PREDICTIVE ANALYTICS

Predictive Analytics all types of questions

Unit I

A. Predictive Analytics Basics

Q1. What is predictive analytics?
Predictive analytics is the branch of data analytics that builds models from historical data to estimate the probability or value of future outcomes (e.g., next month's sales, default risk, churn). It typically combines statistics, machine learning, and domain knowledge.

Q2. How is predictive analytics different from descriptive and prescriptive analytics?
Descriptive analytics summarizes what has happened (reports, dashboards). Predictive analytics estimates what is likely to happen next. Prescriptive analytics suggests what actions to take by optimizing decisions using predictions and business constraints.

Q3. Give three real-world examples of predictive analytics.
Examples: credit-risk scoring in banking; demand forecasting in retail; predicting equipment failure (predictive maintenance) in manufacturing; churn prediction in telecom; click-through rate prediction in online advertising.

Q4. What are the main steps in a predictive analytics project?
Typical steps: problem definition → data collection → data preprocessing/feature engineering → model selection and training → model evaluation and validation → deployment and monitoring.

B. Machine Learning and Its Types

Q5. What is machine learning?
Machine learning is a set of methods that allow computers to learn patterns from data and make predictions or decisions without being explicitly programmed with rules.

Q6. What are the main categories of machine learning?
The main categories are supervised learning (labeled outputs), unsupervised learning (no labels), semi-supervised learning (few labels + many unlabeled samples), and reinforcement learning (learning via rewards from interaction with an environment).

Q7. When would you choose supervised vs unsupervised learning?
Use supervised learning when you have labeled data and a clear target variable (price, class, probability). Use unsupervised learning when labels are not available and the goal is to discover structure, clusters, or lower-dimensional representations in the data.

Q8. What is reinforcement learning in one sentence?
Reinforcement learning trains an agent to take actions in an environment to maximize cumulative reward, using feedback (rewards/penalties) instead of labeled examples.

C. Supervised Learning

Q9. Define supervised learning.
Supervised learning is a type of ML where models are trained on input–output pairs (x,y)(*x*,*y*); the algorithm learns a mapping from features to target so it can predict y*y* for new x*x*.

Q10. What is the difference between regression and classification?
Regression predicts continuous numeric outputs (e.g., house price). Classification predicts discrete categories or labels (e.g., spam vs non-spam, disease vs no disease).

Q11. List common supervised learning algorithms.
Examples: linear regression, logistic regression, decision trees, random forests, gradient boosting (XGBoost, LightGBM), support vector machines, k-nearest neighbors, and neural networks.

Q12. What is overfitting and how can you reduce it in supervised learning?
Overfitting occurs when a model learns noise and performs well on training data but poorly on unseen data. It can be reduced by using more data, simplifying the model, regularization (L1/L2), pruning trees, early stopping, or cross-validation to tune hyperparameters.

Q13. What is cross-validation and why is it used?
Cross-validation (e.g., k-fold) repeatedly splits data into training and validation folds to estimate how well a model generalizes and to choose hyperparameters more robustly.

D. Unsupervised Learning

Q14. Define unsupervised learning.
Unsupervised learning learns patterns from data without target labels, focusing on discovering structure such as clusters, latent factors, or low-dimensional manifolds.

Q15. Name common unsupervised learning techniques.
Clustering (k-means, hierarchical, DBSCAN), dimensionality reduction (PCA, t-SNE, UMAP), association rule mining (Apriori, FP-growth), and anomaly detection methods.

Q16. What is clustering and give a use-case.
Clustering groups similar observations together; for example, segmenting customers into behavior-based groups for targeted marketing when labels such as “high value” are not predefined.

Q17. What is dimensionality reduction and why is it helpful?
Dimensionality reduction transforms high-dimensional data into fewer features while

preserving structure; it helps with visualization, noise reduction, and mitigating the curse of dimensionality.

E. Data Preprocessing Basics

Q18. What is data preprocessing and why is it crucial?
Data preprocessing consists of cleaning and transforming raw data into a suitable form for modeling (handling missing values, scaling, encoding, etc.). Good preprocessing often has more impact on model performance than trying many algorithms.

Q19. What are typical data quality problems you may face?
Common issues: missing values, duplicates, inconsistent formats, outliers, wrong data types, categorical values with spelling variations, and data leakage (using future information).

Q20. What is the difference between training, validation, and test sets?
Training data is used to fit the model; validation data is used for tuning hyperparameters and model selection; the test set is held out until the end to estimate true generalization performance.

F. Handling Missing Data

Q21. Why do missing values cause problems?
Many algorithms cannot handle missing values directly, and patterns in missingness can bias results if not treated carefully.

Q22. What are common strategies for handling missing data?
Options include deletion (dropping rows/columns with too many missing values), simple imputation (mean/median/mode), model-based imputation (k-NN, regression, MICE), or using indicator variables to mark missingness.

