Machine Learning - 100 Answers

Pages 12+ Part 4: With Explanations

Nova IMS
Generated: January 20, 2026

LEVEL 1: SUPPORT VECTOR MACHINES DETAILED

Q1. What is a hard margin SVM?

ANSWER: B

Explanation: Hard margin SVM requires all data points to be correctly classified with no points inside the margin. This only works for perfectly linearly separable data without noise.

Q2. What is a soft margin SVM?

ANSWER: B

Explanation: Soft margin SVM allows some misclassifications and points within the margin, introducing slack variables to handle non-separable data and noise. The penalty for violations is controlled by parameter C.

Q3. What does the C parameter control in SVM?

ANSWER: B

Explanation: The C parameter controls the tradeoff between maximizing the margin (model simplicity) and minimizing classification errors (training accuracy). It's the regularization parameter for SVM.

Q4. What happens with large C value?

ANSWER: B

Explanation: Large C heavily penalizes misclassifications, forcing a narrow margin that tries to classify all points correctly. This risks overfitting by being too sensitive to individual points.

Q5. What happens with small C value?

ANSWER: B

Explanation: Small C allows more margin violations, creating a wider margin that tolerates some errors. This increases regularization and improves generalization but may underfit.

Q6. What is the linear kernel?

ANSWER: B

Explanation: The linear kernel performs no transformation: K(x,y) = x^T y (simple dot product). It searches for the best linear hyperplane in the original feature space.

Q7. When should you use linear kernel?

ANSWER: B

Explanation: Use linear kernel when data is linearly separable or when you have very high-dimensional data (text data with thousands of features), where linear boundaries often work well and training is fast.

Q8. What is the polynomial kernel?

ANSWER: B

Explanation: The polynomial kernel transforms data using polynomial combinations: K(x,y) = (x^T y + c)^d, where d is the degree. It can capture polynomial relationships between features.

Q9. What does the degree parameter control in polynomial kernel?

ANSWER: B

Explanation: The degree parameter (d) controls the polynomial order: d=2 gives quadratic boundaries, d=3 gives cubic boundaries, etc. Higher degrees create more complex, flexible boundaries.

Q10. What is the RBF kernel formula?

ANSWER: B

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

Q11. What does gamma (γ) control in RBF kernel?

ANSWER: B

Explanation: Gamma (γ) controls the influence radius of a single training example. High gamma means only nearby points strongly influence the decision boundary; low gamma means far points also matter.

Q12. What happens with large gamma?

ANSWER: B

Explanation: Large gamma creates very localized influence - only very close points affect classification. This creates complex, wiggly boundaries that risk overfitting by closely following training points.

Q13. What happens with small gamma?

ANSWER: B

Explanation: Small gamma creates far-reaching influence - distant points also affect decisions. This produces smoother, more generalized boundaries that may underfit if too small.

Q14. What is the sigmoid kernel?

ANSWER: B

Explanation: Sigmoid kernel is K(x,y) = tanh(γx^T y + c), similar to neural network activation functions. It's less commonly used and can be unstable in some cases.

Q15. Which kernel is most popular for SVM?

ANSWER: B

Explanation: RBF kernel is most popular because it handles non-linear patterns well, has only two hyperparameters (C and gamma vs polynomial's C, degree, and coef0), and works well across many problems.

Q16. Why is RBF kernel most popular?

ANSWER: B

Explanation: Kernel selection depends on data characteristics (linear vs non-linear separability), computational resources, and cross-validation performance. Start with RBF for most problems, try linear for high-dimensional data.

Q17. What is kernel selection based on?

ANSWER: B

Explanation: Yes, SVMs are very sensitive to feature scales because they use distance-based calculations. Features with larger scales will dominate the distance metric. Always standardize features before SVM.

Q18. Do SVMs require feature scaling?

