. What's the most likely culprit? *

Mark only one oval.

- Softmax saturation causing numerically zero gradients when predictions are overconfident (near 0 or 1)
- In-place operations in final layer modifying tensors needed for gradient computation
- Custom loss function returns scalar without maintaining computational graph via torch.mean() or similar
- Dead neurons in final layer (all ReLU outputs zero) preventing gradient flow

289. Multi-class classifier (50 classes) predicts only classes 0, 1, 2 regardless of input after 100 epochs. Training loss converged to 2.8. Class distribution in training set is uniform. Which failure mode best explains this? *

Mark only one oval.

- Final layer bias initialised too negative for classes 3-49, creating insurmountable logit differences during early training
- Class imbalance in mini-batches due to improper shuffling, causing model to optimise for frequent classes
- Learning rate too high for final layer, causing weights to diverge while early layers converge
- Weight initialisation drew final layer weights from distribution with wrong scale, causing partial mode collapse

290. Training loss decreases smoothly for 80 epochs (2.1 → 0.3), then suddenly diverges to infinity at epoch 81. No hyperparameter changes, no data loading errors. What's the most sophisticated explanation? *

Mark only one oval.

- Accumulated floating-point errors in optimiser state (momentum buffers) exceeded numerical precision threshold
- Learning rate schedule caused transition into unstable region of loss landscape where curvature exceeds inverse learning rate
- Batch normalisation running statistics corrupted by outlier batch, causing future normalisation to amplify activations exponentially
- Model entered sharp minimum at epoch 80, then learning rate kicked it into high-loss region due to insufficient generalisation

291. Custom CUDA kernel for attention mechanism produces correct outputs but gradients fail torch.autograd.gradcheck with relative error >0.1. Which debugging strategy is most rigorous? *

Mark only one oval.

- Implement analytical backward pass manually, verify against finite difference approximation with multiple perturbation scales (ε = 1e-3, 1e-5, 1e-7)
- Use torch.autograd.grad with create_graph=True to verify second-order derivatives, then test gradient flow through concatenated operations
- Decompose custom kernel into PyTorch primitives (matmul, softmax), verify equivalent forward pass, then compare gradient outputs elementwise
- Enable CUDA-MEMCHECK for memory access violations, profile with nvprof for race conditions, then validate atomic operations in backward kernel

292. A ResNet-101 training job takes 72 hours on a single V100 GPU (batch size 32, mixed precision disabled). Management requests 10x speedup while maintaining convergence quality. Which combination provides optimal speed-accuracy tradeoff? *

Mark only one oval.

- Distributed data parallel across 8 GPUs with linear learning rate scaling (LR × 8), gradient accumulation to maintain effective batch size 256
- Mixed precision (FP16) + larger batch size (128) + TensorFloat-32 on A100 GPUs, accepting potential generalisation degradation
- Model architecture search for EfficientNet variant with 40% fewer FLOPs, knowledge distillation from original ResNet-101
- Gradient checkpointing + activation recomputation to enable batch size 256 on single GPU, with learning rate warmup extended to 10 epochs

293. Two training runs with identical architecture, data, and optimiser differ only in learning rate schedules. Run A uses constant LR=0.001, achieving 92% validation accuracy. Run B uses 1cycle policy (LR peaks at 0.01), achieving 94% accuracy. Why does Run B outperform? *

Mark only one oval.

- A) High LR phase explores loss landscape broadly, escaping sharp minima; low LR phase exploits flat minima for better generalisation
- B) Learning rate momentum allows model to traverse saddle points faster, reducing training time while maintaining accuracy
- C) Cyclical learning rates prevent overfitting by introducing noise into the optimisation trajectory, acting as implicit regularization
- D) Higher peak learning rate enables larger effective batch sizes through gradient noise, improving convergence rate

294. Training a Transformer (GPT-2 scale: 124M params) from scratch with AdamW diverges within 500 steps despite stable loss for first 100 steps. After adding 2000-step linear learning rate warmup, training stabilises. What mechanism does warmup provide? *

Mark only one oval.

