Machine Learning - 100 Answers

Pages 12+ Part 3: With Explanations

Nova IMS
Generated: January 16, 2026

LEVEL 1: RANDOM FOREST BASICS

Q1. What type of ensemble method is Random Forest?

ANSWER: B

Explanation: Random Forest is a bagging (Bootstrap Aggregating) ensemble method that trains multiple decision trees on different bootstrap samples and averages their predictions.

Q2. What is bootstrap sampling in Random Forest?

ANSWER: B

Explanation: Bootstrap sampling draws samples with replacement from the original dataset, creating new datasets of the same size. Each bootstrap sample contains ~63% unique samples, with ~37% left out (out-of-bag).

Q3. What percentage of data does each bootstrap sample typically contain?

ANSWER: B

Explanation: Each bootstrap sample contains approximately 63% of the original data because sampling with replacement means some samples appear multiple times while others don't appear at all. The remaining ~37% are out-of-bag samples.

Q4. What are the two sources of randomness in Random Forest?

ANSWER: B

Explanation: Random Forest introduces randomness through (1) bootstrap sampling of data for each tree, and (2) random feature subsampling at each split (each split considers only a random subset of features).

Q5. How many features does each split consider in Random Forest?

ANSWER: B

Explanation: At each split, Random Forest randomly selects a subset of features to consider. Typically $\sqrt{n}$ features for classification and n/3 for regression, where n is total features. This decorrelates trees.

Q6. Why does Random Forest use feature subsampling at splits?

ANSWER: B

Explanation: Feature subsampling decorrelates trees by preventing strong features from dominating every tree. Without it, all trees would use the same strong features first and be highly correlated, reducing ensemble benefit.

Q7. How does Random Forest make predictions for classification?

ANSWER: B

Explanation: For classification, Random Forest uses majority vote across all trees. Each tree votes for a class, and the class with the most votes is the final prediction.

Q8. How does Random Forest make predictions for regression?

ANSWER: B

Explanation: For regression, Random Forest averages the predictions from all trees: ■ = (1/n_trees) Σ tree_predictions. This averaging reduces variance.

Q9. What is Out-of-Bag (OOB) error?

ANSWER: B

Explanation: OOB (Out-of-Bag) error is the validation error computed using samples not included in each tree's bootstrap sample (~37% per tree). Each sample is predicted by trees that didn't see it during training.

Q10. What is an advantage of OOB error?

ANSWER: B

Explanation: OOB error provides built-in validation without requiring a separate validation set. It's an efficient way to estimate generalization error during training.

Q11. What is the n_estimators hyperparameter?

ANSWER: B

Explanation: n_estimators is the number of decision trees in the forest. More trees generally improve performance but increase computation time and memory.

Q12. What is a typical range for n_estimators?

ANSWER: B

Explanation: Typical values range from 100-500 trees. Common starting point is 100, then increase if needed. Beyond 500 rarely provides significant improvement.

Q13. Does Random Forest typically overfit with more trees?

ANSWER: B

Explanation: No, Random Forest rarely overfits with more trees. Performance typically plateaus but doesn't degrade. More trees increase computation time but not overfitting risk.

Q14. What is the max_features hyperparameter?

ANSWER: B

Explanation: max_features is the number of features to randomly consider at each split. Common values: $\sqrt{n}$ for classification, $n/3$ for regression, where n is total features.

Q15. What is the max_depth hyperparameter in Random Forest?

ANSWER: B

Explanation: max_depth limits the maximum depth of each tree. Random Forest often uses unlimited depth (None) because ensemble averaging prevents overfitting from deep trees.

Q16. Do Random Forest trees need pruning?

ANSWER: B

Explanation: No, Random Forest trees don't need pruning. Deep trees with high variance are fine because averaging across many decorrelated trees reduces overall variance.

Q17. What is an advantage of Random Forest?

ANSWER: B

Explanation: Advantages: Excellent performance out-of-box with minimal tuning, naturally handles non-linear relationships, provides feature importance, reduces overfitting through averaging, robust to outliers and noise.

Q18. What is a limitation of Random Forest?

ANSWER: B

Explanation: Limitations: Less interpretable than single tree (black box), slower prediction (must query all trees), requires more memory (stores all trees), harder to understand model decisions.

Q19. Does Random Forest require feature scaling?

ANSWER: B

Explanation: No, Random Forest doesn't require feature scaling because it's tree-based. Splits are based on feature thresholds which are scale-invariant.

Q20. How does Random Forest handle feature importance?