ANSWER: B

Explanation: SVMs can achieve high accuracy in high dimensions, use only support vectors (memory efficient), and have strong theoretical foundations. However, they require careful scaling and tuning.

Q19. What is an advantage of SVM?

ANSWER: B

Explanation: SVMs have $O(n^2)$ to $O(n^3)$ training complexity (slow for large datasets), require feature scaling, need careful hyperparameter tuning (C, gamma, kernel), and are sensitive to class imbalance.

Q20. What is a limitation of SVM?

ANSWER: B

Explanation: No, SVM decision functions output distances from the hyperplane, not probabilities. Platt scaling (sigmoid calibration) is needed to convert these to probability estimates.

Q21. Does SVM naturally output probabilities?

ANSWER: B

Explanation: SVMs work best with medium-sized datasets (100-10,000 samples) with clear margins, and are excellent for text classification, image classification, and bioinformatics where high-dimensional data is common.

Q22. What is SVM good for?

ANSWER: B

Explanation: SVR (Support Vector Regression) extends SVM to regression by using an ε-insensitive loss function. It predicts continuous values while maintaining the SVM's margin-based approach.

Q23. What is SVR (Support Vector Regression)?

ANSWER: B

Explanation: Epsilon (ε) defines the width of the 'tube' around predictions where errors are tolerated. Points within this tube have no penalty; points outside are penalized based on distance.

Q24. What is ε (epsilon) in SVR?

ANSWER: B

Explanation: One-class SVM learns the boundary of a single class, treating all training data as 'normal'. It's used for anomaly detection - new points outside the learned boundary are flagged as anomalies.

Q25. What is one-class SVM used for?

ANSWER: B

Explanation: One-class SVM is commonly used for anomaly/outlier detection, novelty detection, and situations where you have data from only one class (e.g., only normal equipment operation data, not failure data).

LEVEL 2: HYPERPARAMETER OPTIMIZATION

Q26. What are hyperparameters?

ANSWER: B

Explanation: Hyperparameters are set before training and control the learning process (e.g., learning rate, tree depth, C, gamma). They're not learned from data like model parameters.

Q27. What are model parameters?

ANSWER: B

Explanation: Model parameters (weights, coefficients) are learned from data during training through optimization. Hyperparameters are set beforehand and control how learning happens.

Q28. Examples of hyperparameters include:

ANSWER: B

Explanation: Common hyperparameters include: learning rate, number of estimators/trees, max depth, regularization strength (C, λ, α), batch size, number of layers/neurons, K in KNN, gamma in SVM.

Q29. What is Grid Search?

ANSWER: B

Explanation: Grid Search exhaustively tries all combinations of hyperparameter values in a predefined grid, guaranteeing the best combination within that grid will be found.

Q30. How does Grid Search work?

ANSWER: B

Explanation: Grid Search creates a Cartesian product of all hyperparameter values (e.g., learning_rate=[0.01, 0.1, 1.0] × max_depth=[3, 5, 7]) and evaluates every combination using cross-validation.

Q31. What is an advantage of Grid Search?

ANSWER: B

Explanation: Grid Search guarantees finding the optimal combination within the defined grid (complete coverage), is easily parallelizable, and requires no assumptions about hyperparameter relationships.

Q32. What is a disadvantage of Grid Search?

ANSWER: B

Explanation: Grid Search becomes exponentially expensive with more hyperparameters: 10 values each for 4 parameters = 10,000 combinations to evaluate, making it impractical for high-dimensional search.

Q33. If testing 10 values each for 3 hyperparameters, how many combinations?

ANSWER: B

Explanation: With 10 values per hyperparameter and 3 hyperparameters: 10 × 10 × 10 = 1,000 combinations. Each requires full cross-validation (e.g., 5-fold = 5 model trainings).

Q34. What is Random Search?

ANSWER: B

Explanation: Random Search samples hyperparameter combinations randomly from specified distributions rather than exhaustively trying all combinations.

