1. Define Artificial Intelligence (AI)

AI = Machines that think and act like humans. They can understand, learn, and make decisions like humans without being explicitly programmed for every task. Goal: Mimic human intelligence.

Example: Siri, Google Assistant, ChatGPT.

2. Explain the differences between Artificial Intelligence (AI), Machine Learning (ML), Deep Learning (DL), and Data Science (DS)

AI (Artificial Intelligence): The broadest concept. Machines mimicking human intelligence.

ML (Machine Learning): A subset of AI. Systems that learn patterns from data and make predictions.

DL (Deep Learning): A subset of ML. Uses artificial neural networks to process large and complex data (like images, videos).

DS (Data Science): A field that uses tools, algorithms, and ML to extract knowledge from data and solve real-world problems.

3. How does AI differ from traditional software development

Traditional software follows fixed rules written by developers. AI systems, especially ML-based, learn patterns from data and improve over time. Traditional = Rule-based. AI = Data-based + self-improving.

4. Provide examples of AI, ML, DL, and DS applications

AI: Self-driving cars, smart assistants.

ML: Spam filters, recommendation engines.

DL: Face recognition, voice assistants.

DS: Business dashboards, churn prediction models.

5. Discuss the importance of AI, ML, DL, and DS in today's world

These technologies automate tasks, improve decision-making, personalize user experiences, and drive innovation in every industry — from healthcare to e-commerce. They are the backbone of today’s digital transformation.

6. What is Supervised Learning

A type of ML where models are trained on labeled data — input is given with the correct output. The model learns from this to predict future outputs.

7. Provide examples of Supervised Learning algorithms

Linear Regression, Logistic Regression, Decision Trees, Random Forest, SVM (Support Vector Machine), K-Nearest Neighbors (KNN).

8. Explain the process of Supervised Learning

Collect and clean labeled data.

Split the data into training and testing sets.

Choose a suitable algorithm.

Train the model on training data.

Evaluate using test data.

Improve and deploy.

9. What are the characteristics of Unsupervised Learning

No labeled data is provided.

Model finds hidden patterns or groups in the data.

Used mainly for clustering or dimensionality reduction.

No correct output is given during training.

10. Give examples of Unsupervised Learning algorithms

K-Means, DBSCAN, Hierarchical Clustering, PCA (Principal Component Analysis), t-SNE.

11. Describe Semi-Supervised Learning and its significance

Uses a small amount of labeled data with a large amount of unlabeled data. It’s useful when labeling data is expensive or time-consuming. It combines the benefits of supervised and unsupervised learning.

12. Explain Reinforcement Learning and its applications

Reinforcement Learning is where an agent learns by interacting with its environment and receiving rewards or penalties. It learns what actions yield the most reward over time.

Examples: Game AI (chess, Go), robotics, self-driving cars.

13. How does Reinforcement Learning differ from Supervised and Unsupervised Learning

Supervised: Learns from labeled data.

Unsupervised: Learns from unlabeled data.

Reinforcement: Learns from reward signals after performing actions.

14. What is the purpose of the Train-Test-Validation split in machine learning

To train the model, tune its hyperparameters, and test it fairly. This ensures the model performs well on unseen data and avoids overfitting.

15. Explain the significance of the training set

This is the core data the model learns from. A well-prepared training set helps the model understand the patterns and relationships in the data.

16. How do you determine the size of the training, testing, and validation sets

Common splits:

70% train, 15% validation, 15% test

80% train, 10% validation, 10% test Split size depends on data availability and project needs.

17. What are the consequences of improper Train-Test-Validation splits

Overfitting or underfitting

Biased performance metrics

Poor generalization to real-world data

Wasted resources during tuning and testing

18. Discuss the trade-offs in selecting appropriate split ratios

More training data = better learning

Less test data = less reliable performance metrics

Balanced split is key to good evaluation and generalization

19. Define model performance in machine learning

Model performance tells how well the model makes correct predictions. It reflects accuracy, reliability, and usefulness in real-world situations.

20. How do you measure the performance of a machine learning model

For classification:

Accuracy

Precision

Recall

F1 Score

ROC-AUC

For regression:

MAE (Mean Absolute Error)

MSE (Mean Squared Error)

RMSE (Root Mean Squared Error)

R^2 Score

21. What is overfitting and why is it problematic:

Overfitting happens when a model learns the training data too well — including noise and irrelevant details. It performs well on training data but poorly on new, unseen data. Problem: Poor generalization = bad real-world performance.

22. Techniques to address overfitting:

Cross-validation

Reduce model complexity (simpler algorithms)

Prune decision trees

Regularization (L1, L2)

Dropout (for neural networks)

Early stopping

More training data

Data augmentation

23. What is underfitting and its implications:

Underfitting occurs when a model is too simple to learn the underlying patterns of the data. Implications: Poor performance on both training and testing sets — the model fails to capture the signal in the data.

24. How to prevent underfitting:

Use a more complex model

Train longer (more epochs)

Reduce regularization

Provide more relevant features

Use better feature engineering

Tune hyperparameters properly

25. Balance between bias and variance in model performance:

Bias: Error due to overly simplistic models (underfitting)

Variance: Error due to too complex models (overfitting) A good model finds the sweet spot — low bias + low variance — also called the bias-variance trade-off.

