Answering viva and assignments

LP3_Lab Manual1.pdf
PDF

in the manual after every assignment there are viva questions and Assignments questions give answer for all the questions atleast 3-5 lines long

Stopped reading documents
Got it 👍 — I can generate **3–5 line detailed answers** for *all Viva Questions* and *Assignment Questions* from each experiment in your **LP3 Lab Manual (DAA + ML sections)**.

Before proceeding, please confirm:
Do you want me to start with
✅ **Group A: Design and Analysis of Algorithms** (Assignments 1–6)
or
✅ **Group B: Machine Learning** (Assignments 1–4)?

Once you confirm, I'll give complete, well-formatted answers for every Viva + Assignment question in that group (with 3–5 lines each).

A then B

Perfect 👍
Let's begin with **Group A: Design and Analysis of Algorithms** from your LP3 Lab Manual.
Below are **3–5 line answers** for **all Viva Questions** and **Assignment Questions** from **Assignments 1–6**.

🧮 Assignment No. 1 – Fibonacci Numbers and Step Count

🔹 Viva Questions

1. What is the Fibonacci sequence of numbers?
The Fibonacci sequence is a series of numbers in which each number is the sum of the two preceding ones. It begins with 0 and 1, producing the sequence: 0, 1, 1, 2, 3, 5, 8, 13, etc. It appears frequently in nature, mathematics, and computer algorithms.

2. How does the Fibonacci sequence work?
The Fibonacci sequence works recursively — each term $F(n)$ is computed as $F(n-1) + F(n-2)$. The process continues until the desired term is reached. The sequence represents exponential growth patterns in nature and computation.

3. What is the Golden Ratio?
The Golden Ratio (approximately 1.618) is the ratio between successive Fibonacci numbers as they

increase. It represents aesthetic and natural balance, often found in art, design, and natural phenomena such as flowers and shells.

4. What is the Fibonacci Search technique?
Fibonacci Search is a search algorithm that uses Fibonacci numbers to divide the array and reduce search time. It's similar to binary search but uses Fibonacci intervals instead of halving, which can optimize search in sorted arrays.

5. What is the real application for the Fibonacci series?
Fibonacci numbers are used in computer algorithms (like dynamic programming), financial market analysis, data compression, and biological studies such as leaf arrangement and spiral patterns in plants.

🔹 Assignment Questions

1. Write algorithm/pseudo code to calculate Fibonacci numbers using non-recursive and recursive method.
The non-recursive (iterative) method uses a loop and stores previous two terms to compute the next. The recursive method calls itself until the base case (n ≤ 1) is reached. The iterative version is more efficient as it avoids redundant computations.

2. Explain time and space complexity of the recursive implementation to calculate Fibonacci numbers.
The recursive Fibonacci algorithm has exponential time complexity $O(2^n)$ due to repeated recomputation of subproblems. Its space complexity is $O(n)$ because of the recursive call stack depth.

3. Explain time and space complexity of the non-recursive implementation to calculate Fibonacci numbers.
The iterative Fibonacci implementation has linear time complexity $O(n)$ since each value is computed once, and constant space complexity $O(1)$, as only two variables are needed to store results.

4. Write a recurrence relation to find the nth Fibonacci number and solve it.
The recurrence relation is $F(n) = F(n-1) + F(n-2)$ with base cases $F(0) = 0, F(1) = 1$. Solving this recurrence gives the closed-form **Binet's formula**:

$$F(n) = \frac{(1+\sqrt{5})^n - (1-\sqrt{5})^n}{2^n\sqrt{5}}$$

5. Explain improvement over the purely recursive version for iterative computation of Fibonacci numbers.
Iterative methods eliminate redundant recursive calls by using loops and storing previous results, drastically reducing time from exponential to linear. It also minimizes stack usage, improving space efficiency.