Q35. How does Random Search compare to Grid Search?

ANSWER: B

Explanation: Random Search often finds better solutions than Grid Search with the same computational budget because it explores more unique values for each hyperparameter, especially important ones.

Q36. Why is Random Search effective?

ANSWER: B

Explanation: Often only a few hyperparameters truly matter for performance. Random Search tests more distinct values for these important hyperparameters while Grid Search wastes evaluations on less important ones.

Q37. How many hyperparameters favor Random Search over Grid?

ANSWER: B

Explanation: Random Search becomes preferable over Grid Search when you have 4+ hyperparameters, where Grid Search becomes prohibitively expensive and Random Search's broader exploration is more efficient.

Q38. What is Bayesian Optimization?

ANSWER: B

Explanation: Bayesian Optimization builds a probabilistic model (Gaussian Process) of the objective function and uses it to intelligently select the most promising hyperparameters to evaluate next.

Q39. How does Bayesian Optimization work?

ANSWER: B

Explanation: Bayesian Optimization iteratively: (1) builds a surrogate model of performance, (2) uses an acquisition function to select the most promising next point, (3) evaluates it, (4) updates the model, (5) repeats.

Q40. What is an advantage of Bayesian Optimization?

ANSWER: B

Explanation: Bayesian Optimization is the most sample-efficient method - it needs fewer evaluations to find good hyperparameters by learning from previous evaluations and focusing on promising regions.

Q41. When is Bayesian Optimization preferred?

ANSWER: B

Explanation: Use Bayesian Optimization when each model training is very expensive (hours per model), you can only afford 20-100 evaluations total, and sample efficiency is more important than simplicity.

Q42. What is the acquisition function in Bayesian Optimization?

ANSWER: B

Explanation: The acquisition function (e.g., Expected Improvement, Upper Confidence Bound) balances exploration (trying uncertain regions) and exploitation (trying regions likely to be good) to select the next hyperparameters.

Q43. What libraries provide Bayesian Optimization?

ANSWER: B

Explanation: Popular Bayesian Optimization libraries include Optuna (user-friendly), Hyperopt (established), and scikit-optimize (sklearn integration). Each implements Bayesian methods with different interfaces.

Q44. Should hyperparameter tuning use cross-validation?

ANSWER: B

Explanation: Yes, always use cross-validation within hyperparameter search to get reliable performance estimates. A single train/test split can give misleading results and overfit to that specific split.

Q45. What is nested cross-validation?

ANSWER: B

Explanation: Nested CV has an outer loop for unbiased performance estimation (testing) and an inner loop for hyperparameter selection (validation). This prevents hyperparameter tuning from contaminating performance estimates.

Q46. Why use nested CV?

ANSWER: B

Explanation: Nested CV provides truly unbiased estimates because for each outer fold: (1) hyperparameters are optimized only on that fold's training data, (2) tested on that fold's test data, (3) no information leakage between optimization and evaluation.

Q47. What is the search space in hyperparameter tuning?

ANSWER: B

Explanation: The search space defines the range and distribution of hyperparameters to explore, such as [0.001, 1.0] for learning rate or [1, 100] for number of trees. Design affects what solutions can be found.

Q48. How should learning rates be sampled?

ANSWER: B

Explanation: Learning rates should be sampled on log scale (e.g., 0.001, 0.01, 0.1, 1.0) because performance often varies logarithmically - 0.001 vs 0.01 is as important as 0.1 vs 1.0.

Q49. What is a good strategy for hyperparameter tuning?

ANSWER: B

Explanation: Start with a coarse, broad grid to identify promising regions (e.g., learning_rate=[0.001, 0.1, 10]), then narrow to a fine grid around the best values (e.g., [0.05, 0.075, 0.1, 0.125, 0.15]).

Q50. Should you tune all hyperparameters simultaneously?