ANSWER: B

Explanation: Random Forest calculates feature importance based on average decrease in impurity (Gini/entropy) when that feature is used for splits, weighted by the number of samples. Alternatively, permutation importance measures performance drop when feature values are shuffled.

LEVEL 2: GRADIENT BOOSTING FUNDAMENTALS

Q21. What type of ensemble method is Gradient Boosting?

ANSWER: B

Explanation: Gradient Boosting is a boosting ensemble method where models are trained sequentially, with each new model correcting errors made by the previous ensemble.

Q22. How are models trained in boosting?

ANSWER: B

Explanation: In boosting, models are trained sequentially. Each new model focuses on correcting the mistakes (residual errors) of the current ensemble, with models weighted by their performance.

Q23. What does each new tree in Gradient Boosting learn?

ANSWER: B

Explanation: Each new tree in Gradient Boosting learns to predict the residual errors (negative gradients) of the current ensemble, gradually reducing overall prediction error.

Q24. What is the general Gradient Boosting algorithm?

ANSWER: B

Explanation: Algorithm: (1) Initialize with simple model (often mean/median), (2) Compute residuals (errors), (3) Train new tree on residuals, (4) Add tree to ensemble with learning rate, (5) Repeat until stopping criterion.

Q25. What does Gradient Boosting minimize?

ANSWER: B

Explanation: Gradient Boosting minimizes a differentiable loss function using gradient descent in function space, where 'functions' are the trees being added to the ensemble.

Q26. What is the learning rate (shrinkage) in Gradient Boosting?

ANSWER: B

Explanation: Learning rate (shrinkage, typically 0.01-0.3) scales the contribution of each tree. Lower learning rates mean each tree has less influence, requiring more trees but often achieving better generalization.

Q27. What is the effect of a low learning rate?

ANSWER: B

Explanation: Low learning rate requires more trees to reach the same training performance but typically achieves better generalization by making smaller, more careful steps toward the minimum.

Q28. What is the typical tree depth in Gradient Boosting?

ANSWER: B

Explanation: Gradient Boosting typically uses shallow trees (depth 3-8), much shallower than Random Forest. Depth 3-5 is common (often called 'stumps' for depth 1).

Q29. Why use shallow trees in Gradient Boosting?

ANSWER: B

Explanation: Shallow trees prevent individual trees from overfitting. Each weak learner captures simple patterns, and the ensemble combines them to learn complex relationships. Deep trees would overfit to the current residuals.

Q30. What is the subsample hyperparameter?

ANSWER: B

Explanation: Subsample parameter (e.g., 0.8) means each tree is trained on only a random fraction (80%) of training samples. This adds randomness similar to Random Forest.

Q31. What does subsample < 1.0 do?

ANSWER: B

Explanation: Subsample < 1.0 introduces stochastic gradient boosting, adding randomness that helps prevent overfitting and can improve generalization, similar to bagging.

Q32. What is XGBoost?

ANSWER: B

Explanation: XGBoost (eXtreme Gradient Boosting) is an optimized implementation of gradient boosting featuring regularization (L1/L2), parallel processing, handling missing values, tree pruning, and sparsity awareness.

Q33. What are key innovations in XGBoost?

ANSWER: B

Explanation: XGBoost innovations: Built-in regularization to prevent overfitting, parallelized tree construction, automatic handling of missing values, intelligent tree pruning (max_depth then prune back), cache optimization.

Q34. What is LightGBM?

ANSWER: B

Explanation: LightGBM (Light Gradient Boosting Machine) is a faster gradient boosting implementation using leaf-wise growth and histogram-based learning, designed for large datasets.

Q35. What is unique about LightGBM's tree growth?

ANSWER: B

Explanation: LightGBM uses leaf-wise (best-first) growth: it splits the leaf with maximum delta loss rather than growing level by level. This is faster and often more accurate but risks overfitting.

Q36. What is CatBoost?

ANSWER: B

Explanation: CatBoost (Categorical Boosting) is designed to handle categorical features natively using ordered target statistics, reducing target leakage and preprocessing needs.

Q37. What is an advantage of Gradient Boosting?

ANSWER: B

Explanation: Gradient Boosting typically achieves state-of-the-art performance on tabular/structured data, often winning Kaggle competitions. It's the go-to algorithm for structured data.

Q38. What is a limitation of Gradient Boosting?

ANSWER: B

Explanation: Limitations: Sequential training is computationally expensive (can't parallelize across trees), prone to overfitting without proper regularization, requires careful hyperparameter tuning, sensitive to outliers.

Q39. What is early stopping in Gradient Boosting?