6. Explain different techniques/algorithms to make Fibonacci sequence algorithm more efficient/optimized.
Optimizations include Dynamic Programming (memoization), Matrix Exponentiation (O(log n)), and using the Golden Ratio formula. These reduce redundant computations and improve runtime efficiency significantly.

🧩 Assignment No. 2 – Huffman Encoding (Greedy Strategy)

🔹 Viva Questions

1. What is Huffman Encoding?
Huffman encoding is a lossless data compression algorithm that assigns variable-length codes to characters based on their frequency. Characters with higher frequency get shorter codes to minimize total bit usage.

2. How many bits may be required for encoding the message 'mississippi'?
It depends on character frequency and generated Huffman codes. Since 's' and 'i' occur frequently, their codes are shorter, resulting in total bits far fewer than fixed-length encoding (roughly 19–20 bits for "mississippi").

3. Which tree is used in Huffman encoding? Give one Example.
A **Huffman Tree** is used — a binary tree where each leaf node represents a character, and its path defines the binary code. For example, if 'a' and 'b' have lowest frequencies, they combine first to form an internal node.

4. Why Huffman coding is lossless compression?
Huffman coding is lossless because no information is discarded during encoding; each character can be uniquely decoded using prefix-free binary codes that avoid ambiguity.

🔹 Assignment Questions

1. Write algorithm/pseudo code for Huffman Encoding.
The algorithm builds a min-heap of characters with frequencies, repeatedly merges the two smallest nodes, and assigns binary codes based on tree traversal — left as '0' and right as '1'.

2. Explain complexity analysis of Huffman Encoding using a greedy strategy.
Time complexity is $O(n \log n)$, dominated by min-heap operations for n characters. Space complexity is $O(n)$, storing tree nodes and frequency mappings.

3. Explain the purpose of Huffman coding. What are the basic principles of Huffman coding?
The purpose is to compress data efficiently. It follows principles of **frequency-based prioritization** and **prefix-free binary encoding**, ensuring no code is a prefix of another.

4. Find Huffman code for each symbol in the given text.
To find Huffman codes, count frequencies, construct a min-heap, repeatedly combine lowest frequency nodes, and assign binary values to edges until all characters have codes.

5. How do you calculate average code length in Huffman coding?
Average code length = Σ (frequency × code length) / total symbols. It measures compression efficiency and helps evaluate coding performance.

6. How are Huffman coding bits calculated? Explain with example.
Each character's bit length equals its tree depth. Multiply character frequency by code length to find total bits used. For example, if 'a' has code length 2 and occurs 10 times, it contributes 20 bits.

🎒 Assignment No. 3 – Fractional Knapsack (Greedy Method)

🔹 Viva Questions

1. What is Greedy Approach?
A greedy algorithm chooses the best possible option at each step with the hope of finding the global optimum. It doesn't reconsider previous decisions.

2. Explain concept of fractional knapsack.
In fractional knapsack, items can be divided; parts of an item can be included in the knapsack. Items are chosen based on the highest profit-to-weight ratio until the capacity is filled.

3. Difference between Fractional and 0/1 Knapsack.
In fractional knapsack, items can be divided into fractions, while in 0/1 knapsack, each item must be taken completely or left. Fractional knapsack uses a greedy approach, while 0/1 uses dynamic programming.

🔹 Assignment Questions

1. Explain greedy knapsack problem with example.
The greedy knapsack problem selects items based on the highest profit-to-weight ratio. For example, if an item has the best ratio but exceeds capacity, a fractional part is taken to maximize total profit.

2. Compute the optimal solution for N=3, M=20, (p)=(25,24,15), (w)=(18,15,10).
Profit/weight ratios: 1.39, 1.6, and 1.5. Take item 2 and 3 fully (15+10=25 > 20), so take item 3 fully (10 kg) and part of item 2 (10/15 fraction). Total profit ≈ 15 + (24×10/15) = 31.

3. Discuss solution for knapsack problem using branch and bound technique.
Branch and Bound explores all feasible combinations using bounding functions to eliminate unpromising branches. It ensures an optimal 0/1 solution but is computationally more intensive.