ANSWER: B

Explanation: Start with the most impactful hyperparameters (learning rate, regularization strength, model capacity), get those roughly right, then fine-tune others. Tuning all simultaneously is expensive and unnecessary.

LEVEL 3: MODEL INTERPRETATION

Q51. Why is model interpretation important?

ANSWER: B

Explanation: Model interpretation builds trust with stakeholders, helps debug errors, ensures fairness/detects bias, satisfies regulatory requirements (e.g., GDPR's right to explanation), and guides feature engineering.

Q52. What is feature importance?

ANSWER: B

Explanation: Feature importance quantifies each feature's contribution to predictions, helping identify which variables matter most for the model's decisions.

Q53. How do tree-based models calculate feature importance?

ANSWER: B

Explanation: Tree-based models calculate importance by summing the total reduction in impurity (Gini or entropy) achieved by splits using that feature, weighted by the number of samples affected by those splits.

Q54. What is permutation importance?

ANSWER: B

Explanation: Permutation importance randomly shuffles a feature's values and measures how much model performance drops. Large drops indicate important features; small drops indicate unimportant ones.

Q55. What is an advantage of permutation importance?

ANSWER: B

Explanation: Permutation importance is model-agnostic - it works with any model (trees, neural nets, SVMs, etc.) by only measuring prediction changes, not relying on model-specific internals.

Q56. What is SHAP (SHapley Additive exPlanations)?

ANSWER: B

Explanation: SHAP (SHapley Additive exPlanations) uses game theory (Shapley values) to assign each feature a contribution value for a specific prediction, ensuring fair allocation of the prediction among features.

Q57. What does SHAP provide?

ANSWER: B

Explanation: SHAP provides both local explanations (why this specific prediction?) showing feature contributions for individual instances, and global importance (which features matter overall?) aggregated across the dataset.

Q58. What is a SHAP value?

ANSWER: B

Explanation: A SHAP value represents how much a feature's value contributes to pushing the prediction away from the base value (average prediction). Positive SHAP = increases prediction, negative SHAP = decreases it.

Q59. What is LIME (Local Interpretable Model-agnostic Explanations)?

ANSWER: B

Explanation: LIME (Local Interpretable Model-agnostic Explanations) explains individual predictions by fitting an interpretable linear model locally around that instance, showing which features influenced that specific prediction.

Q60. What are partial dependence plots (PDP)?

ANSWER: B

Explanation: Partial Dependence Plots (PDP) show the marginal effect of one or two features on predictions by averaging over all other features, visualizing the relationship between features and predictions.

Q61. What is a benefit of partial dependence plots?

ANSWER: B

Explanation: PDPs help visualize non-linear relationships (e.g., U-shaped effect of age on risk), identify thresholds (sharp changes at specific values), and understand feature effects while accounting for other variables.

Q62. What is Individual Conditional Expectation (ICE)?

ANSWER: B

Explanation: ICE (Individual Conditional Expectation) plots show the effect for each individual instance separately (disaggregated), revealing heterogeneous effects that PDPs (which average) might hide.

Q63. What are surrogate models?

ANSWER: B

Explanation: Surrogate models are simple, interpretable models (like linear regression or decision trees) trained to approximate a complex black-box model's predictions, making the black box's behavior easier to understand.

Q64. Why use surrogate models?

ANSWER: B

Explanation: By analyzing a simple surrogate model that mimics the complex model, we can understand which features matter, their relationships, and decision logic - insights difficult to extract from the original black box.

Q65. What is the interpretation-accuracy tradeoff?

ANSWER: B

Explanation: More interpretable models (linear, trees) are often less accurate than complex models (deep neural networks, ensembles). There's a tradeoff between explaining the model and maximizing predictive performance.

LEVEL 4: ML PROJECT WORKFLOW

Q66. What is the first step in an ML project?

ANSWER: B

Explanation: Every ML project starts by clearly defining the business problem, framing it as an ML task (classification/regression), and establishing success metrics that align with business objectives.