ANSWER: B

Explanation: Early stopping monitors validation error and stops training when it stops improving for a specified number of rounds (patience). This prevents overfitting by stopping before memorizing training noise.

Q40. When is Gradient Boosting preferred over Random Forest?

ANSWER: B

Explanation: Gradient Boosting is preferred when maximum accuracy is critical and computational cost is acceptable, especially for structured/tabular data. Random Forest is faster and simpler but often slightly less accurate.

LEVEL 3: NEURAL NETWORKS ARCHITECTURE

Q41. What are the three types of layers in neural networks?

ANSWER: B

Explanation: Neural networks have three layer types: Input layer (receives features), Hidden layers (learn representations), Output layer (produces predictions).

Q42. What does the input layer do?

ANSWER: B

Explanation: The input layer receives feature values, with one neuron per feature. It has no computation - it simply passes values to the first hidden layer.

Q43. What do hidden layers do?

ANSWER: B

Explanation: Hidden layers perform computations to learn hierarchical feature representations. Early layers learn simple patterns (edges in images), deeper layers learn complex patterns (objects, faces).

Q44. What does the output layer do?

ANSWER: B

Explanation: The output layer produces final predictions. Number of neurons depends on task: 1 for binary classification (sigmoid) or regression (linear), k for k-class classification (softmax).

Q45. How many output neurons for binary classification?

ANSWER: B

Explanation: Binary classification uses 1 output neuron with sigmoid activation, outputting probability $P(y=1)$. Threshold at 0.5: if output>0.5, predict class 1, else class 0.

Q46. How many output neurons for multiclass classification with 10 classes?

ANSWER: B

Explanation: Multiclass classification with k classes uses k output neurons with softmax activation. Each neuron represents one class's probability, and outputs sum to 1.

Q47. How many output neurons for regression?

ANSWER: B

Explanation: Regression uses 1 output neuron with linear activation (no activation function), outputting continuous values from $-\infty$ to $+\infty$.

Q48. What is a neuron's computation?

ANSWER: B

Explanation: Each neuron computes: z = Σ(w■x■) + b (weighted sum plus bias), then a = activation(z). The activation function introduces non-linearity.

Q49. What are weights in neural networks?

ANSWER: B

Explanation: Weights are learnable parameters (connections between neurons) that determine the strength and direction of information flow. They're updated during training via backpropagation.

Q50. What is bias in a neuron?

ANSWER: B

Explanation: Bias is an additional learnable parameter (offset/intercept) for each neuron, allowing neurons to activate even when inputs are zero. It shifts the activation function.

Q51. What is forward propagation?

ANSWER: B

Explanation: Forward propagation passes inputs through the network layer by layer, computing weighted sums and applying activations, until reaching output predictions.

Q52. What is the universal approximation theorem?

ANSWER: B

Explanation: The Universal Approximation Theorem states that a neural network with at least one hidden layer and sufficient neurons can approximate any continuous function to arbitrary accuracy.

Q53. How many hidden layers defines a "deep" neural network?

ANSWER: B

Explanation: A 'deep' neural network has 2 or more hidden layers. 'Shallow' networks have 0-1 hidden layers. Deep networks learn hierarchical representations.

Q54. What is a fully connected (dense) layer?

ANSWER: B

Explanation: A fully connected (dense) layer connects every neuron in the layer to every neuron in the previous layer. For layer with m neurons and previous layer with n neurons, there are m×n weights.

Q55. What architecture is used for images?

ANSWER: B

Explanation: CNNs (Convolutional Neural Networks) use convolutional and pooling layers designed to process grid-structured data like images, exploiting spatial locality and translation invariance.

Q56. What architecture is used for sequential data?

ANSWER: B

Explanation: RNNs (Recurrent Neural Networks) and LSTMs process sequential data (text, time series, audio) by maintaining hidden states that capture information from previous time steps.

Q57. What are skip connections (residual connections)?

ANSWER: B

Explanation: Skip connections (residual connections) add direct connections that bypass one or more layers (e.g., x + F(x) instead of just F(x)). Used in ResNet architecture.

Q58. Why use skip connections?

ANSWER: B

Explanation: Skip connections help gradients flow backward through very deep networks, mitigating vanishing gradient problem and enabling training of networks with 100+ layers.

Q59. What is network width?

ANSWER: B

Explanation: Network width is the number of neurons per layer. Wider networks can learn more complex functions but require more computation and data.

Q60. What is network depth?

ANSWER: B

Explanation: Network depth is the number of layers. Deeper networks learn hierarchical representations and can model more complex functions with fewer total parameters.