4. Write algorithm/pseudo code for fractional Knapsack using a greedy method.

1. Compute profit/weight ratio for each item.
2. Sort items in descending ratio order.
3. Add items until the knapsack is full; if overfilled, take fraction of the next.
4. Compute total profit.

5. Discuss time and space complexity.
Time complexity is $O(n \log n)$ due to sorting. Space complexity is $O(1)$ since only variables are needed for tracking ratios and profit.

6. Time complexity if objects are already sorted?
If items are pre-sorted by ratio, time complexity reduces to $O(n)$, as no sorting is needed — only linear iteration to fill capacity.

7. Why Greedy approach not optimal for 0-1 knapsack?
Because once a partial decision is made, it cannot be reversed; the greedy choice may ignore better global combinations, leading to suboptimal solutions.

Would you like me to continue with **Assignments 4–6 (Dynamic Programming Knapsack, Randomized Quick Sort, and Mini Project)** next before moving to **Group B (Machine Learning)**?

yes

Excellent ✅ — continuing with **Group A: Design and Analysis of Algorithms**, here are complete 3–5 line answers for **Assignments 4, 5, and 6**.

💼 Assignment No. 4 – 0/1 Knapsack using Dynamic Programming

🔹 Viva Questions

1. What is Dynamic Approach?
Dynamic Programming (DP) is a problem-solving technique that breaks problems into smaller overlapping subproblems, solving each only once and storing results for reuse. It optimizes recursive solutions by eliminating redundant calculations.

2. Explain concept of 0/1 Knapsack.
In 0/1 Knapsack, items cannot be divided — each item must be taken completely or not at all. The objective is to maximize total profit without exceeding the knapsack's weight limit using DP to ensure optimality.

3. Difference between Dynamic and Branch and Bound Approach. Which is best?
Dynamic Programming guarantees an optimal solution by systematically computing all subproblems, while Branch and Bound prunes the search tree to reduce computation. DP is preferred for smaller problem sizes; Branch and Bound is better for larger search spaces.

🔹 Assignment Questions

1. Explain 0-1 Knapsack problem with example.
Given a knapsack with limited capacity and several items with specific weights and profits, we aim to select a subset maximizing total profit. For example, using DP, we build a table to find the combination yielding the best profit without exceeding the capacity.

2. Compute the optimal solution for weights {3,4,6,5}, profits {2,3,1,4}, W=8.
By dynamic programming:

- Max value combinations are (3,4) = 5 weight → 5 profit, (3,5) = 8 weight → 6 profit, (4,5) = 9 weight (too heavy).
  ✅ Optimal solution = Profit = 6 (items with weights 3 & 5).

3. Write algorithm/pseudo code for 0/1 Knapsack using DP.

1. Initialize DP table `T[i][j]` = 0.
2. For each item `i` and capacity `j`:
   - If `weight[i] ≤ j`, then `T[i][j] = max(T[i-1][j], value[i] + T[i-1][j-weight[i]])`.
   - Else, `T[i][j] = T[i-1][j]`.
3. The last cell contains the maximum profit.

4. Discuss time and space complexity of 0/1 Knapsack problem.
Time complexity = $O(nW)$, as every cell of the DP table is computed once.
Space complexity = $O(nW)$, but can be optimized to $O(W)$ by using a 1D array.

5. Can knapsack be done in polynomial time? Explain.
The 0/1 Knapsack is NP-Complete, so no polynomial-time algorithm exists for all instances. However, pseudo-polynomial time $O(nW)$ solutions exist using DP when weights are small.

6. Discuss: 0/1 knapsack problem is an NP-Complete problem.
It's NP-Complete because there's no known polynomial-time algorithm that guarantees an optimal solution for all cases. It requires exhaustive or exponential computation without approximation techniques.