- Prevents large gradient updates from poorly-initialised parameters creating catastrophic Adam second-moment (v_t) estimates
- Gradually exposes model to full dataset complexity, allowing early layers to learn basic features before tackling difficult examples
- Stabilises batch normalisation statistics by allowing running mean/variance to accumulate before high learning rates
- Allows optimiser momentum terms to initialise properly, preventing oscillations from zero-initialised momentum buffers

295. Experimental results show ResNet-50 on ImageNet: *

- Batch 256: 76.5% top-1, 8 hours/epoch

- Batch 2048: 75.8% top-1, 1.5 hours/epoch

- Batch 8192: 73.2% top-1, 0.5 hours/epoch

Compute budget allows 90 epochs. Which strategy maximises validation accuracy?

Mark only one oval.

- Batch 256 baseline, train maximum epochs within time budget (train ~15 epochs only due to time = 120 hours)
- Batch 2048 with LARS optimiser + label smoothing + 5-epoch warmup, balancing speed and generalisation (train 90 epochs = 135 hours)
- Batch 8192 with gradient accumulation (4 micro-batches of 2048), maintaining 2048 statistics while using 8192 hardware throughput
- Adaptive batch size: start at 256 (first 10 epochs), increase to 2048 (next 40), finish at 8192 (last 40) as loss landscape flattens

296. You're training a BERT model (110M parameters) on a 16GB GPU. Batch size 32 causes "CUDA out of memory" error. Batch size 8 fits in memory but training is unstable. You implement gradient accumulation with accumulation_steps=4. What is the effective outcome? *

Mark only one oval.

- Training speed increases 4x because gradients are computed in parallel across accumulation steps
- Memory usage decreases by 75% because gradients are discarded after each micro-batch
- Model accuracy improves because gradient accumulation acts as implicit regularisation through gradient noise
- Effective batch size becomes 32 (8×4) while maintaining 8-sample memory footprint; gradients averaged across 4 micro-batches before optimiser step

297. Training a ResNet-34 on CIFAR-10 with constant learning rate 0.1 for 200 epochs. Loss decreases rapidly for 50 epochs, then plateaus at 0.45 with training accuracy 88% and validation accuracy 85%. The loss curve shows continued oscillation around 0.45 rather than smooth convergence. What learning rate schedule would most improve final performance? *

Mark only one oval.

- Step decay: reduce LR by 10× at epochs 100 and 150 - allows initial fast convergence then refined optimisation near minimum
- Start with LR 0.01 from epoch 0 - prevents initial overshooting and maintains stability throughout training
- Increase LR gradually from 0.01 to 0.1 over first 100 epochs - enables careful exploration of loss landscape
- Keep constant LR 0.1 but increase batch size from 128 to 512 - effectively reduces learning rate through larger gradient averaging

298. Why might Adam optimiser be preferred over Stochastic Gradient Descent (SGD)? *

Mark only one oval.

- Always gives better results
- Simpler
- Adaptive learning rates per parameter, faster convergence
- No particular reason

299. Data augmentation primarily helps: *

Mark only one oval.

- Speed up training
- Reduce model size
- Improve test speed
- Reduce overfitting by increasing effective dataset size

300. Mixed precision training (FP16) benefits: *

Mark only one oval.

- Faster training and reduced memory with minimal accuracy loss
- Always reduces accuracy
- Only for CPUs
- No benefits

301. When might you prefer Stochastic Gradient Descent (SGD) with momentum over Adam? *

Mark only one oval.

- For better generalisation in some settings
- Never
- Always
- Random choice

302. ML pipeline typically includes: *

Mark only one oval.