Q23. When is it dangerous to drop all rows with missing values?
If missingness is systematic (not random), dropping rows can bias the sample—for instance, removing all patients who skipped follow-ups may hide high-risk cases.

G. Feature Scaling and Transformation

Q24. Why is feature scaling important?
Many algorithms (k-NN, SVM, gradient descent-based models, clustering) are sensitive to feature magnitude; scaling ensures that features contribute comparably to distance or gradient calculations.

Q25. What are standardization and normalization?
Standardization transforms features to zero mean and unit variance. Normalization (min-max scaling) maps data to a fixed range, typically.

Q26. How would you handle highly skewed features?
Apply transformations such as logarithm, square root, or Box-Cox/Yeo-Johnson to reduce skewness, or use robust models less sensitive to distribution shape.

H. Encoding Categorical Variables

Q27. Why can't most ML algorithms use raw categorical text?
Algorithms expect numeric inputs; categorical texts must be converted into numerical representations without leaking order where none exists.

Q28. Explain one-hot encoding and when it is appropriate.
One-hot encoding creates a binary column for each category; it is suitable for low-cardinality nominal variables where the number of categories is manageable.

Q29. What is target/mean encoding and what is the risk?
Target encoding replaces categories with summary statistics of the target (e.g., average label). It can reduce dimensionality but risks target leakage and overfitting if not regularized with cross-validation or noise.

I. Outliers and Noise

Q30. What is an outlier and why does it matter?
An outlier is an observation far from the bulk of the data. Outliers can distort parameter estimates (e.g., in linear regression) and mislead models.

Q31. How can you detect outliers?
Approaches include visualizations (boxplots, scatter plots), z-scores, IQR rules, clustering-based anomaly scores, isolation forests, or domain-specific rules.

Q32. How might you treat outliers?
Options: verify and correct data entry errors, cap or winsorize extreme values, transform variables, use robust models, or explicitly label them as anomalies.

J. Feature Engineering

Q33. What is feature engineering?
Feature engineering creates new input variables from raw data (aggregations, ratios, domain-specific transformations) to help the model capture relevant patterns.

Q34. Give examples of feature engineering for a sales forecasting problem.
Examples: lag features (sales in previous weeks), rolling averages, day-of-week and holiday indicators, price difference vs competitor, promotion flags.

Q35. What is feature selection and why is it useful?
Feature selection chooses a subset of informative variables to reduce overfitting, improve

interpretability, and speed up training; methods include filter (correlation), wrapper (recursive feature elimination), and embedded (L1 regularization).

K. Evaluation Basics (Predictive Context)

Q36. What metrics are used for regression models?
Common metrics: MAE, MSE, RMSE, R2$R2$, and MAPE; choice depends on whether you care more about absolute error, squared error, or relative error.

Q37. What metrics are used for classification models?
Accuracy, precision, recall, F1-score, ROC-AUC, PR-AUC, and confusion matrix-based rates; choice depends on class imbalance and the cost of false positives vs false negatives.

Q38. Why is accuracy often misleading in imbalanced datasets?
If 99% of examples are negative, a model that always predicts negative has 99% accuracy but is useless for detecting the rare positive class.

L. Conceptual / HR-style Questions

Q39. Explain predictive analytics to a non-technical manager.
You could say: “We analyze historical data to learn patterns, then build models that estimate what is likely to happen next so the business can act earlier—for example, predicting which customers are likely to leave and targeting them with offers.”

Q40. How do you decide whether to use a simple model or a complex one?
Start with simple, interpretable models; only move to more complex models if they significantly improve validation performance and the gain justifies extra complexity, latency, and maintenance.

Q41. How do you ensure that a predictive model will generalize to future data?
Use proper train/validation/test splits, cross-validation, time-based splits for temporal data, avoid leakage, monitor performance after deployment, and retrain when data drifts.

Unit II

Concept and Types of Regression

Q1. What is regression in supervised learning?
Regression estimates a relationship between input variables (features) and a continuous output variable, allowing prediction of numeric values like price, demand, or temperature.

Q2. Distinguish between simple and multiple linear regression.
Simple linear regression uses one predictor and fits a line y=β0+β1x$y=\beta0+\beta1x$. Multiple linear regression uses several predictors: y=β0+β1x1+…+βpxp$y=\beta0+\beta1x1+…+\betapxp$.

Q3. What is polynomial regression and why is it used?
Polynomial regression models non-linear relationships by including powers of a feature, e.g. y=β0+β1x+β2x2*y=β0+β1x+β2x2*. It's still linear in parameters but non-linear in x*x*.

Q4. How is logistic regression different from linear regression?
Linear regression predicts continuous values and minimizes squared error. Logistic regression predicts probability of a binary class by modeling log-odds and applying the sigmoid function, usually optimized via cross-entropy loss.

Q5. Can logistic regression be used for multi-class problems? How?
Yes, using one-vs-rest (train a separate binary model for each class) or multinomial (softmax) logistic regression.

