Machine Learning Viva Questions from ML.pdf (Lab Programs)

Based on the ML.pdf lab manual, here are practical viva questions:

Program 1: Jupyter, Pandas, and NumPy

Q1: What is Jupyter Notebook?

Answer: Jupyter Notebook is an open-source, interactive development environment for data science and machine learning. It allows writing and running code in cells, mixing code with text, images, and equations in one place. Files are saved as .ipynb format containing both code and outputs.

Q2: What are the key features of Jupyter Notebook?

Answer:

- Interactive cell-by-cell code execution
- Supports multiple languages (Python, R, Julia)
- Visualization friendly for graphs and charts
- Combines code, documentation, and output in one place
- Ideal for data exploration and presentations

Q3: What is NumPy and why is it used?

Answer: NumPy (Numerical Python) is the foundational library for numerical computing in Python. It provides:

- N-dimensional arrays (ndarray) - faster than Python lists
- Vectorized operations (element-wise operations)
- Mathematical functions (trigonometric, statistical, algebraic)
- Matrix operations for linear algebra
- Efficient memory usage for large datasets

Q4: Give examples of NumPy operations.

Answer:

```
import numpy as np

# Creating arrays
arr = np.array([1, 2, 3])
```

```
# Mathematical operations
np.mean(arr)      # Calculate mean
np.sum(arr)       # Calculate sum
np.dot(a, b)      # Dot product

# Random numbers
np.random.rand(3, 3)  # Random 3x3 matrix
```

Q5: What is Pandas and what are its main data structures?

Answer: Pandas is a Python library for data analysis and manipulation. Two main structures:

1. **Series:** One-dimensional labeled data (like a column)
2. **DataFrame:** Two-dimensional table with rows and columns (like Excel spreadsheet)

Q6: What are the key features of Pandas?

Answer:

- Data cleaning: handling missing values, duplicates
- Data loading: CSV, Excel, SQL, JSON formats
- Data manipulation: sorting, filtering, grouping, merging
- Statistical analysis: mean, median, mode calculations
- Integration with NumPy and Matplotlib

Q7: Give examples of Pandas operations.

Answer:

```
import pandas as pd

# Reading data
df = pd.read_csv("data.csv")

# Viewing data
df.head()              # First 5 rows
df.info()              # Data types and info
df.describe()          # Statistical summary

# Data manipulation
df.groupby("category").mean()
df[df['age'] > 25]     # Filtering
df.sort_values('score') # Sorting
```

Q8: How do NumPy and Pandas work together?

Answer:

- NumPy provides fast numerical operations
- Pandas uses NumPy internally for managing tabular data

- Pandas DataFrames are built on top of NumPy arrays
- Both integrate seamlessly with Jupyter for visualization and analysis

Program 2: Simple Linear Regression

Q9: What is Simple Linear Regression?

Answer: Simple Linear Regression models the relationship between one independent variable (X) and one dependent variable (Y) using a straight line. The equation is: $Y = \beta_0 + \beta_1 X + \varepsilon$ where β_0 is intercept, β_1 is slope, and ε is error term.

Q10: What is the purpose of train_test_split?

Answer: train_test_split divides the dataset into training and testing sets:

- **Training set:** Used to train the model (typically 70-80%)
- **Testing set:** Used to evaluate model performance (typically 20-30%)
- Ensures model generalization to unseen data
- Prevents overfitting

Q11: Explain the LinearRegression model in sklearn.

Answer:

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)  # Train the model

# Model parameters
model.coef_[0]      # Slope (β₁)
model.intercept_    # Intercept (β₀)

# Prediction
model.predict([[6]])  # Predict for new value
```

Q12: What do model coefficient and intercept represent?

Answer:

- **Coefficient (Slope):** Change in Y for one unit change in X. Shows the strength and direction of relationship.
- **Intercept:** Value of Y when X = 0. Starting point of the regression line.
- Example: $Y = 25 + 7X$ means starting at 25, for each unit increase in X, Y increases by 7.

Q13: How do you visualize Simple Linear Regression?

Answer:

```
import matplotlib.pyplot as plt

# Scatter plot of actual data
plt.scatter(hours, scores)

# Regression line
plt.plot(hours, model.predict(hours))

plt.xlabel("Study Hours")
plt.ylabel("Scores")
plt.title("Simple Linear Regression")
plt.show()
```

Q14: What does test_size=0.2 mean in train_test_split?

Answer: test_size=0.2 means 20% of data is used for testing and 80% for training. This is a common split ratio that provides enough training data while reserving sufficient data for testing model performance.

Program 3: Logistic Regression

Q15: What is Logistic Regression and when is it used?

Answer: Logistic Regression is a classification algorithm (not regression despite the name) used for binary classification problems. It predicts probability (0 to 1) using sigmoid function. Used when output is categorical (Pass/Fail, Yes/No, 0/1).

Q16: What is the sigmoid function?

Answer: Sigmoid function converts any real number to value between 0 and 1: σ(z) = 1 / (1 + e^(-z))

- Output > 0.5 → Class 1
- Output < 0.5 → Class 0
- Creates S-shaped curve for probability estimation

Q17: Explain the LogisticRegression model in sklearn.

Answer:

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
model.fit(X_train, y_train)  # Train model

# Predictions
model.predict([[5]])           # Binary prediction (0 or 1)
model.predict_proba([[5]])     # Probability for each class
```

Q18: What is the difference between predict() and predict_proba()?

Answer:

- **predict():** Returns binary class label (0 or 1)
- **predict_proba():** Returns probability array [P(class 0), P(class 1)]
- Example: predict_proba([[5]]) might return [0.2, 0.8] meaning 20% chance of class 0, 80% chance of class 1

Q19: What does random_state parameter do?

Answer: random_state ensures reproducibility by setting a seed for random number generation. Using same random_state value gives identical train-test splits across different runs. Example: random_state=0 always produces same split.