- Data collection → Preprocessing → Training → Evaluation → Deployment
- Deployment → Evaluation → Data collection → Preprocessing →Training
- Preprocessing → Data collection → Training → Evaluation → Deployment
- Data collection → Preprocessing → Evaluation → Training → Deployment

303. Consider two experimental protocols: *

Protocol A: 70% train, 30% test. Iterate: train model → evaluate test accuracy → adjust hyperparameters → retrain. Report best test accuracy across 50 iterations.

Protocol B: 60% train, 20% validation, 20% test. Train model → tune hyperparameters on validation → evaluate test once. Report single test accuracy.

Protocol A reports 94.2% test accuracy. Protocol B reports 91.8%. Which protocol provides more credible generalisation estimates?

Mark only one oval.

- Protocol A - larger training set (70% vs 60%) and extensive hyperparameter search (50 iterations) produces genuinely better model with higher true performance
- Protocol A - multiple test evaluations provide confidence intervals through repeated sampling, making 94.2% more reliable than single 91.8% measurement
- Protocol B - test set remains untouched during hyperparameter search, preventing selection bias; Protocol A's test accuracy is optimistically biased due to adaptive overfitting
- Both equivalent - test sets are identically sized (30% vs 20%), so generalisation estimates have similar statistical power

304. A tabular dataset for credit risk prediction contains: numerical features (income, age), categorical features (employment_type, education), missing values (15% of records), and outliers (income values reaching $50M). Which preprocessing pipeline is most appropriate before model training? *

Mark only one oval.

- Remove all rows with missing values → normalise numerical features to [0,1] range → label encode categorical variables → proceed to training
- Fill missing values with zeros → remove outlier rows (>3σ) → leave numerical features as-is → convert categorical to integer codes
- Impute missing values (median for numerical, mode for categorical) → detect and cap outliers via IQR method → standardise numerical features (z-score) → one-hot encode categorical variables
- Apply PCA dimensionality reduction → normalise to unit variance → encode categoricals with target encoding → impute remaining missing values

305. A binary classification dataset contains three features: age (18-80 years), income ($20K-$500K), and credit_score (300-850). Training a logistic regression model without normalisation achieves 72% accuracy. After applying StandardScaler (z-score normalisation), accuracy improves to 84%. What mechanism explains this improvement? *

Mark only one oval.

- Normalisation removes outliers in the income distribution, preventing the model from overfitting to high-earners
- Normalisation increases the effective learning rate for low-variance features while decreasing it for high-variance features, balancing convergence
- StandardScaler centers data at zero, which activates the sigmoid function's linear region for more expressive decision boundaries
- Gradient descent converges faster when features have similar scales - large income values (up to 500K) dominate gradient updates, while small age values contribute negligibly without normalisation

306. A production machine learning system needs to retrain daily on new customer transaction data. The current manual process takes 6 hours: download CSV from database (30 min), clean missing values (1 hour), feature engineering (2 hours), train model (1.5 hours), deploy (1 hour). The team proposes building a data pipeline. What are the essential components of this pipeline? *

Mark only one oval.

- Cron job to run training script at midnight, bash script to download data, Python script for preprocessing, manual model deployment after checking metrics
- Data warehouse for storage, Jupyter notebook for analysis, CSV export functionality, email notifications when training completes
- Orchestration (schedule/trigger), ETL stages (extract from DB, transform/clean/engineer features, load to training), model training/validation, deployment automation with versioning
- Distributed computing cluster (Spark), real-time streaming ingestion (Kafka), feature store, MLOps platform for monitoring

307. Training a convolutional neural network for medical image classification (X-rays: 512×512 pixels, grayscale, pixel values 0-255). Three preprocessing strategies are proposed: *

- Strategy A: Min-Max scaling to [0,1] - subtract min(0), divide by max(255)
- Strategy B: Z-score normalisation - subtract mean(127.5), divide by std(across entire dataset)
- Strategy C: Per-image standardisation - for each image, subtract its own mean, divide by its own std

Validation Results:
- Strategy A: 82.3% accuracy, stable training
- Strategy B: 84.7% accuracy, stable training
- Strategy C: 79.1% accuracy, unstable loss curves

Why does Strategy B outperform despite all images having similar value ranges?