LEVEL 4: ACTIVATION FUNCTIONS & TRAINING

Q61. Why are activation functions necessary?

ANSWER: B

Explanation: Activation functions introduce non-linearity, enabling networks to learn complex non-linear patterns. Without them, networks reduce to linear models regardless of depth.

Q62. What happens without activation functions (or with only linear)?

ANSWER: B

Explanation: Without activation functions (or with only linear activations), multiple layers collapse to a single linear transformation: f(f(x)) is still linear. The network cannot learn non-linear patterns.

Q63. What is the sigmoid activation function?

ANSWER: B

Explanation: Sigmoid function: σ(x) = 1/(1+e^(-x)), mapping any real value to range (0,1). S-shaped curve.

Q64. What is sigmoid's output range?

ANSWER: B

Explanation: Sigmoid outputs range from 0 to 1, making it suitable for binary classification probabilities. As x→+∞, σ→1; as x→-∞, σ→0.

Q65. What is a problem with sigmoid?

ANSWER: B

Explanation: Sigmoid suffers from vanishing gradients: for large |x|, the gradient approaches zero. During backpropagation, gradients become extremely small, preventing learning in deep networks.

Q66. What is the tanh activation function?

ANSWER: B

Explanation: Tanh (hyperbolic tangent): tanh(x) = (e^x - e^(-x))/(e^x + e^(-x)), mapping inputs to range (-1,1). Zero-centered (unlike sigmoid).

Q67. What is tanh's output range?

ANSWER: B

Explanation: Tanh outputs range from -1 to 1. Being zero-centered helps optimization converge faster compared to sigmoid's (0,1) range.

Q68. What is ReLU (Rectified Linear Unit)?

ANSWER: B

Explanation: ReLU (Rectified Linear Unit): ReLU(x) = max(0,x). For positive inputs returns x, for negative returns 0.

Q69. What is ReLU's output range?

ANSWER: B

Explanation: ReLU outputs range from 0 to infinity [0,∞). It's unbounded on the positive side.

Q70. Why is ReLU most popular for hidden layers?

ANSWER: B

Explanation: ReLU is most popular because: (1) Computationally very simple, (2) No vanishing gradient for positive values, (3) Sparse activation (many zeros), (4) Empirically works very well.

Q71. What is the "dying ReLU" problem?

ANSWER: B

Explanation: 'Dying ReLU' occurs when neurons output 0 for all inputs (neuron 'dies'). This happens when weights are updated such that the neuron never activates (always in negative region), causing zero gradients and no learning.

Q72. What is Leaky ReLU?

ANSWER: B

Explanation: Leaky ReLU: f(x) = max(αx, x) where α is small (typically 0.01). For negative inputs returns αx instead of 0, allowing small gradient flow and preventing dying neurons.

Q73. What is the softmax function used for?

ANSWER: B

Explanation: Softmax converts raw scores (logits) into a probability distribution for multiclass classification: softmax(z■) = e^(z■)/Σ■e^(z■). Each output is in (0,1) and sum to 1.

Q74. What does softmax output sum to?

ANSWER: B

Explanation: Softmax outputs sum exactly to 1.0, forming a valid probability distribution over all classes.

Q75. What is backpropagation?

ANSWER: B

Explanation: Backpropagation computes gradients of the loss function with respect to all weights using the chain rule, propagating errors backward through the network from output to input.

Q76. What is the chain rule used for in backpropagation?

ANSWER: B

Explanation: The chain rule allows computing gradients layer by layer: ∂Loss/∂w■ = (∂Loss/∂output) × (∂output/∂hidden) × (∂hidden/∂w■). This enables gradient flow through multiple layers.

Q77. What is gradient descent?

ANSWER: B

Explanation: Gradient descent iteratively updates weights to minimize loss: w_new = w_old - learning_rate × gradient. It takes steps in the direction that reduces the loss.

Q78. What is the learning rate in neural networks?

ANSWER: B

Explanation: Learning rate controls the step size for weight updates. Too large causes overshooting/divergence, too small causes slow convergence. Typical range: 0.001-0.1.

Q79. What happens with too large learning rate?

ANSWER: B

Explanation: Too large learning rate causes overshooting: weights jump past the minimum, loss oscillates or diverges. Training becomes unstable, potentially exploding to infinity.

Q80. What happens with too small learning rate?

ANSWER: B

Explanation: Too small learning rate causes very slow convergence - training takes extremely long and may get stuck in local minima or plateau regions before reaching good performance.

LEVEL 5: NEURAL NETWORK REGULARIZATION & OPTIMIZATION

Q81. What is dropout?