Q20: Give a real-world example of Logistic Regression.

Answer:

- **Email spam detection:** Spam (1) or Not Spam (0)
- **Medical diagnosis:** Disease (1) or No Disease (0)
- **Loan approval:** Approved (1) or Rejected (0)
- **Student admission:** Pass (1) or Fail (0) based on study hours

Program 4: Decision Tree - ID3 Algorithm

Q21: What is a Decision Tree?

Answer: Decision Tree is a tree-structured classifier where:

- **Internal nodes:** Represent decisions on features
- **Branches:** Represent outcomes of decisions
- **Leaf nodes:** Represent final class labels Easy to interpret, handles both numerical and categorical data.

Q22: What is the ID3 Algorithm?

Answer: ID3 (Iterative Dichotomiser 3) builds decision trees using Information Gain. Steps:

1. Calculate entropy of dataset
2. For each feature, calculate information gain
3. Select feature with highest information gain
4. Split dataset based on that feature

5. Repeat recursively until all data is classified

Q23: What is Entropy in Decision Trees?

Answer: Entropy measures impurity/uncertainty in dataset: Entropy(S) = $-\Sigma(p_i * \log_2(p_i))$

- Entropy = 0: Pure node (all same class)
- Entropy = 1: Maximum impurity (equal distribution)
- Goal: Reduce entropy through splits

Q24: What is Information Gain?

Answer: Information Gain measures reduction in entropy after splitting: IG = Entropy(parent) - Weighted_Entropy(children)

- Feature with highest IG is selected for splitting
- Shows how much uncertainty is reduced
- Helps build optimal decision tree

Q25: Explain the entropy calculation in the code.

Answer:

```
def entropy(column):
    values, counts = np.unique(column, return_counts=True)
    ent = 0
    for i in range(len(values)):
        p = counts[i] / sum(counts)
        ent += -p * math.log2(p)
    return ent
```

- Gets unique values and their counts
- Calculates probability (p) for each value
- Applies entropy formula: $-\Sigma(p * \log_2(p))$
- Returns entropy value

Q26: What is the output format of the ID3 algorithm?

Answer: ID3 outputs a nested dictionary representing the decision tree structure:

```
{
  'Weather': {
    'Sunny': {...},
    'Overcast': 'Yes',
    'Rain': {...}
  }
}
```

- Keys are feature names
- Values are either subtrees (dict) or leaf labels (string)

Q27: What are the limitations of ID3?

Answer:

- Only works with categorical attributes (requires discretization for numerical)
- Prone to overfitting on small datasets
- Uses greedy approach (may not find global optimum)
- Biased towards features with many values
- No pruning mechanism

Program 5: k-Nearest Neighbors (k-NN)

Q28: What is the k-NN algorithm?

Answer: k-NN is a non-parametric, instance-based classification algorithm. It classifies a data point based on majority class of its k nearest neighbors. No explicit training phase - stores all training data for prediction.

Q29: How does k-NN work?

Answer:

1. Choose value of k (number of neighbors)
2. Calculate distance between test point and all training points
3. Select k nearest points
4. Perform majority voting
5. Assign most common class to test point

Q30: What distance metrics are used in k-NN?

Answer:

- **Euclidean Distance:** $\sqrt{\Sigma(x_i - y_i)^2}$ (most common)
- **Manhattan Distance:** $\Sigma|x_i - y_i|$
- **Minkowski Distance:** Generalization of both
- Choice depends on data characteristics and problem

Q31: How do you choose optimal value of k?

Answer:

- Small k (e.g., k=1): Sensitive to noise, high variance, overfitting

- Large k: More robust, may underfit, captures broad trends
- Use cross-validation to test different k values
- Odd k preferred for binary classification (avoids ties)
- Common rule of thumb: k = √n

Q32: Explain the KNeighborsClassifier parameters.

Answer:

```
from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier(n_neighbors=3)
model.fit(X_train, y_train)

# n_neighbors: Number of k neighbors
# metric: Distance metric (default: 'minkowski')
# weights: 'uniform' or 'distance' based weighting
```

Q33: What is the Iris dataset?

Answer: Iris dataset is a classic dataset with 150 samples of iris flowers:

- **Features:** Sepal length, sepal width, petal length, petal width (4 features)
- **Classes:** Setosa, Versicolor, Virginica (3 species)
- Used for classification tasks
- Well-balanced, clean dataset for learning

Q34: What does model.score() return?

Answer: model.score(X_test, y_test) returns the accuracy of the model:

- Accuracy = (Correct Predictions) / (Total Predictions)
- Range: 0 to 1 (or 0% to 100%)
- Measures overall performance on test set

Q35: Advantages and disadvantages of k-NN.

Answer: Advantages:

- Simple to understand and implement
- No training phase (lazy learning)
- Naturally handles multi-class problems
- Non-parametric (no assumptions about data)

Disadvantages:

- Computationally expensive at prediction time

- Requires feature scaling
- Sensitive to irrelevant features
- Memory intensive (stores all training data)
- Curse of dimensionality

Program 6: Naive Bayes Classifier

Q36: What is Naive Bayes Classifier?

Answer: Naive Bayes is a probabilistic classifier based on Bayes' theorem with "naive" assumption that features are independent. Despite this simplification, it works well for many real-world problems.

Q37: What is Bayes' Theorem?

Answer: P(A|B) = P(B|A) * P(A) / P(B)

- P(A|B): Posterior probability
- P(B|A): Likelihood
- P(A): Prior probability
- P(B): Marginal probability

Q38: Why is it called "Naive"?

Answer: Called "Naive" because it assumes all features are independent of each other, which is rarely true in real-world data. Despite this unrealistic assumption, it often performs surprisingly well.

Q39: What is GaussianNB?