Ordinary Least Squares (OLS) and Assumptions

Q6. What is Ordinary Least Squares estimation?
OLS estimates regression coefficients by minimizing the sum of squared residuals ∑(yi−y^i)2∑(*yi*−*y*^*i*)2; the solution has a closed form β^=(X⊤X)−1X⊤y*β*^=(*X*⊤*X*)−1*X*⊤*y* when certain conditions hold.

Q7. What are the key assumptions of classical linear regression?
Linearity in parameters, independence of errors, homoscedasticity (constant variance), no perfect multicollinearity, and errors that are approximately normally distributed for valid inference.

Q8. What happens if these assumptions are violated?
Predictions can remain reasonable, but coefficient estimates may become biased or inefficient, standard errors and p-values become unreliable, and prediction intervals may be wrong.

Q9. What is multicollinearity and why is it a problem?
Multicollinearity occurs when predictors are highly correlated; it makes coefficient estimates unstable and inflates their variances, so small data changes can flip signs and magnitudes.

Q10. How can you detect and handle multicollinearity?
Detect via correlation matrix or Variance Inflation Factor (VIF). Handle by removing or combining correlated variables, using dimensionality reduction (PCA), or applying regularized regression (ridge, lasso).

Correlation

Q11. What does correlation measure in this context?
Correlation (usually Pearson) measures linear association between two variables, ranging from −1 (perfect negative) to +1 (perfect positive); 0 means no linear correlation.

Q12. Why is “correlation does not imply causation” important in regression?
Variables can be correlated due to confounders or coincidental patterns; building a regression model only on correlation does not prove one variable causes another.

Q13. How would you use a correlation matrix before modeling?
To identify strong relationships between features and the target, and to detect highly correlated features that might lead to multicollinearity.

Model Interpretation

Q14. How do you interpret the slope coefficient in simple linear regression?
β1*β*1 represents the expected change in y*y* for a one-unit increase in x*x*, holding other factors constant (in multiple regression, “holding others constant” applies).

Q15. What is the intercept and is it always meaningful?
The intercept β0*β*0 is the predicted value of y*y* when all predictors are zero; sometimes this scenario is outside the data range and lacks practical meaning.

Q16. In logistic regression, how do you interpret a coefficient?
A coefficient indicates the change in log-odds of the positive class for a one-unit increase in the predictor; exponentiating it gives the odds ratio.

Evaluation Metrics: MAE, MSE, RMSE, R²

Q17. Define Mean Absolute Error (MAE).
MAE is the average absolute difference between actual and predicted values:
MAE=1n∑ ∣ yi−y^i ∣ MAE=*n*1∑ ∣ *yi*−*y*^*i* ∣ ; it measures typical error magnitude in original units.

Q18. Define Mean Squared Error (MSE).
MSE is the average squared difference: MSE=1n∑(yi−y^i)2MSE=*n*1∑(*yi*−*y*^*i*)2; it penalizes large errors more strongly than MAE.

Q19. Define Root Mean Squared Error (RMSE).
RMSE is the square root of MSE; it has the same units as the target and emphasizes large errors due to the squaring.

Q20. What is R2*R*2 (coefficient of determination)?
R2=1−SSresSStot*R*2=1−SStotSSres shows the proportion of variance in the target explained by the model, ranging from 0 to 1 (and occasionally negative on test data if the model is worse than predicting the mean).

Q21. When might you prefer MAE over RMSE?
When you want a robust measure less sensitive to outliers and want a “typical” absolute error that is easy to interpret.

Q22. Why can two models with similar R2$R2$ have very different usefulness?
Because R2$R2$ ignores the distribution of errors and business cost; one model might be biased in critical ranges while overall variance explained looks similar.

Practical and Scenario-Based Questions

Q23. How do you choose between linear and polynomial regression for a dataset?
Plot the data and residuals: if patterns are clearly curved, try polynomial terms and compare via cross-validated error; avoid very high degree polynomials to prevent overfitting.

Q24. You fitted a regression model with very high training R2$R2$ but poor test performance. What is happening and what do you do?
Likely overfitting; simplify the model (fewer features, lower polynomial degree), use regularization, or collect more data and rely on cross-validation for model selection.

Q25. How would you evaluate a regression model properly?
Split into train/validation/test (or use k-fold CV), scale/transform features as needed, train on train, tune hyperparameters on validation using metrics like RMSE/MAE, and report performance on the unseen test set.

Q26. In a business context, how do you decide which error metric to optimize?
Map prediction errors to business cost: if large errors are very costly, use MSE/RMSE; if all errors are equally costly per unit, MAE may be better; sometimes custom loss functions reflect asymmetric penalties.

Q27. Why is logistic regression considered a classification algorithm even though it has "regression" in the name?
Because it outputs probabilities of class membership and is used to classify instances, not to predict continuous numeric values; "regression" refers to modeling log-odds as a linear function of features.