⚡ Assignment No. 5 – Quick Sort (Deterministic & Randomized Variants)

🔹 Viva Questions

1. What is a Randomized Algorithm?
A randomized algorithm uses random numbers in its decision-making process to reduce expected runtime or avoid worst-case inputs. The randomness improves efficiency and average performance.

2. What is deterministic Quick Sort?
Deterministic Quick Sort always chooses a fixed pivot (like the first, last, or middle element). Its time complexity can degrade to $O(n^2)$ for certain inputs, though it's efficient on average.

3. What is randomized Quick Sort?
Randomized Quick Sort randomly selects a pivot element before partitioning, ensuring that all permutations are equally likely. This reduces the chance of encountering the worst-case scenario.

4. What is the advantage of randomized algorithms?
They are simple, efficient, and often yield good average performance by minimizing dependence on input order. Randomness ensures probabilistic balance and avoids consistently poor partitions.

5. What is the difference between Monte Carlo and Las Vegas algorithms?
Las Vegas algorithms always produce correct results with variable execution time. Monte Carlo algorithms have fixed execution time but may yield incorrect results with a small probability.

🔹 Assignment Questions

1. Explain difference between Quick Sort and Randomized Quick Sort. What is deterministic Quick Sort?
Quick Sort uses a fixed pivot selection, whereas Randomized Quick Sort randomly chooses a pivot to balance partitions probabilistically. Deterministic Quick Sort may face worst-case $O(n^2)$ if pivot selection is poor.

2. Why is Randomized Quick Sort preferable to normal Quick Sort? Write pseudo code.
Randomized Quick Sort prevents worst-case inputs by random pivot selection, leading to average $O(n \log n)$ performance.
Pseudo code:

```
css

RandomizedPartition(A, p, r):
   i ← random(p, r)
   swap(A[i], A[r])
   return Partition(A, p, r)
RandomizedQuickSort(A, p, r):
   if p < r:
      q = RandomizedPartition(A, p, r)
      RandomizedQuickSort(A, p, q-1)
      RandomizedQuickSort(A, q+1, r)
```

3. Discuss complexity analysis of Randomized Quick Sort. What is overall Time Complexity?
Expected time complexity = $O(n \log n)$.
Worst-case complexity = $O(n^2)$ (very rare due to randomness).
Space complexity = $O(\log n)$ for recursion stack. The algorithm is not stable since element order may change during partitioning.

4. Discuss advantages of randomized algorithms. How to analyze them?
Advantages include simplicity, speed, and robustness to varied inputs. They're analyzed using expected running time, considering the average over all possible random choices.

5. Which is better – Randomized Quick Sort or Merge Sort? Why?
Randomized Quick Sort generally performs faster in practice due to in-place sorting and smaller constant factors. Merge Sort, though stable, requires extra space $O(n)$ but guarantees $O(n \log n)$ time in all cases.

🧠 Assignment No. 6 – Mini Project

🔹 Topics (any one)

- Matrix multiplication vs. Multithreaded Matrix multiplication
- Merge sort vs. Multithreaded Merge sort
- Naïve vs. Rabin–Karp String Matching
- Travelling Salesperson Problem (Exact & Approximation algorithms)

🔹 Viva Questions (Generalized)

1. What is the purpose of multithreading in algorithms?
Multithreading allows multiple computations to run concurrently, significantly reducing execution time for large datasets. It improves CPU utilization and efficiency in parallelizable problems.

2. Why is matrix multiplication important in computing?
Matrix multiplication is fundamental to linear algebra, computer graphics, machine learning, and scientific computation. It helps in transformations, neural networks, and systems of equations.

3. What is the difference between single-threaded and multithreaded merge sort?
A single-threaded merge sort runs sequentially, while a multithreaded version divides subarrays and sorts them in parallel. Multithreading enhances performance, especially on multicore processors.

4. How does the Rabin–Karp algorithm differ from Naïve string matching?
Naïve string matching checks all positions one by one, whereas Rabin–Karp uses hash values for fast pattern matching, reducing average time complexity to $O(n + m)$.