Answer: GaussianNB is Naive Bayes variant that assumes features follow Gaussian (normal) distribution. Used for continuous features. Formula involves mean and variance of each feature for each class.

Q40: Explain the GaussianNB model code.

Answer:

```
from sklearn.naive_bayes import GaussianNB

model = GaussianNB()
model.fit(X_train, y_train)  # Train model

# Prediction
model.predict(sample)        # Class prediction
model.score(X_test, y_test)  # Accuracy
```

Q41: What are the types of Naive Bayes?

Answer:

1. **GaussianNB:** For continuous data (assumes Gaussian distribution)
2. **MultinomialNB:** For discrete counts (text classification, word counts)
3. **BernoulliNB:** For binary/boolean features
4. **ComplementNB:** For imbalanced datasets

Q42: Advantages of Naive Bayes.

Answer:

- Fast training and prediction
- Works well with high-dimensional data
- Requires small training dataset
- Handles multi-class classification naturally
- Not sensitive to irrelevant features
- Good for text classification and spam filtering

Q43: When should you use Naive Bayes?

Answer:

- Text classification (spam detection, sentiment analysis)
- Categorical input variables
- Real-time predictions needed
- Limited training data available
- High-dimensional feature space
- Quick baseline model

Program 7: PCA and LDA on Iris Dataset

Q44: What is PCA (Principal Component Analysis)?

Answer: PCA is unsupervised dimensionality reduction technique that:

- Transforms data to new coordinate system
- First principal component has largest variance
- Each successive component has maximum remaining variance
- Components are orthogonal (uncorrelated)
- Reduces features while retaining information

Q45: What are principal components?

Answer: Principal components are new features created by linear combination of original features. They represent directions of maximum variance in data. PC1 captures most variance, PC2 second most, etc.

Q46: Explain the PCA code.

Answer:

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)  # Reduce to 2 dimensions
X_pca = pca.fit_transform(X)

# Explained variance ratio
pca.explained_variance_ratio_
# Shows percentage of variance captured by each component
```

Q47: What is explained variance ratio?

Answer: Explained variance ratio shows the proportion of total variance captured by each principal component. Example: [0.92, 0.05] means:

- PC1 captures 92% of variance
- PC2 captures 5% of variance
- Together: 97% of information retained

Q48: What is LDA (Linear Discriminant Analysis)?

Answer: LDA is supervised dimensionality reduction technique that:

- Maximizes class separability
- Projects data to lower dimensions
- Requires class labels
- Finds directions that maximize between-class variance and minimize within-class variance
- Limited to C-1 features (C = number of classes)

Q49: Difference between PCA and LDA.

Answer:

Feature	PCA	LDA
Type	Unsupervised	Supervised
Goal	Maximize variance	Maximize class separability
Labels	Not required	Required
Components	Up to n features	Up to C-1 classes

Feature	PCA	LDA
Use case	Data compression	Classification

Q50: When to use PCA vs LDA?

Answer:

- **Use PCA when:**
  - No class labels available
  - Want to reduce dimensions for visualization
  - Data compression needed
  - Exploratory data analysis
- **Use LDA when:**
  - Have labeled data
  - Want to maximize class separation
  - Classification task
  - Number of classes < number of features

Program 8: DBSCAN Clustering

Q51: What is DBSCAN?

Answer: DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm that:

- Groups points in high-density regions
- Identifies outliers as noise
- Can find arbitrary-shaped clusters
- Doesn't require specifying number of clusters

Q52: What are the parameters of DBSCAN?

Answer:

- **eps (ε):** Maximum distance between two points to be neighbors
- **min_samples:** Minimum points required to form dense region (core point)
- These parameters define what "density" means for the algorithm

Q53: Explain core points, border points, and noise.

Answer:

- **Core Point:** Has at least min_samples points within eps distance (including itself)
- **Border Point:** Has fewer than min_samples neighbors but is reachable from a core point
- **Noise Point:** Neither core nor border; isolated points marked as outliers (-1 label)

Q54: Explain the DBSCAN code.

Answer:

```
from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=0.8, min_samples=2)
labels = dbscan.fit_predict(X)

# labels: Cluster assignments (-1 for noise)
# Clusters are numbered 0, 1, 2, ...
# -1 indicates noise/outliers
```

Q55: How do you create sample data with make_blobs?

Answer:

```
from sklearn.datasets import make_blobs

X, _ = make_blobs(
    n_samples=15,       # Number of data points
    centers=2,          # Number of cluster centers
    cluster_std=0.60,   # Standard deviation of clusters
    random_state=0      # For reproducibility
)
```

Q56: How does DBSCAN determine clusters?

Answer:

1. Pick arbitrary unvisited point
2. If it's a core point, start new cluster
3. Add all density-reachable points to cluster
4. If border point, add to existing cluster
5. If noise point, mark as outlier
6. Repeat until all points visited

Q57: Advantages of DBSCAN.

Answer:

- Doesn't require specifying number of clusters
- Finds arbitrary-shaped clusters
- Robust to outliers (marks them as noise)
- Can identify noise points
- Requires only two parameters

Q58: Disadvantages of DBSCAN.

Answer:

- Sensitive to eps and min_samples parameters
- Struggles with varying density clusters
- Not suitable for high-dimensional data (curse of dimensionality)
- Border points assigned to first cluster found (arbitrary)

Program 9: K-Medoids Clustering

Q59: What is K-Medoids clustering?

Answer: K-Medoids (PAM - Partitioning Around Medoids) is similar to K-Means but uses actual data points (medoids) as cluster centers instead of centroids. More robust to outliers and noise.

Q60: Difference between K-Means and K-Medoids.

Answer:

Feature	K-Means	K-Medoids
Center	Centroid (mean)	Medoid (actual point)
Outlier sensitivity	High	Low
Distance metric	Euclidean	Any metric
Computation	Faster	Slower
Cluster shape	Spherical	Flexible

Q61: What is a medoid?

Answer: A medoid is an actual data point in the cluster that minimizes the sum of distances to all other points in the cluster. It's the most centrally located point, not a calculated average.

Q62: Explain the K-Medoids code.

Answer:

```
from pyclustering.cluster.kmedoids import kmedoids

# Initial medoids (indices)
initial_medoids = [1, 7]

# Create model
kmedoids_instance = kmedoids(data, initial_medoids)
kmedoids_instance.process()

# Get results
clusters = kmedoids_instance.get_clusters()
```

```
medoids = kmedoids_instance.get_medoids()
```

Q63: What is PAM algorithm?

Answer: PAM (Partitioning Around Medoids) is the original K-Medoids algorithm:

1. Select k initial medoids randomly
2. Assign points to nearest medoid
3. For each medoid, try swapping with non-medoid point
4. Calculate total cost
5. If cost decreases, make swap permanent
6. Repeat until no improvement

Q64: What are CLARA and CLARANS?

Answer:

- **CLARA (Clustering Large Applications):** Extension of PAM for large datasets. Samples data, applies PAM to samples, selects best result.
- **CLARANS (Randomized CLARA):** Examines neighbors of nodes instead of data samples. More efficient with $O(n^2)$ complexity.

Q65: When to use K-Medoids over K-Means?

Answer: Use K-Medoids when:

- Data has outliers or noise
- Need robust clustering
- Distance metric other than Euclidean required
- Medoid interpretation is important
- Small to medium datasets
- Non-spherical clusters expected

Program 10: K-Means on Handwritten Dataset

Q66: What is the digits dataset?

Answer: The digits dataset contains 8×8 pixel images of handwritten digits (0-9):

- 1797 samples total
- 64 features (8×8 pixels flattened)
- 10 classes (digits 0-9)
- Grayscale values (0-16)
- Used for classification and clustering tasks

Q67: How does K-Means clustering work?

Answer:

1. Initialize k cluster centroids randomly
2. Assign each point to nearest centroid
3. Recalculate centroids (mean of assigned points)
4. Repeat steps 2-3 until convergence

- Minimizes within-cluster sum of squares (WCSS)

Q68: Explain the K-Means code.

Answer:

```
from sklearn.cluster import KMeans

# Create model
kmeans = KMeans(n_clusters=10, random_state=42)
kmeans.fit(X)  # Train model

# Get cluster labels
clusters = kmeans.labels_

# Cluster centers
kmeans.cluster_centers_
```

Q69: Why map clusters to real labels?

Answer: K-Means assigns arbitrary cluster numbers (0-9) that don't correspond to actual digit labels. Mapping finds which cluster best represents each digit by finding the most common true label in each cluster using:

```
label_map[cluster] = np.bincount(true_labels).argmax()
```

Q70: What is a confusion matrix in clustering?

Answer: Confusion matrix shows clustering performance:

- Rows: Actual digit labels
- Columns: Predicted cluster labels
- Diagonal: Correct assignments
- Off-diagonal: Misclassifications
- Used to calculate accuracy and identify confusion patterns

Q71: How do you visualize K-Means results?

Answer:

```
fig, axes = plt.subplots(2, 5, figsize=(10, 4))

for ax, img, label in zip(axes.flat, X, clusters):
    ax.imshow(img.reshape(8, 8), cmap="gray")
    ax.set_title(f"Cluster: {label}")
    ax.axis("off")

plt.show()
```

Shows sample images with their assigned cluster labels.

Q72: What does the accuracy score tell us?

Answer: Accuracy = Correct Predictions / Total Predictions

- Example: 0.795 (79.5%) means 795 out of 1000 digits correctly clustered
- Shows overall clustering quality
- Limited metric for clustering (unsupervised) compared to classification

Q73: Why use random_state in K-Means?

Answer: random_state ensures reproducibility:

- K-Means initializes centroids randomly
- Different initializations can give different results
- Setting random_state (e.g., 42) ensures same initialization every time
- Allows consistent results across runs

Q74: What are the limitations of K-Means?

Answer:

- Must specify k (number of clusters) beforehand
- Sensitive to initial centroid positions
- Assumes spherical clusters of similar size
- Sensitive to outliers (they affect centroid calculation)
- Converges to local minima (may not be global optimum)
- Uses only Euclidean distance

Q75: How can you improve K-Means performance?

Answer:

- Use K-Means++ initialization (sklearn default)
- Run multiple times with different initializations
- Standardize/normalize features
- Remove outliers before clustering

- Use elbow method to determine optimal k
- Try alternative algorithms (DBSCAN, Hierarchical) for complex shapes

General Lab Questions

Q76: What is the purpose of importing libraries?

Answer:

```
import numpy as np           # Numerical operations
import pandas as pd          # Data manipulation
import matplotlib.pyplot as plt  # Visualization
from sklearn... import ...   # Machine learning models
```

Libraries provide pre-built functions to avoid writing everything from scratch.

Q77: Why do we use reshape(-1, 1)?

Answer: reshape(-1, 1) converts 1D array to 2D column vector:

- sklearn expects 2D arrays for features
- -1 means "infer this dimension"
- 1 means one column
- Example: [1,2,3] becomes [[1],[2],[3]]

Q78: What does fit(), predict(), and score() do?

Answer:

- **fit(X, y):** Train the model on training data
- **predict(X):** Make predictions on new data
- **score(X, y):** Evaluate model accuracy on test data
- These are standard sklearn model methods

Q79: What is the purpose of random_state?

Answer: random_state ensures reproducibility:

- Sets seed for random number generation
- Same value produces same results across runs
- Important for debugging and comparing models
- Common values: 0, 42

Q80: How do you interpret model.coef_ and model.intercept_?

Answer: For linear models (Linear Regression, Logistic Regression):

- **coef_:** Weights/coefficients for features (slopes)
- **intercept_:** Bias term (y-intercept)
- Equation: y = intercept + coef[0]*x1 + coef[1]*x2 + ...

Q81: What is the difference between classification and clustering?

Answer: Classification:

- Supervised learning
- Requires labeled data
- Predicts class for new data
- Examples: Logistic Regression, k-NN, Decision Tree

Clustering:

- Unsupervised learning
- No labels required
- Groups similar data together
- Examples: K-Means, DBSCAN, K-Medoids

Q82: Why normalize/standardize data?

Answer:

- Features on different scales affect distance-based algorithms
- Prevents features with large ranges from dominating
- Improves gradient descent convergence
- Required for: k-NN, SVM, K-Means, PCA
- Not required for: Decision Trees, Naive Bayes

Q83: What is overfitting and how to prevent it?

Answer: Overfitting: Model learns training data too well (including noise) **Prevention:**

- Use more training data
- Cross-validation
- Regularization (L1, L2)
- Reduce model complexity
- Pruning (for trees)
- Early stopping
- Feature selection

Q84: What metrics evaluate classification models?

Answer:

- **Accuracy:** Overall correctness
- **Precision:** Correctness of positive predictions
- **Recall:** Coverage of actual positives
- **F1-Score:** Balance of precision and recall
- **Confusion Matrix:** Detailed breakdown
- **ROC-AUC:** Overall discrimination ability

Q85: What metrics evaluate clustering models?

Answer:

- **Silhouette Score:** How well points fit their clusters (-1 to 1)
- **Inertia/WCSS:** Within-cluster sum of squares (lower is better)
- **Davies-Bouldin Index:** Average similarity between clusters (lower is better)
- **Calinski-Harabasz Index:** Ratio of between/within cluster variance (higher is better)

Q86: What is the elbow method?

Answer: Method to find optimal k in K-Means:

1. Plot WCSS vs number of clusters
2. Look for "elbow" point where WCSS decrease slows
3. Elbow indicates optimal k
4. Balance between fit and complexity

Q87: What is a hyperparameter?

Answer: Parameters set before training (not learned from data):

- K-Means: k (number of clusters)
- k-NN: k (number of neighbors)
- Decision Tree: max_depth, min_samples_split
- DBSCAN: eps, min_samples
- Tuned using cross-validation, grid search

Q88: What is cross-validation?

Answer: Technique to evaluate model by splitting data into folds:

- K-fold CV: Split into k parts, train on k-1, test on 1
- Repeat k times, each fold used as test once
- Average results for robust estimate
- Prevents overfitting, better than single train-test split

Q89: Difference between supervised and unsupervised learning?

Answer: Supervised:

- Has labeled data (input-output pairs)
- Learns to predict labels
- Examples: Classification, Regression
- Algorithms: Linear Regression, Logistic Regression, k-NN, SVM

Unsupervised:

- No labels, only input data
- Finds patterns/structure
- Examples: Clustering, Dimensionality Reduction
- Algorithms: K-Means, PCA, DBSCAN

Q90: What is feature engineering?

Answer: Creating new features or transforming existing ones:

- **Feature extraction:** Derive new features (e.g., PCA)
- **Feature selection:** Choose relevant features
- **Feature transformation:** Scaling, encoding
- **Feature creation:** Combining features (e.g., age groups from age)
- Improves model performance and interpretability

Q91: How do you handle imbalanced datasets?

Answer: Techniques:

- Resampling: Oversample minority, undersample majority
- SMOTE: Synthetic minority oversampling
- Class weights: Penalize misclassification of minority class
- Different metrics: Use F1-score, precision-recall instead of accuracy
- Ensemble methods: Random Forest, XGBoost with balancing

Q92: What is the curse of dimensionality?

Answer: Problems in high-dimensional spaces:

- Distance becomes less meaningful
- Data becomes sparse
- More data needed for accurate models
- Computational complexity increases **Solutions:** PCA, LDA, feature selection, regularization

Q93: What is bias-variance tradeoff?

Answer:

- **Bias:** Error from wrong assumptions (underfitting)
- **Variance:** Error from sensitivity to training data (overfitting)
- **Tradeoff:** Decreasing one increases the other
- **Goal:** Find balance minimizing total error
- Complex models: low bias, high variance
- Simple models: high bias, low variance

Q94: How do you select a machine learning algorithm?

Answer: Consider:

- **Problem type:** Classification, regression, clustering
- **Data size:** Some algorithms need more data
- **Data quality:** Outliers, missing values
- **Interpretability:** Decision trees vs neural networks
- **Training time:** Real-time vs batch
- **Accuracy requirements:** Balance with complexity
- Start simple (baseline), then try complex models

Q95: What is the difference between parameters and hyperparameters?

Answer: Parameters:

- Learned during training
- Examples: weights, coefficients, centroids
- Optimized by algorithm

Hyperparameters:

- Set before training
- Examples: k, learning rate, max_depth
- Tuned manually or via grid/random search

Q96: Explain train, validation, and test sets.

Answer:

- **Training set (60-80%):** Train model
- **Validation set (10-20%):** Tune hyperparameters, model selection
- **Test set (10-20%):** Final evaluation (use only once)
- Prevents overfitting and data leakage
- Test set must remain unseen during development

Q97: What is gradient descent?

Answer: Optimization algorithm to minimize loss function:

1. Start with random parameters
2. Calculate gradient (slope) of loss
3. Update parameters in opposite direction of gradient
4. Repeat until convergence

- Learning rate controls step size
- Used in: Linear Regression, Logistic Regression, Neural Networks

Q98: What is regularization and why use it?

Answer: Technique to prevent overfitting by adding penalty:

- **L1 (Lasso):** Adds |coefficients|, can zero out features
- **L2 (Ridge):** Adds coefficients^2, shrinks but doesn't eliminate
- **Elastic Net:** Combines L1 and L2
- Controls model complexity
- Improves generalization to new data

Q99: What are ensemble methods?

Answer: Combining multiple models for better performance:

- **Bagging:** Train on random subsets (Random Forest)
- **Boosting:** Sequential training on errors (AdaBoost, XGBoost)
- **Stacking:** Use meta-model on predictions
- **Voting:** Majority vote or average
- Reduces overfitting and variance
- Often wins competitions

Q100: What is the no free lunch theorem?

Answer: States that no single machine learning algorithm works best for all problems. Algorithm performance depends on:

- Problem characteristics
- Data distribution
- Evaluation metric **Implication:** Must try different algorithms and evaluate empirically for each specific problem.

These 100 questions cover all the practical programs from your ML lab manual and should thoroughly prepare you for viva examinations!

Machine Learning Viva Questions with Answers

Based on the provided ML documents, here's a comprehensive list of viva questions with answers:

Unit 1: Introduction to Machine Learning

Q1: What is Machine Learning?

Answer: Machine Learning is a branch of artificial intelligence that enables computers to learn from data and improve their performance on tasks over time, without being explicitly programmed. It involves building mathematical models based on sample data (training data) to make data-driven predictions or decisions.

Q2: Explain the different types of Machine Learning.

Answer: There are four main types:

1. **Supervised Learning:** Trained on labeled data where input-output pairs are known (e.g., classification, regression)
2. **Unsupervised Learning:** Works with unlabeled data to find patterns (e.g., clustering, dimensionality reduction)
3. **Semi-Supervised Learning:** Uses both labeled and unlabeled data
4. **Reinforcement Learning:** Agent learns by interacting with environment and receiving rewards/penalties

Q3: What is the difference between overfitting and underfitting?

Answer:

- **Overfitting:** Model performs well on training data but poorly on new data. It learns noise in training data along with patterns. Characterized by low training error but high validation/test error.
- **Underfitting:** Model is too simple to capture underlying patterns. Shows poor performance on both training and testing data. High bias, low variance.

Q4: Explain the Bias-Variance Tradeoff.

Answer:

- **Bias:** Error due to overly simplistic models that fail to capture complex patterns (leads to underfitting)
- **Variance:** Error due to models being too complex and capturing noise (leads to overfitting)
- The goal is to find a balance that minimizes total error. As model complexity increases, bias decreases but variance increases.

Q5: What is the difference between classification and regression?

Answer:

- **Classification:** Predicts discrete categorical labels (e.g., spam/not spam, disease/no disease)
- **Regression:** Predicts continuous numerical values (e.g., house prices, temperature)

Unit 2: Regression

Q6: What is Simple Linear Regression?

Answer: Simple Linear Regression models the linear relationship between one independent variable (X) and one dependent variable (Y). The model is represented as: $Y = \beta_0 + \beta_1X + \varepsilon$, where β_0 is intercept, β_1 is slope, and ε is error term.

Q7: What is the OLS (Ordinary Least Squares) method?

Answer: OLS is a method for estimating parameters in linear regression by minimizing the sum of squared residuals (differences between observed and predicted values). It finds the line that best fits the data by minimizing $\Sigma(y_i - \hat{y}_i)^2$.

Q8: Explain R-squared and Adjusted R-squared.

Answer:

- **R^2:** Proportion of variance in dependent variable explained by independent variables. Range: 0-1. $R^2 = 1 - (SSR/SST)$
- **Adjusted R^2:** Modified R^2 that adjusts for number of predictors. Prevents overfitting by penalizing addition of unnecessary variables. Formula: $1 - (1-R^2)(n-1)/(n-k-1)$

Q9: What is Logistic Regression?

Answer: Logistic Regression is used for binary classification problems. It predicts probability that input belongs to a particular class using sigmoid function. Output ranges from 0 to 1. Despite the name, it's a classification algorithm, not regression.

Q10: What is PCA (Principal Component Analysis)?

Answer: PCA is an unsupervised dimensionality reduction technique that transforms data into a lower-dimensional space while retaining maximum variance. It identifies principal components (directions of maximum variance) using eigenvectors of the covariance matrix. Used for feature extraction and visualization.

Q11: Difference between PCA and LDA.

Answer:

- **PCA:** Unsupervised, maximizes variance, doesn't use class labels, projects to uncorrelated components
- **LDA:** Supervised, maximizes class separability, requires class labels, considers both within-class and between-class scatter

Q12: What is ICA (Independent Component Analysis)?

Answer: ICA is a computational technique for separating a multivariate signal into independent non-Gaussian components. It assumes sources are statistically independent and non-Gaussian. Commonly used in blind source separation (e.g., cocktail party problem).

Unit 3: Classification

Q13: What is the k-Nearest Neighbors (k-NN) algorithm?

Answer: k-NN is a non-parametric, instance-based classification algorithm. It classifies a data point based on the majority class of its k nearest neighbors. No explicit training phase - stores all training data. Uses distance metrics like Euclidean or Manhattan distance.