Mark only one oval.

- Z-score normalisation centers activations around zero, placing inputs in the most sensitive region of activation functions (tanh/sigmoid) and enabling better gradient flow
- Strategy B normalises using dataset-wide statistics, preserving relative brightness differences between images that carry diagnostic information; Strategy C destroys this signal
- Min-max scaling (Strategy A) compresses dynamic range when images don't span full [0,255] range; z-score preserves contrast information
- Strategy B's centering at zero enables batch normalisation layers to work optimally, as BN expects zero-mean inputs for numerical stability

308. A fraud detection model is retrained weekly. After 6 weeks, the compliance team requests: "Show us the model from Week 3 that flagged transaction #X as fraudulent." The ML engineer cannot reproduce Week 3's model because they only saved the latest checkpoint. What information must be versioned to fully reproduce a trained model? *

Mark only one oval.

- Model weights (.pth file) and training script (.py file) - these are sufficient to reload and make predictions
- Model weights/checkpoint, training data version/hash, hyperparameters, random seeds, library versions (Python, PyTorch, scikit-learn), preprocessing artifacts (scalers, encoders)
- Git commit hash of training code and model accuracy metrics on validation set
- Model architecture definition (layers, dimensions) and final loss value from training

309. A startup's ML system has grown organically over 2 years. Current state: *

- Development: Data scientists train models in Jupyter notebooks, manually copy best model to production folder

- Deployment: DevOps team manually updates Flask API with new model file once a month

- Monitoring: Check Cloudwatch logs for 5xx errors, no model-specific metrics

- Data: Training data downloaded from production DB, no version control

- Rollback: If model breaks, restore previous Flask container (last month's version)

The CTO wants to "implement MLOps." Which system components provide maximum operational maturity improvement?

Mark only one oval.

- Hire ML platform engineer, buy commercial MLOps tool (Databricks, SageMaker), migrate all workflows to new platform
- Move Jupyter notebooks to git repository, add unit tests to training code, create Docker container for reproducibility
- Experiment tracking (MLflow), automated training pipelines (Airflow), model registry with staging/production promotion, automated deployment (CI/CD), monitoring dashboards (Grafana) with prediction drift detection
- Implement real-time predictions (replace batch), add more powerful GPUs, retrain models daily instead of monthly

310. Your deployed loan approval model shows these metrics: *

Week 1: Approval rate = 40%, Error rate = 0.1%

Week 2: Approval rate = 35%, Error rate = 0.1%

Week 3: Approval rate = 30%, Error rate = 0.1%

Ground truth labels (loan repayment outcomes) take 3 months to collect. Which monitoring metric would alert you to potential model problems FIRST?

Mark only one oval.

- Error rate
- Approval rate
- Ground truth accuracy
- No metric can detect problems until ground truth labels arrive

311. A credit scoring model trained on 2023 data is deployed in January 2024. By June 2024, you observe: *

Input Data Changes:

- Average income: $58K → $52K (decreased)

- Gig-economy workers: 5% → 15% (tripled)

- Average age: 42 → 38 years

Model Performance:

- January 2024 accuracy: 81%

- June 2024 accuracy: 76% (5% drop)

Business Context: Economic downturn in March 2024 caused unemployment to rise and shift to gig jobs.

What is happening and what should you do?

Mark only one oval.

- The model's decision threshold is miscalibrated - adjust the threshold from 0.5 to a different value without retraining
- The input data distribution has changed (people applying for loans look different now) - retrain the model on recent 2024 data that includes these new patterns
- This is normal variation - 5% accuracy drop is acceptable, continue monitoring
- The features are invalid - remove income and employment_type features entirely