26. Common techniques to handle missing data:

Remove rows or columns with missing values

Imputation (mean, median, mode)

KNN imputation

Use ML models to predict missing values

Use indicators (e.g., "Missing" as a category)

27. Implications of ignoring missing data:

Biased or inaccurate models

Loss of valuable data

Reduced model performance

Can lead to wrong conclusions, especially in critical domains (healthcare, finance)

28. Pros and cons of imputation methods:

Pros:

Keeps the dataset size intact

Simple methods are easy to apply

Can improve model performance

Cons:

May introduce bias

Imputation might distort real data distribution

Complex methods (like ML-based imputation) are time-consuming

29. How does missing data affect model performance:

Missing data can lead to inaccurate models, biased predictions, and reduced overall performance. Some models may even fail to train if essential features are missing.

30. Define imbalanced data in machine learning:

Imbalanced data occurs when the distribution of classes in a classification problem is uneven — one class has significantly more samples than the other(s). Example: 95% no-disease, 5% disease.

31. Challenges posed by imbalanced data:

Biased models that favor the majority class

Poor performance on minority class

Misleading accuracy scores

Model might ignore rare but important cases

32. Techniques to handle imbalanced data:

Resampling (up-sampling, down-sampling)

SMOTE (Synthetic Minority Oversampling Technique)

Using different evaluation metrics (Precision, Recall, F1 Score, ROC-AUC)

Cost-sensitive learning

Ensemble methods (like Balanced Random Forest)

33. Up-sampling and Down-sampling:

Up-sampling: Increase minority class samples by duplication or synthetic generation

Down-sampling: Reduce majority class samples by randomly removing data points

34. When to use up-sampling vs down-sampling:

Use up-sampling when you have limited data and don't want to lose majority class info

Use down-sampling when you have enough data and want faster computation with balance

35. What is SMOTE and how does it work:

SMOTE (Synthetic Minority Oversampling Technique) generates synthetic samples for the minority class by interpolating between existing minority samples. It avoids duplication and improves model learning.

36. Role of SMOTE in imbalanced data:

SMOTE balances the dataset, improves recall for minority class, and helps models learn better without overfitting like random duplication might cause.

37. SMOTE – Advantages and Limitations:

Advantages:

Better generalization

Reduces bias toward majority class

Limitations:

Risk of overfitting

Doesn't consider class boundaries

Not ideal for high-dimensional sparse data

38. Examples where SMOTE is beneficial:

Medical diagnosis (rare diseases)

Fraud detection

Spam detection

Churn prediction

39. Define data interpolation and its purpose:

Data interpolation estimates unknown values between two known values. It is used to fill missing data, smooth time series, or reconstruct signals.

40. Common methods of data interpolation: Linear interpolation

Polynomial interpolation

Spline interpolation

Nearest neighbor

Time-based interpolation (for time series)

41. Implications of using data interpolation in ML:

Interpolation helps maintain dataset integrity but can introduce bias or fake patterns if not applied carefully. It should be used with domain knowledge.

42. What are outliers in a dataset:

Outliers are extreme values that differ significantly from the rest of the data. They can result from errors or rare events.

43. Impact of outliers on ML models:

Skew model training

Distort regression lines

Reduce accuracy

Increase variance

44. Techniques to identify outliers:

Box plot (IQR method)

Z-score method

Scatter plots

DBSCAN (for clustering)

Isolation Forest

45. How to handle outliers:

Remove them

Cap or floor them (Winsorizing)

Transform data (log, square root)

Treat as a separate class (if meaningful)

46. Filter vs Wrapper vs Embedded Feature Selection:

Filter: Uses statistical tests, independent of ML model

Wrapper: Uses ML model to test feature subsets

Embedded: Feature selection happens during model training

47. Examples of algorithms:

Filter: Chi-Square, ANOVA, Mutual Information

Wrapper: Recursive Feature Elimination (RFE)

Embedded: Lasso Regression, Tree-based models (Random Forest)

48. Pros and Cons:

Filter

Fast, scalable – Ignores feature interaction

Wrapper

Accurate – Slow, computationally expensive

Embedded

Efficient, accurate – Depends on model used

49. Define feature scaling:

Feature scaling transforms features to the same scale, improving model performance, especially for distance-based models.

50. Process of standardization:

Standardization = (value − mean) / standard deviation It centers data at 0 with a standard deviation of 1.

51. Mean normalization vs standardization:

Mean normalization: (value − mean) / (max − min)

Standardization: (value − mean) / standard deviation Standardization is more robust to outliers.

52. Min-Max Scaling – Pros and Cons:

Pros:

Preserves data distribution

Useful when range [0,1] is needed

Cons:

Sensitive to outliers

Not ideal for skewed data

53. Purpose of unit vector scaling:

It scales data to have unit norm (length = 1). Useful when direction matters more than magnitude, such as in cosine similarity.