Q14: How do you choose the optimal value of k in k-NN?

Answer:

- Use cross-validation to try different k values
- Choose k that minimizes classification error
- Small k: sensitive to noise, high variance
- Large k: more robust but may underfit
- Odd values preferred for binary classification to avoid ties
- Common rule: $k = \sqrt{n}$ (where n is number of samples)

Q15: What is a Decision Tree?

Answer: Decision Tree is a tree-structured classifier where internal nodes represent decisions on features, branches represent outcomes, and leaf nodes represent class labels. Easy to interpret and visualize. Can handle both numerical and categorical data.

Q16: Explain Entropy and Information Gain.

Answer:

- **Entropy:** Measure of impurity/uncertainty in dataset. Entropy(S) = $-\Sigma(p_i * \log_2(p_i))$. Range: 0 (pure) to 1 (maximum impurity)
- **Information Gain:** Reduction in entropy after splitting on a feature. IG = Entropy(parent) - Weighted_Entropy(children). Feature with highest IG is selected for splitting.

Q17: What is the ID3 algorithm?

Answer: ID3 (Iterative Dichotomiser 3) is a decision tree learning algorithm that uses Information Gain to select features for splitting. It builds tree top-down, recursively selecting best feature at each node. Works with categorical data. Prone to overfitting. Limitation: greedy approach doesn't guarantee global optimum.

Q18: What is Gini Impurity?

Answer: Gini Impurity measures how often a randomly chosen element would be incorrectly labeled. Formula: Gini(S) = $1 - \Sigma(p_i^2)$. Range: 0 (pure) to 0.5 (maximum impurity for binary). Used in CART algorithm. Lower Gini means better split.

Q19: Explain pruning in Decision Trees.

Answer: Pruning reduces tree size to prevent overfitting:

- **Pre-pruning:** Stop tree growth early by setting constraints (max depth, min samples per leaf)
- **Post-pruning:** Grow full tree then remove branches with little importance
- Helps improve generalization and reduce complexity

Q20: What is Random Forest?

Answer: Random Forest is an ensemble learning method using bagging. It creates multiple decision trees trained on random subsets of data and features. Final prediction is majority vote (classification) or average (regression). Reduces overfitting, handles large datasets, provides feature importance.

Q21: What is Naive Bayes Classifier?

Answer: Naive Bayes is a probabilistic classifier based on Bayes' theorem. "Naive" because it assumes features are independent (which may not be true). Formula: $P(A|B) = P(B|A)*P(A)/P(B)$. Types: Gaussian, Multinomial, Bernoulli. Fast, works well with high dimensions, requires less training data.

Q22: What is Bayes' Theorem?

Answer: Bayes' Theorem describes probability of event based on prior knowledge: P(H|E) = P(E|H) * P(H) / P(E) Where:

- P(H|E): Posterior probability
- P(E|H): Likelihood
- P(H): Prior probability
- P(E): Marginal probability

Q23: What is Support Vector Machine (SVM)?

Answer: SVM is a supervised learning algorithm that finds optimal hyperplane to maximize margin between classes. Support vectors are data points closest to hyperplane. Can handle non-linear data using kernel trick. Effective in high dimensions. Types of kernels: Linear, Polynomial, RBF, Sigmoid.

Q24: What is the kernel trick in SVM?

Answer: Kernel trick transforms non-linearly separable data into higher-dimensional space where it becomes linearly separable, without explicitly computing transformation. Common kernels:

- Linear: $K(x,y) = x\cdot y$
- Polynomial: $K(x,y) = (x\cdot y + c)$^d
- RBF: $K(x,y) = \exp(-\gamma\|x-y\|^2)$

Q25: What are the hyperparameters in SVM?

Answer:

- **C (Regularization):** Controls trade-off between margin maximization and classification error. High C: fewer errors, may overfit. Low C: wider margin, may underfit.
- **Gamma (for RBF kernel):** Defines influence of single training example. High gamma: close influence, may overfit. Low gamma: far influence.

Q26: What is ensemble learning?

Answer: Ensemble learning combines multiple models to improve prediction accuracy and robustness. Types:

- **Bagging:** Trains models on random subsets (e.g., Random Forest)
- **Boosting:** Sequential training focusing on errors (e.g., AdaBoost, XGBoost)
- **Stacking:** Combines predictions using meta-model
- **Voting:** Majority vote or weighted average

Q27: Difference between Bagging and Boosting.

Answer:

- **Bagging:** Parallel training, equal weight to models, reduces variance, Random Forest
- **Boosting:** Sequential training, weights adjusted based on errors, reduces bias, AdaBoost/XGBoost
- Bagging: independent models; Boosting: dependent models

Q28: What is AdaBoost?

Answer: AdaBoost (Adaptive Boosting) combines weak learners sequentially. Misclassified samples get higher weights in next iteration. Final model is weighted combination of all weak learners. Simple and fast, but sensitive to noisy data and outliers.

Q29: What is XGBoost?

Answer: XGBoost (Extreme Gradient Boosting) is optimized gradient boosting algorithm. Features:

- Regularization (L1, L2) to prevent overfitting
- Handles missing values
- Parallel processing for speed
- Tree pruning using depth-first approach
- Built-in cross-validation
- Highly efficient and accurate

Unit 4: Clustering

Q30: What is Clustering?

Answer: Clustering is unsupervised learning technique that groups similar objects together. Objectives:

- Maximize intra-cluster similarity (within cluster)
- Maximize inter-cluster dissimilarity (between clusters)
- No labeled data required
- Used for pattern discovery, customer segmentation, anomaly detection

Q31: What is K-Means Clustering?

Answer: K-Means is partitioning method that divides data into k clusters. Algorithm:

1. Initialize k centroids randomly
2. Assign points to nearest centroid

3. Update centroids (mean of points in cluster)
4. Repeat until convergence

- Fast and simple, but requires k to be specified, sensitive to outliers, assumes spherical clusters

Q32: What is the Elbow Method?

Answer: Elbow method helps determine optimal k in K-Means. Plot Within-Cluster Sum of Squares (WCSS) vs. number of clusters. Look for "elbow point" where adding more clusters doesn't significantly reduce WCSS. The elbow indicates optimal k value.