Q28. What is regularization and how does it relate to regression?
Regularization (e.g., L1 in lasso, L2 in ridge) adds a penalty on large coefficients to the loss function; it shrinks weights, reduces variance, and helps combat multicollinearity and overfitting.

Q29. How do you check if your residuals satisfy linear regression assumptions?
Plot residuals vs fitted values (look for randomness), check histogram/QQ-plot of residuals (approximate normality), and inspect residuals against each predictor for patterns or heteroscedasticity.

Q30. In logistic regression, which evaluation metrics replace MAE/MSE?
Common metrics are accuracy, precision, recall, F1, ROC-AUC, log-loss, and confusion matrix; choice depends on class imbalance and error costs.

Concepts and Algorithms

Q1. What is classification in supervised learning?
Classification learns a mapping from features to discrete class labels using labeled examples, then assigns each new sample to one of the predefined classes.

Q2. What is “lazy learning” and how does k-nearest neighbors (k-NN) fit in?
Lazy learning postpones generalization until query time; k-NN stores the training set and, for each new point, finds the k closest training points (by a distance metric) and predicts the majority class among them.

Q3. What are the advantages and disadvantages of k-NN?
Advantages: simple, non-parametric, works well with enough data and a good distance metric. Disadvantages: expensive at prediction time, sensitive to feature scaling and irrelevant features, performance degrades in high dimensions.

Q4. What is Naïve Bayes and why is it called “naïve”?
Naïve Bayes is a probabilistic classifier based on Bayes’ theorem, assuming features are conditionally independent given the class. The “naïve” assumption simplifies computation but is rarely exactly true.

Q5. Give an example where Naïve Bayes is effective.
Text classification (spam detection, sentiment) often works well with multinomial Naïve Bayes because word occurrences given topic are reasonably independent and high-dimensional.

Q6. How do decision trees classify data?
Decision trees recursively split the feature space based on conditions that maximize impurity reduction (e.g., Gini, entropy), forming a tree where each leaf corresponds to a class prediction.

Q7. What are key strengths and weaknesses of decision trees?
Strengths: interpretability, handling mixed data types, nonlinear boundaries, little preprocessing. Weaknesses: tendency to overfit, high variance, unstable to small data changes.

Q8. What is a rule-based classifier?
Rule-based classifiers use explicit IF–THEN rules derived from data or expert knowledge (e.g., “IF age < 25 AND income low THEN risk = high”), often extracted from decision trees or specialized algorithms.

Q9. What is a support vector machine (SVM)?
SVM finds a decision boundary (hyperplane) that maximizes the margin between classes; with kernels (RBF, polynomial), it can learn nonlinear boundaries in a transformed feature space.

Q10. When is SVM a good choice?
SVMs work well on medium-sized, high-dimensional datasets (e.g., text, bioinformatics) where a clear margin exists; they can struggle with very large datasets and need careful kernel and parameter tuning.

Evaluation Metrics and Confusion Matrix

Q11. What is a confusion matrix?
For binary classification, it is a 2×2 table of counts: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), summarizing prediction outcomes.

Q12. Define accuracy and when it is misleading.
Accuracy = (TP+TN)/(TP+TN+FP+FN)$(TP+TN)/(TP+TN+FP+FN)$. It is misleading on imbalanced data (e.g., rare fraud) because a model that predicts the majority class only can have high accuracy but zero usefulness on the minority class.

Q13. Define precision and recall.
Precision = TP/(TP+FP)$TP/(TP+FP)$ measures correctness among predicted positives.
Recall (sensitivity) = TP/(TP+FN)$TP/(TP+FN)$ measures how many actual positives are captured.

Q14. What is the F1 score and why use it?
F1 is the harmonic mean of precision and recall:
F1=2 · precision · recallprecision+recall$F1=2 \cdot \frac{precision \cdot recall}{precision+recall}$. It balances both metrics and is useful on imbalanced datasets.

Q15. What is ROC-AUC?
The ROC curve plots true positive rate vs false positive rate for varying thresholds; AUC is the area under this curve, representing the probability that the classifier ranks a random positive higher than a random negative.

Q16. What is logarithmic loss (log-loss) in classification?
Log-loss measures the quality of predicted probabilities: it penalizes confident wrong predictions heavily; lower log-loss indicates better calibrated, more accurate probability estimates.

Q17. When would you optimize for precision vs recall?
Optimize precision when false positives are very costly (e.g., flagging legitimate transactions as fraud). Optimize recall when missing positives is worse (e.g., disease screening), even at the cost of more false alarms.

Q18. How do you choose the classification threshold for a probability output model?
Use validation data to analyze precision-recall or ROC curves and pick a threshold that balances business trade-offs (e.g., maximize F1, or meet a minimum recall while keeping FP below a limit).