Q67. What should success metrics align with?

ANSWER: B

Explanation: Success metrics must align with business objectives to ensure the model optimizes for what actually matters. Technical metrics (accuracy) may not reflect true business value (revenue, customer satisfaction).

Q68. What is Exploratory Data Analysis (EDA)?

ANSWER: B

Explanation: EDA (Exploratory Data Analysis) involves understanding data through visualizations, statistics, and summaries to identify distributions, relationships, patterns, missing values, outliers, and data quality issues.

Q69. What should EDA include?

ANSWER: B

Explanation: Good EDA includes: distribution plots (histograms), correlation matrices, missing value patterns, outlier detection, target variable balance, feature relationships (scatter plots), and summary statistics (mean, median, std).

Q70. When should you split data into train/validation/test?

ANSWER: B

Explanation: Split data early (before any preprocessing or EDA involving the target) to prevent any test set information from influencing decisions. The test set must remain completely untouched until final evaluation.

Q71. What is a data pipeline?

ANSWER: B

Explanation: A data pipeline is an automated, reproducible sequence of data transformations (loading $\rightarrow$ cleaning $\rightarrow$ preprocessing $\rightarrow$ feature engineering) that ensures consistent processing across training and production.

Q72. What should be included in a pipeline?

ANSWER: B

Explanation: Pipelines should include: data loading, missing value imputation, outlier handling, feature scaling, categorical encoding, feature engineering, and feature selection - all preprocessing steps applied during training.

Q73. Why use sklearn.pipeline.Pipeline?

ANSWER: B

Explanation: sklearn.Pipeline ensures transformations are fit only on training data and applied consistently to train/validation/test/production data, preventing data leakage and ensuring reproducibility.

Q74. What is the benefit of pipelines?

ANSWER: B

Explanation: Pipelines ensure reproducibility (same transformations always applied consistently), prevent leakage (fit only on training), simplify deployment (single object to save/load), and reduce errors (fewer manual steps).

Q75. What comes after baseline model?

ANSWER: B

Explanation: After baseline, iteratively: try more complex models (ensemble methods, neural networks), engineer new features, tune hyperparameters, try different preprocessing, and compare results systematically.

Q76. Why start with baseline models?

ANSWER: B

Explanation: Baseline models (simple linear/logistic regression, basic decision tree) provide a reference point. They show whether added complexity actually improves performance, preventing over-engineering.

Q77. What should you track during experiments?

ANSWER: B

Explanation: Track: hyperparameters used, model type, performance metrics (train/validation/test), data versions, code versions/commits, feature sets, random seeds, training time, and any other experimental conditions.

Q78. What tools help with experiment tracking?

ANSWER: B

Explanation: Experiment tracking tools like MLflow, Weights & Biases (wandb), Neptune.ai automate logging of experiments, visualize results, compare runs, and ensure reproducibility.

Q79. When should you use the test set?

ANSWER: B

Explanation: Use the test set only once at the very end for final unbiased performance evaluation. Using it multiple times causes overfitting to test set - you'll tune decisions based on test performance.

Q80. Why is test set sacred?

ANSWER: B

Explanation: Using test set during development causes indirect overfitting - you make decisions (feature selection, model choice) influenced by test performance, making test results overly optimistic and unreliable.

Q81. What is model card documentation?

ANSWER: B

Explanation: Model cards are standardized documentation describing the model's intended use, training data, performance across groups, limitations, ethical considerations, and potential biases.

Q82. What should model documentation include?

ANSWER: B

Explanation: Documentation should include: dataset description, features used, algorithm/architecture, hyperparameters, performance metrics, limitations/failure modes, fairness analysis, intended use cases, maintenance plan.

Q83. What is technical debt in ML?

ANSWER: B

Explanation: Technical debt in ML refers to long-term costs from shortcuts: undocumented pipelines, hard-coded values, missing tests, data dependencies, monitoring gaps - making maintenance and updates difficult.