Q33: Difference between K-Means and K-Medoids.

Answer:

Aspect	K-Means	K-Medoids
Center	Centroid (mean)	Medoid (actual point)
Outliers	Sensitive	Robust
Speed	Faster	Slower
Distance	Euclidean	Any metric

Q34: What is Hierarchical Clustering?

Answer: Hierarchical clustering builds tree-like structure (dendrogram) of clusters. Types:

- **Agglomerative (Bottom-up):** Start with individual points, merge similar clusters
- **Divisive (Top-down):** Start with one cluster, split recursively
- Doesn't require specifying k, produces dendrogram for visualization

Q35: Explain linkage methods in Hierarchical Clustering.

Answer:

- **Single Linkage:** Minimum distance between clusters (nearest neighbor)
- **Complete Linkage:** Maximum distance between clusters (furthest neighbor)
- **Average Linkage:** Average distance between all pairs
- **Centroid Linkage:** Distance between centroids
- **Ward's Linkage:** Minimizes within-cluster variance (most common)

Q36: What is DBSCAN?

Answer: DBSCAN (Density-Based Spatial Clustering of Applications with Noise) identifies clusters based on density. Parameters:

- **eps (ε):** Maximum distance between points
- **minPts:** Minimum points to form dense region
- Can find arbitrary-shaped clusters, handles noise, doesn't require k
- Sensitive to parameters, struggles with varying densities

Q37: Explain core points, border points, and noise in DBSCAN.

Answer:

- **Core Point:** Has at least minPts points within eps distance
- **Border Point:** Has fewer than minPts neighbors but is within eps of a core point
- **Noise Point:** Neither core nor border; isolated point
- Clusters formed by connecting core points and their borders

Q38: What is a Dendrogram?

Answer: Dendrogram is tree-like diagram showing hierarchical relationship in clustering. Y-axis shows distance at which clusters merge. Used to:

- Visualize clustering structure
- Decide optimal number of clusters by cutting at appropriate height
- Understand data hierarchy
- Closer objects merge at lower height

Q39: What is the Silhouette Score?

Answer: Silhouette Score measures how similar an object is to its own cluster compared to other clusters. Formula: $s(i) = (b(i) - a(i)) / \max(a(i), b(i))$

- $a(i)$: Average distance to points in same cluster
- $b(i)$: Average distance to nearest cluster
- Range: -1 to +1 (higher is better)
- Used to evaluate clustering quality

Q40: What is Gaussian Mixture Model (GMM)?

Answer: GMM is probabilistic model assuming data is generated from mixture of multiple Gaussian distributions. Uses Expectation-Maximization (EM) algorithm. Features:

- Soft clustering (probabilistic assignments)
- Can model clusters of different shapes and sizes
- Each cluster is Gaussian with mean (μ) and covariance (Σ)
- More flexible than K-Means

Evaluation Metrics

Q41: What is a Confusion Matrix?

Answer: Confusion Matrix is table showing model's prediction results:

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

- Used to calculate precision, recall, F1-score, accuracy

Q42: Explain Precision, Recall, and F1-Score.

Answer:

- **Precision:** TP/(TP+FP) - How many predicted positives are actually positive
- **Recall (Sensitivity):** TP/(TP+FN) - How many actual positives were correctly identified
- **F1-Score:** 2*(Precision*Recall)/(Precision+Recall) - Harmonic mean, balances precision and recall
- Used for imbalanced datasets

Q43: What is ROC-AUC?

Answer:

- **ROC (Receiver Operating Characteristic):** Curve plotting TPR (True Positive Rate) vs FPR (False Positive Rate) at various thresholds
- **AUC (Area Under Curve):** Measures overall model performance
- AUC = 1: Perfect classifier
- AUC = 0.5: Random classifier
- Higher AUC indicates better model

Q44: Difference between Accuracy and Precision.

Answer:

- **Accuracy:** (TP+TN)/(TP+TN+FP+FN) - Overall correctness
- **Precision:** TP/(TP+FP) - Correctness of positive predictions
- Accuracy can be misleading with imbalanced data
- Precision important when false positives are costly

Q45: What is cross-validation?

Answer: Cross-validation evaluates model by splitting data into k folds. Process:

1. Divide data into k equal parts

2. Train on k-1 folds, test on remaining fold
3. Repeat k times, each fold used as test once
4. Average results

- k-fold CV (typically k=5 or 10)
- Provides robust estimate of model performance
- Reduces overfitting

General Questions

Q46: What is feature scaling and why is it important?

Answer: Feature scaling transforms features to similar scale. Methods:

- **Normalization:** Scale to [0,1]. Formula: $(x-\min)/(\max-\min)$
- **Standardization:** Mean=0, SD=1. Formula: $(x-\mu)/\sigma$
- Important for distance-based algorithms (KNN, SVM, K-Means)
- Gradient descent converges faster with scaling

Q47: How do you handle missing data?

Answer: Techniques:

1. **Deletion:** Remove rows/columns with missing values
2. **Imputation:** Fill with mean, median, mode, or predicted values
3. **Forward/Backward Fill:** For time-series
4. **Mark as separate category:** For categorical data
5. **Use algorithms that handle missing values:** XGBoost

- Choice depends on amount and pattern of missing data

Q48: What is the curse of dimensionality?

Answer: As dimensions increase:

- Distance between points becomes less meaningful
- Data becomes sparse
- More data needed for accurate modeling
- Computational complexity increases
- Solutions: dimensionality reduction (PCA, LDA), feature selection

Q49: Difference between parametric and non-parametric models.

Answer:

- **Parametric:** Fixed number of parameters, assumes data distribution (e.g., Linear Regression, Naive Bayes). Faster, less data needed, but may underfit.
- **Non-parametric:** Parameters grow with data, makes fewer assumptions (e.g., KNN, Decision Trees). Flexible but computationally expensive.

Q50: What is regularization?

Answer: Regularization prevents overfitting by adding penalty term to loss function:

- **L1 (Lasso):** Adds |coefficients|, can reduce coefficients to zero (feature selection)
- **L2 (Ridge):** Adds coefficients^2, shrinks coefficients but doesn't eliminate
- **Elastic Net:** Combination of L1 and L2
- Controls model complexity and improves generalization

These 50 questions cover the major topics from your ML syllabus and should prepare you well for viva examinations!