54. Define PCA (Principal Component Analysis):

PCA is a dimensionality reduction technique that transforms data into new features (principal components) that capture the most variance.

55. Steps involved in PCA:

Standardize data

Calculate covariance matrix

Find eigenvalues and eigenvectors

Sort eigenvectors by descending eigenvalues

Project data on top k components

56. Significance of eigenvalues and eigenvectors in PCA:

Eigenvectors define the directions of new feature axes

Eigenvalues define the magnitude (importance) of these axes

57. How PCA helps in dimensionality reduction:

By keeping only the top-k components with the most variance, PCA reduces feature count while retaining most information, speeding up models and reducing overfitting.

58. Define data encoding and its importance:

Data encoding transforms categorical data into numerical format for machine learning models to understand. Example: Label Encoding, One-Hot Encoding. It’s essential for converting text or labels into usable input.

59. Explain Nominal Encoding and provide an example.

Nominal Encoding is a technique used to convert categorical variables with no inherent order (nominal data) into a numerical format for machine learning models.

There are two common types of nominal encoding:

1. Label Encoding

Each unique category is assigned a number.

Drawback: May introduce unintended ordinal relationships.

Example:

Color: Red Blue Green Encoded: 0 1 2

2. One-Hot Encoding

Creates a new binary column for each category.

Recommended when there's no rank/order among categories.

Example:

Color:
Red → [1, 0, 0]
Blue → [0, 1, 0]
Green → [0, 0, 1]

Best Practice: Use One-Hot Encoding for nominal data to avoid misleading models with false ordering.

60. Discuss the process of One Hot Encoding

One Hot Encoding converts categorical values into multiple binary columns. Each category becomes a new column with a value of 1 (present) or 0 (absent).

61. How do you handle multiple categories in One Hot Encoding?

If a feature has many categories, it creates many new columns. To handle this:

Use dimensionality reduction (e.g., hashing).

Combine rare categories into an “Other” group.

Use alternatives like Target Encoding or Embedding.

62. Explain Mean Encoding and its advantages

Mean Encoding replaces a category with the mean of the target variable for that category. Advantage: Keeps numerical relation with target, useful in predictive models. Example: If category “A” has an average churn rate of 0.3, then "A" → 0.3.

63. Examples of Ordinal Encoding and Label Encoding

Ordinal Encoding (for ordered categories): Education → [High School=1, Bachelor=2, Master=3, PhD=4]

Label Encoding (for nominal data): City → [Delhi=0, Mumbai=1, Bangalore=2]

64. What is Target Guided Ordinal Encoding and how is it used?

It ranks categories based on the target variable's mean and assigns integers accordingly. Use: Helps models understand impact of categories on the target.

65. Define covariance and its significance in statistics

Covariance measures the direction of a linear relationship between two variables. Positive = both increase; Negative = one increases, the other decreases.

66. Explain the process of correlation check

Calculate correlation (like Pearson or Spearman) between all numeric features and the target. Identify highly correlated pairs (above threshold) to drop/reduce redundancy.

67. What is the Pearson Correlation Coefficient?

Measures linear relationship between two variables (ranges from -1 to 1). 1 = perfect positive, -1 = perfect negative, 0 = no correlation.

68. How does Spearman's Rank Correlation differ from Pearson's?

Pearson: Measures linear correlation.

Spearman: Measures monotonic relationship (based on rank). Useful when data is not normally distributed.

69. Importance of Variance Inflation Factor (VIF) in feature selection

VIF detects multicollinearity. High VIF (>5 or 10) means the feature is highly correlated with others and may be dropped.

70. Define feature selection and its purpose

Feature selection is the process of choosing the most relevant variables for a model. Purpose: Reduce overfitting, speed up training, and improve model performance.

71. Explain Recursive Feature Elimination (RFE)

RFE recursively removes least important features based on model weights, and ranks them. Best subset is selected based on performance.

72. How does Backward Elimination work?

Start with all features → remove one least significant (based on p-value) → retrain → repeat until optimal subset remains.

73. Advantages and limitations of Forward Elimination

Advantage: Simple, avoids overfitting with fewer features. Limitation: Might miss combinations that work well together.

74. What is feature engineering and why is it important?

Creating new features or modifying existing ones to improve model performance. Important because it brings domain knowledge into the data.

75. Steps in feature engineering

Understand data

Clean and preprocess

Create new features

Transform existing ones

Validate impact using model performance

76. Examples of feature engineering techniques

Extracting year/month from dates

Binning continuous variables

Creating interaction terms

Log transformation

77. How does feature selection differ from feature engineering?

Feature engineering: Creating or modifying features.

Feature selection: Choosing best among existing features.

78. Importance of feature selection in ML pipelines

It simplifies the model, avoids overfitting, reduces training time, and improves generalization.

79. Impact of feature selection on model performance

Good feature selection can increase accuracy, reduce noise, and enhance interpretability.

80. How to determine which features to include in a machine learning model?

Use statistical tests (e.g., chi-square, ANOVA)

Use model-based methods (RFE, Lasso)

Evaluate with cross-validation

Remove multicollinearity using VIF