Q19. Explain macro vs micro-averaged precision/recall in multi-class.
Macro averages metrics per class then averages across classes (treats classes equally). Micro aggregates TP, FP, FN across all classes first (weights by class frequency).

Scenario-Based Questions

Q20. You have highly imbalanced data (1% positive). Which metrics and methods do you use?
Use precision, recall, F1, PR-AUC; apply techniques like class weighting, resampling (SMOTE, undersampling), threshold tuning, or anomaly-detection approaches instead of relying on accuracy.

Q21. How do you pick k in k-NN?
Use cross-validation: small k may overfit (noisy boundary), large k may oversmooth. Try a range of odd values, choose the one with best validation performance.

Q22. What preprocessing does k-NN or SVM require?
Scale features (standardization/normalization), remove or engineer features to reduce noise, sometimes reduce dimensionality (PCA) to combat the curse of dimensionality.

Q23. Why can Naïve Bayes perform well even when independence is violated?
Even if independence is not exact, the model's decision boundary can still approximate the Bayes optimal classifier; errors in joint probability often cancel out in class comparisons.

Q24. How would you prevent a decision tree from overfitting?
Set maximum depth, minimum samples per split/leaf, use pruning based on validation error, or switch to ensemble methods like random forests/gradient boosting.

Q25. Compare decision trees and SVMs in terms of interpretability and performance.
Decision trees are easy to interpret but can be less accurate and unstable; SVMs often yield higher accuracy and robust margins but are less interpretable and more complex to tune.

Unit IV

K-Means Clustering and Intuition

Q1. What is K-means clustering?
K-means partitions data into K*K* clusters by assigning each point to the nearest cluster centroid and then recomputing centroids as the mean of assigned points; it iterates until assignments stabilize.

Q2. What is the main objective function in K-means?
To minimize the sum of squared distances between each point and its assigned cluster centroid (within-cluster sum of squares).

Q3. What assumptions does K-means implicitly make about clusters?
Clusters are roughly spherical, of similar size/density, and separable in Euclidean space; it works best when these assumptions approximately hold.

Q4. Why should features be scaled before K-means?
Because Euclidean distance is sensitive to scale; without scaling, features with larger numeric ranges dominate the distance calculation and distort clusters.

Random Initialization Trap and Improvements

Q5. What is the random initialization trap in K-means?
Choosing poor random initial centroids can lead the algorithm to a bad local minimum, giving different and sub-optimal clustering each run.

Q6. How can you mitigate the random initialization problem?
Use K-means++ initialization (spreads centroids apart), run K-means multiple times with different seeds and choose the best objective value, or use more robust clustering methods.

Q7. What is K-means++?
K-means++ initializes the first centroid randomly, then chooses subsequent centroids with probability proportional to the squared distance from existing centroids, improving convergence and solution quality.

Choosing the Number of Clusters

Q8. Why is selecting the right number of clusters K*K* important?
Too few clusters merge distinct groups; too many over-segment and capture noise. Both hurt interpretability and may reduce downstream performance.

Q9. What is the elbow method?
Plot the within-cluster sum of squares versus K*K*; the “elbow” point where marginal improvement sharply decreases is a good candidate for K*K*.

Q10. What is the silhouette score and how is it interpreted?
For each point, the silhouette compares cohesion (distance to its own cluster) vs separation (distance to nearest other cluster); values near +1 indicate well-clustered points, 0 overlaps, negative possible mis-assignment. Average silhouette across points can guide K*K*.

Q11. Name other practical considerations for choosing K*K*.
Domain knowledge (expected number of segments), business constraints (manageable number of groups), cluster stability across random seeds, and visualization.

Hierarchical Clustering

Q12. What is hierarchical clustering?
Hierarchical clustering produces a tree (dendrogram) of nested clusters by either repeatedly

merging smaller clusters (agglomerative) or splitting a big cluster (divisive).

Q13. Distinguish between agglomerative and divisive clustering.
Agglomerative starts with each point as its own cluster and merges them step by step; divisive starts with one cluster containing all points and recursively splits it.

Q14. What is a dendrogram and how is it used?
A dendrogram is a tree diagram showing how clusters are merged or split; cutting it at a chosen height yields a specific number of clusters based on distance thresholds.

Linkage Types

Q15. What is linkage in hierarchical clustering?
Linkage defines how distance between clusters is computed when deciding which clusters to merge.

Q16. Explain single, complete, average, and centroid linkage.

- Single: distance between the closest pair of points (one from each cluster).
- Complete: distance between the farthest pair of points.
- Average: average distance over all point pairs.
- Centroid: distance between cluster centroids.

Q17. When might single linkage be problematic?
It can suffer from the “chaining” effect, causing elongated clusters that connect distant points through sequences of close points.

Q18. Why is complete linkage often preferred in practice?
It tends to form more compact, spherical clusters and is less susceptible to chaining, though it can be sensitive to outliers.