Q84. What causes ML technical debt?

ANSWER: B

Explanation: Common sources: undocumented preprocessing, scattered feature engineering code, no data versioning, missing model monitoring, hard-coded paths/values, lack of tests, unclear data dependencies.

Q85. How to minimize technical debt?

ANSWER: B

Explanation: Minimize debt through: version control (Git), automated testing, comprehensive documentation, modular code, data versioning (DVC), pipeline tools (Airflow), monitoring systems, and code reviews.

LEVEL 5: MODEL DEPLOYMENT & MONITORING

Q86. What is model deployment?

ANSWER: B

Explanation: Model deployment means making the trained model available in a production environment where it can receive inputs and generate predictions for real users/systems.

Q87. What are common deployment patterns?

ANSWER: B

Explanation: Common patterns: Batch prediction (periodic processing of many samples), Real-time API (on-demand individual predictions), Edge deployment (model runs on devices), Embedded systems (model in applications).

Q88. What is batch prediction?

ANSWER: B

Explanation: Batch prediction processes many samples together periodically (nightly, weekly) and stores results in a database. It's used when predictions don't need to be immediate.

Q89. When is batch prediction appropriate?

ANSWER: B

Explanation: Batch prediction is appropriate when: predictions can be made in advance (not real-time), you process many records together, you can tolerate latency (hours/days), like monthly customer churn scores.

Q90. What is real-time prediction?

ANSWER: B

Explanation: Real-time prediction provides on-demand predictions via API with low latency (milliseconds), responding immediately when a user/system makes a request.

Q91. What is a REST API for ML?

ANSWER: B

Explanation: A REST API exposes model predictions through HTTP endpoints (e.g., POST /predict), allowing applications to send feature data and receive predictions over the web using standard protocols.

Q92. What frameworks are used for ML APIs?

ANSWER: B

Explanation: Common frameworks: Flask and FastAPI (lightweight Python web frameworks), TensorFlow Serving (optimized for TensorFlow models), and cloud services (AWS SageMaker, Azure ML, Google AI Platform).

Q93. What is model monitoring?

ANSWER: B

Explanation: Model monitoring continuously tracks model performance metrics, prediction distributions, input data distributions, errors, and system health in production to detect issues early.

Q94. What should be monitored in production?

ANSWER: B

Explanation: Monitor: prediction distribution drift, input feature distributions, model performance metrics (accuracy, latency), error rates, data quality issues, system metrics (CPU, memory), business metrics (revenue impact).

Q95. What is data drift?

ANSWER: B

Explanation: Data drift occurs when the statistical properties of input features change over time (e.g., customer demographics shifting), potentially degrading model performance if the model was trained on different distributions.

Q96. What is concept drift?

ANSWER: B

Explanation: Concept drift occurs when the relationship between features and target changes over time (e.g., what makes a good investment changes with economic conditions), making the model's learned patterns obsolete.

Q97. How to handle drift?

ANSWER: B

Explanation: Handle drift through: scheduled retraining (monthly/quarterly), online learning (continuous updates), drift detection alerts (monitoring systems), adaptive algorithms, and fallback to simpler models when drift detected.

Q98. What is A/B testing in ML?

ANSWER: B

Explanation: A/B testing deploys the new model to a small percentage of traffic while the baseline model serves the rest, comparing real-world performance metrics before full rollout.

Q99. Why use A/B testing?

ANSWER: B

Explanation: A/B testing validates that the new model actually improves real-world metrics (user engagement, revenue, satisfaction) before risking full deployment. Lab performance doesn't always translate to production.

Q100. What are key considerations for ML in production?

ANSWER: B

Explanation: Key considerations: latency (response time), scalability (handle traffic spikes), reliability (uptime, failover), security (protect model and data), compliance (GDPR, regulations), cost (infrastructure), and maintainability.