ANSWER: B

Explanation: Dropout randomly sets a fraction of neurons to 0 during each training iteration, forcing the network to learn redundant representations and preventing over-reliance on specific neurons.

Q82. What is a typical dropout rate?

ANSWER: B

Explanation: Typical dropout rates are 0.2-0.5 (20-50% of neurons dropped). 0.5 is common for fully connected layers, 0.2 for convolutional layers.

Q83. When is dropout applied?

ANSWER: B

Explanation: Dropout is applied only during training. During inference, all neurons are active and outputs are scaled appropriately to account for all neurons being present.

Q84. What is batch normalization?

ANSWER: B

Explanation: Batch normalization normalizes inputs to each layer by subtracting batch mean and dividing by batch standard deviation, then applying learnable scale and shift parameters.

Q85. What problem does batch normalization address?

ANSWER: B

Explanation: Batch normalization addresses internal covariate shift (changing distributions of layer inputs during training), stabilizing training, allowing higher learning rates, and reducing sensitivity to initialization.

Q86. What is the vanishing gradient problem?

ANSWER: B

Explanation: Vanishing gradient occurs when gradients become extremely small (close to 0) as they propagate backward through many layers, preventing early layers from learning.

Q87. What causes vanishing gradients?

ANSWER: B

Explanation: Vanishing gradients are caused by repeatedly multiplying small gradients through many layers. With sigmoid/tanh (gradients <1), products of many small numbers approach zero exponentially.

Q88. What is the exploding gradient problem?

ANSWER: B

Explanation: Exploding gradient occurs when gradients become extremely large during backpropagation, causing unstable training with wild weight updates and potential overflow (NaN values).

Q89. What is gradient clipping?

ANSWER: B

Explanation: Gradient clipping limits gradient magnitude to a threshold. If ||gradient|| > threshold, scale it down: gradient = threshold × gradient/||gradient||. This prevents exploding gradients.

Q90. What is the Adam optimizer?

ANSWER: B

Explanation: Adam (Adaptive Moment Estimation) combines momentum (exponential moving average of gradients) and RMSprop (adaptive learning rates). It adapts learning rates per parameter based on first and second moments.

Q91. What does Adam stand for?

ANSWER: B

Explanation: Adam stands for Adaptive Moment Estimation, referring to its use of first moment (mean) and second moment (variance) estimates of gradients.

Q92. What is momentum in optimization?

ANSWER: B

Explanation: Momentum uses exponentially weighted average of past gradients rather than just current gradient: $v = \beta v + (1-\beta)\nabla L$, then $w = w - \alpha v$. This accelerates in consistent directions.

Q93. Why use momentum?

ANSWER: B

Explanation: Momentum accelerates convergence in relevant directions while dampening oscillations in irrelevant directions. It helps escape plateaus and local minima by building 'velocity' in consistent gradient directions.

Q94. What is an epoch?

ANSWER: B

Explanation: An epoch is one complete pass through the entire training dataset. If you have 1000 samples and train for 10 epochs, the model sees each sample 10 times.

Q95. What is batch size?

ANSWER: B

Explanation: Batch size is the number of samples processed before updating weights. Mini-batch gradient descent processes batches (e.g., 32, 64, 128) - a compromise between batch (all samples) and stochastic (1 sample).

LEVEL 6: SUPPORT VECTOR MACHINES & HYPERPARAMETER TUNING

Q96. What is the goal of SVM?

ANSWER: B

Explanation: SVM finds the hyperplane (decision boundary) that maximizes the margin (distance) between classes. Maximum margin improves generalization to unseen data.

Q97. What is the margin in SVM?

ANSWER: B

Explanation: The margin is the distance from the decision boundary (hyperplane) to the nearest data points of either class. Larger margins generally indicate better generalization.

Q98. What are support vectors?

ANSWER: B

Explanation: Support vectors are the data points that lie on the margin boundaries or within the margin. They're the critical points that define the decision boundary - removing other points doesn't change the boundary.

Q99. What is the kernel trick?

ANSWER: B

Explanation: The kernel trick implicitly maps data to higher-dimensional space without computing the transformation explicitly. This allows non-linear decision boundaries while keeping computation tractable through kernel functions.

Q100. What is the RBF (Gaussian) kernel?

ANSWER: B

Explanation: RBF (Radial Basis Function) kernel: $K(x,y) = \exp(-\gamma||x-y||^2)$, measures similarity based on distance. It's the most popular non-linear kernel, creating smooth, flexible decision boundaries.