Association Rules and Pattern Mining

Q19. What is an association rule?
An association rule has the form A⇒B$A \Rightarrow B$, meaning that when items in set AA appear, items in set BB tend to appear in the same transaction.

Q20. Define support, confidence, and lift.

- Support: proportion of transactions containing both AA and BB.
- Confidence: P(B ∣ A)$P(B \mid A)$ = support(A and B) / support(A).
- Lift: confidence(A→B) / support(B); lift > 1 indicates positive association.

Q21. What is market basket analysis?
It applies association rules to retail transaction data to discover which products are frequently bought together, informing store layout, promotions, and cross-selling.

Q22. Name common algorithms for mining association rules.
Apriori and FP-growth are classic algorithms that find frequent itemsets and then derive rules meeting minimum support and confidence thresholds.

Q23. Why is the Apriori algorithm efficient despite considering many combinations?
It uses the Apriori property: any superset of an infrequent itemset is also infrequent, so large parts of the search space can be pruned early.

Finding Patterns and Practical Considerations

Q24. How does clustering help in pattern detection?
Clustering reveals groups with similar characteristics (e.g., customer segments), which can be interpreted for targeted strategies or as features for other models.

Q25. How can you evaluate the usefulness of clusters for business?
Check whether clusters differ meaningfully in key metrics (revenue, churn, risk), whether stakeholders can describe them intuitively, and whether actions can be tailored to each group.

Q26. What are common pitfalls in association rule mining?
Generating too many trivial rules, misinterpreting correlation as causation, setting thresholds too low or high, and ignoring business context.

Q27. How would you reduce the number of association rules to a manageable set?
Raise support/confidence thresholds, filter by lift or conviction, limit rule length, and focus on rules involving high-value items or categories of interest.

Unit V

Dimensionality Reduction & PCA

Q1. What is dimensionality reduction and why is it useful?
Dimensionality reduction transforms data with many features into a lower-dimensional representation that preserves as much important information (variance or structure) as possible. It helps combat the curse of dimensionality, reduces noise, speeds up training, and improves visualization.

Q2. Distinguish feature selection from feature extraction.
Feature selection chooses a subset of original features (e.g., via correlation or mutual information), leaving them unchanged. Feature extraction creates new features as combinations of originals (e.g., PCA components, autoencoder bottleneck features).

Q3. What is Principal Component Analysis (PCA) conceptually?
PCA finds orthogonal directions (principal components) in feature space along which the data

variance is maximized; projecting data onto the top k*k* components gives a lower-dimensional representation that captures most of the variability.

Q4. What mathematical objects does PCA use?
PCA computes the covariance matrix of standardized features and then its eigenvalues/eigenvectors, or equivalently performs singular value decomposition (SVD) of the data matrix; eigenvectors define components, eigenvalues give their explained variances.

Q5. How do you choose the number of PCA components?
Use the cumulative explained variance plot and choose the smallest k*k* such that, for example, 90–95% of variance is retained, or apply cross-validation on downstream model performance.

Q6. What assumptions and limitations does PCA have?
PCA assumes linear relationships, focuses on directions of maximal variance (which may not correspond to the most predictive features), and can be distorted by unscaled or heavily skewed features and outliers.

Feedforward Neural Networks and MLP

Q7. What is a feedforward neural network?
It is a directed acyclic network of layers where information flows from inputs through one or more hidden layers to outputs; each layer applies a linear transformation followed by a nonlinear activation (e.g., ReLU, sigmoid).

Q8. Describe a multi-layer perceptron (MLP).
An MLP is a standard fully connected feedforward network with an input layer, one or more hidden layers of neurons, and an output layer; every neuron in a layer connects to all neurons in the next layer, making it a universal function approximator for many tasks.

Q9. Why are nonlinear activation functions necessary?
Without nonlinear activations, stacking multiple linear layers collapses into a single linear transformation, so the network cannot model complex nonlinear relationships; nonlinearities (ReLU, tanh, etc.) allow composition of features.

Q10. What is backpropagation?
Backpropagation efficiently computes gradients of the loss with respect to all weights by applying the chain rule from output to input. These gradients are then used by optimization algorithms (like gradient descent, Adam) to update weights.

Q11. What is overfitting in neural networks and how can you reduce it?
Overfitting occurs when the network memorizes training data instead of learning general patterns. Techniques to reduce it include more data, regularization (L2, dropout), early stopping, data augmentation, and simpler architectures.

Convolutional Neural Networks (CNN)

Q12. What problem do CNNs solve better than MLPs?
CNNs are designed for grid-like data such as images; they exploit local spatial structure and parameter sharing to learn translation-invariant features (edges, textures) more efficiently than dense MLPs.

Q13. What is a convolution layer and why is weight sharing important?
A convolution layer applies learnable filters (kernels) that slide over the input and compute dot products with local patches; the same filter weights are reused across all spatial locations, drastically reducing parameters and capturing repeated patterns.

Q14. What roles do pooling layers play in CNNs?
Pooling (e.g., max or average pooling) reduces spatial dimensions, aggregates local features, and introduces invariance to small translations or distortions, which helps generalization and reduces computation.

Q15. Provide a typical CNN architecture flow for image classification.
Input image → stack of [Conv + activation + (optional) pooling] blocks → flatten feature maps → one or more dense layers → output layer with softmax for class probabilities.

Recurrent Neural Networks (RNN)

Q16. When are recurrent neural networks used?
RNNs handle sequential data (time series, text, speech) by maintaining a hidden state that summarizes previous inputs, enabling modeling of temporal dependencies.

Q17. Describe the basic RNN update equation.
At time step tt: hidden ht=f(Wxxt+Whht−1+b)$ht=f(Wxxt+Whht−1+b)$; output ytyt depends on htht. Here ff is a nonlinear activation like tanh or ReLU.

Q18. What problems do vanilla RNNs suffer from, and how do LSTM/GRU address them?
Vanilla RNNs struggle with vanishing and exploding gradients, making learning long-range dependencies difficult. LSTMs and GRUs use gating mechanisms (input, forget, output/update gates) to control information flow and preserve gradients over longer sequences.

Q19. Name three applications of RNN-style models.
Language modeling and text generation, machine translation, sentiment analysis, speech recognition, and sequence-to-sequence prediction in time series.

Comparisons and Practical Questions

Q20. How would you decide between using PCA vs an autoencoder for dimensionality reduction?
PCA is linear, simple, and interpretable; it works well as a first step. Autoencoders (neural

networks trained to reconstruct input through a bottleneck) can learn nonlinear representations and may capture complex manifolds but are more complex to train and tune.

Q21. Compare MLP, CNN, and RNN in terms of data type and structure.
MLP: general-purpose, works on fixed-length feature vectors.
CNN: best for spatially structured data (images, 2D/3D grids).
RNN: best for ordered sequences where order and context over time matter.

Q22. What hyperparameters are most important when training deep networks?
Learning rate (and schedule), batch size, number of layers and units, choice of activation, regularization strength, dropout rate, optimizer (SGD, Adam), and number of training epochs.

Q23. How does batch normalization help training?
Batch norm normalizes layer inputs within a mini-batch to have controlled mean and variance, reducing internal covariate shift, allowing higher learning rates, and often improving convergence and generalization.

Q24. What is the vanishing gradient problem in deep networks?
During backpropagation, gradients can shrink exponentially as they are multiplied through many layers, causing earlier layers to learn extremely slowly; ReLU activations, residual connections, and careful initialization help mitigate this.

Q25. In an interview, how would you briefly explain a neural network to a non-technical stakeholder?
You might say: “A neural network is a flexible mathematical model that learns to map inputs to outputs by adjusting many simple interconnected units, similar to how the brain strengthens connections, so that over time it can recognize patterns like images or speech and make predictions.”

Unit VI

Bias–Variance Trade-off

Q1. What is the bias–variance trade-off?
Bias is error from overly simple assumptions (underfitting); variance is error from sensitivity to small changes in training data (overfitting). Increasing model complexity usually decreases bias but increases variance, so a good model balances both to minimize total generalization error.

Q2. How can you detect high bias vs high variance from learning curves?
High bias: both training and validation errors are high and close to each other even with more data. High variance: training error is low but validation error is much higher; adding more data often helps.

Q3. Give examples of high-bias and high-variance models.
High-bias: linear regression on a strongly nonlinear problem, shallow decision tree.

High-variance: deep unpruned decision tree, k-NN with k=1k=1, very deep neural network with little data.

Q4. What techniques help reduce variance without increasing bias too much?
Use more training data, regularization (L1/L2, dropout), model simplification, early stopping, and ensemble methods like bagging/random forests which average many high-variance models.

Cross-Validation: LOO and K-fold

Q5. Why is cross-validation needed?
A single train/test split may give a noisy estimate of performance. Cross-validation repeatedly splits the data, trains on each training fold, and evaluates on the corresponding validation fold to get a more reliable estimate and tune hyperparameters.

Q6. What is leave-one-out (LOO) cross-validation?
For nn samples, LOO trains the model nn times, each time using n−1n−1 samples for training and 1 sample as validation. The average of the nn validation losses estimates generalization error.

Q7. Advantages and disadvantages of LOO?
Advantages: nearly unbiased estimate of performance and uses almost all data for training each time. Disadvantages: very expensive for complex models and high-variance estimates for unstable learners (small change in data can change model a lot).

Q8. What is K-fold cross-validation?
Data is split into KK roughly equal folds. For each of KK runs, one fold is used as validation and the remaining K−1K−1 as training. The mean validation metric over all folds is used for model comparison or tuning.

Q9. How do you choose K in K-fold CV?
Typical choices are K=5K=5 or 10 as a balance between computational cost and variance of the estimate. Smaller K (e.g., 3) is faster but noisier; larger K approaches LOO with higher cost.

Q10. What is stratified K-fold and when is it important?
Stratified K-fold preserves the class distribution in each fold (e.g., same proportion of positives/negatives as the full dataset), which is crucial for classification with class imbalance.

Bagging (Bootstrap Aggregating)

Q11. What is bagging and what problem does it address?
Bagging trains multiple models on different bootstrap samples (sampling with replacement) of the training data and averages their predictions (or takes majority vote). It primarily reduces variance, stabilizing high-variance models like decision trees.

Q12. Why does averaging predictions reduce variance?
Each model trained on a bootstrap sample makes different errors; averaging independent (or weakly correlated) errors tends to cancel them out, leading to more stable predictions.

Q13. How is a bootstrap sample constructed?
From a dataset of size n*n*, sample n*n* observations with replacement. Some original samples appear multiple times and some not at all; on average, about 63% of unique observations appear in each bootstrap sample.

Q14. What are out-of-bag (OOB) samples and how are they used?
OOB samples for a model are training examples not included in that model's bootstrap sample. Because each tree in a bagging ensemble sees only part of the data, the remaining OOB data can act as a validation set to estimate generalization error without a separate hold-out set.

Boosting

Q15. What is boosting in general terms?
Boosting trains a sequence of weak learners (often shallow trees), where each new learner focuses more on examples that previous learners predicted poorly. Their weighted predictions are combined to form a strong ensemble.

Q16. How is boosting different from bagging?
Bagging trains models independently and in parallel on bootstrap samples, averaging their outputs mainly to reduce variance. Boosting trains models sequentially, with each model depending on the previous ones, reducing both bias and variance but with higher risk of overfitting and more sensitivity to noise.

Q17. Briefly explain the idea of AdaBoost.
AdaBoost starts with equal weights on all training samples. After each weak learner is trained, it increases weights of misclassified samples and decreases weights of correctly classified ones. The next learner is trained on this reweighted data, and final predictions are a weighted majority vote of all learners.

Q18. What about gradient boosting (e.g., XGBoost, LightGBM)?
Gradient boosting views boosting as gradient descent in function space: each new tree is trained to approximate the negative gradient (residual errors) of the loss with respect to the current model's predictions. It supports flexible loss functions and regularization, making it extremely powerful on tabular data.

Q19. How do you control overfitting in boosting algorithms?
Use shallow trees (low depth), learning rate (shrinkage) to scale each tree's contribution, limit number of trees, subsample rows/columns, and use early stopping based on validation performance.

Random Forests

Q20. What is a random forest?
A random forest is an ensemble of decision trees built using bagging plus random feature selection at each split. For prediction, it averages (regression) or votes (classification) across trees.

Q21. Why does random feature selection at splits help?
By restricting each split to a random subset of features, trees become more decorrelated; this reduces the chance that many trees make the same error, further lowering ensemble variance.

Q22. What are key hyperparameters in random forests?
Number of trees, depth of each tree (or min samples per leaf), number of features considered at each split (e.g., dd for classification, d/3d/3 for regression), and minimum samples to split nodes.

Q23. How do random forests handle missing values and feature scaling?
Decision trees (and thus random forests) are relatively robust to unscaled features and can handle missing values via surrogate splits or by sending missing data down both branches with weights (implementation-dependent). However, it is still good practice to handle missing values explicitly.

Q24. What are strengths and weaknesses of random forests?
Strengths: strong performance out of the box, robust to noise and outliers, can capture nonlinearities and interactions, provide variable importance measures. Weaknesses: less interpretable than single trees, can be slow on very large datasets, and model size can be large.

Putting It Together – Scenario Questions

Q25. You have a high-variance decision tree with poor test performance. What steps would you take?
First, try pruning or limiting depth and using cross-validation for tuning. Then, consider bagging or random forests to average multiple trees. Finally, examine feature importance and noise, and possibly gather more data.

Q26. How would you compare a random forest vs gradient boosting model fairly?
Use the same train/validation/test splits or K-fold cross-validation, tune hyperparameters for each model via CV, and compare metrics (e.g., ROC-AUC, RMSE) on the held-out test set; also consider training time, inference speed, and interpretability.

Q27. In an imbalanced classification problem, how do you combine cross-validation with performance metrics?
Use stratified K-fold CV to preserve class ratios and evaluate metrics sensitive to imbalance

(precision, recall, F1, ROC-AUC or PR-AUC) rather than accuracy. Average metrics across folds for stability.

Q28. Explain how bias–variance considerations influence your choice between a linear model, random forest, and gradient boosting.
A linear model has high bias/low variance, good for simple relationships and small data. A random forest has lower bias and moderate variance, robust with minimal tuning. Gradient boosting often achieves lowest bias but can have higher variance and overfitting risk, requiring careful regularization and validation.