5. What are the challenges in implementing multithreaded algorithms?
Synchronization, data sharing, and thread management are key challenges. Improper handling can lead to race conditions, deadlocks, or inconsistent results.

🔹 Assignment Questions (Generic)

1. Describe your chosen mini project and its objective.
Example: *Matrix Multiplication vs. Multithreaded version* — the objective is to compare execution time and efficiency between single-threaded and multi-threaded approaches.

2. Explain the approach used to solve the problem.
The problem is approached by dividing the computation into smaller tasks (rows, columns, or subarrays) executed in parallel threads. Synchronization ensures correct merging of results.

3. Write pseudo code for the implemented approach.
Example for Multithreaded Matrix Multiplication:

```sql

for each row i in Matrix A:
   create thread T[i] to compute row i of result
   for each column j in Matrix B:
      result[i][j] = sum(A[i][k] * B[k][j])
wait for all threads to complete
```

4. Compare time complexity of both versions.
Sequential matrix multiplication: $O(n^3)$.
Multithreaded: roughly $O(n^3/p)$ where *p* = number of threads, with additional overhead for synchronization.

5. What are your observations and conclusions?
Multithreaded algorithms show better performance for large datasets and multicore systems. However, for small data, thread management overhead may negate speed gains.

✅ **Group A (DAA)** complete.
Would you like me to now proceed with **Group B (Machine Learning)** — covering all Viva and Assignment questions (Uber fare prediction, Email spam classification, ANN-based churn prediction, and Diabetes KNN classification)?

> for Group B answer the questions with that also explain the Assignment, the algorithms used in them everything i dont understand anything from Group B

Perfect ✅ — I'll explain **Group B (Machine Learning)** from your LP3 manual **from scratch** — as if you're learning it for the first time.

For every **assignment**, I'll cover:

1. 🧠 **Concepts explained simply**
2. ⚙️ **Algorithms used (step-by-step)**
3. 📊 **Viva & Assignment questions with 3–5 line answers**
4. 🧩 **Key takeaways**

Let's begin.

🧮 Group B – Machine Learning (ML Lab Assignments)

Assignment 1 – Predict the Price of an Uber Ride

🧠 Understanding the Assignment

We need to predict Uber ride fares using **Machine Learning regression algorithms** (Linear Regression & Random Forest Regression).
The dataset contains ride information such as pickup and drop-off locations, distance, and time.

Goal: Predict *fare amount* (a continuous value) using features such as distance, time, and coordinates.

⚙️ Algorithms Used

1️⃣ Linear Regression

- Used for predicting continuous numeric values.
- Finds a straight-line relationship between input variables (X) and output (Y).
- Equation:

$$Y = mX + b$$

  where *m* is the slope (weight) and *b* is the intercept.
- The algorithm learns *m* and *b* to minimize prediction error (using MSE – Mean Squared Error).

2️⃣ Random Forest Regression

- An *ensemble* algorithm combining many decision trees.
- Each tree predicts the fare; the final output is the **average of all trees**.
- Helps handle non-linear data and reduces overfitting.

🧩 Other concepts used

- **Data Preprocessing:** cleaning missing values, removing outliers.
- **Outlier Detection:** use Box Plot or IQR to find extreme values.

- **Correlation:** measures relationships between variables (e.g., distance vs. fare).
- **Haversine Formula:** computes distance between two GPS points using latitude/longitude.
- **Evaluation Metrics:**
  - **R^2 Score:** goodness of fit (closer to 1 is better).
  - **RMSE / MSE:** lower values mean more accurate predictions.

💬 Viva Questions

1. What is data preprocessing?
It is the process of cleaning and transforming raw data into a usable form. It involves handling missing values, converting data types, scaling numerical features, and removing noise to improve model accuracy.

2. Define Outliers.
Outliers are extreme values that differ significantly from most data points. They can occur due to errors or natural variation and can distort statistical analysis if not handled properly.

3. What is Linear Regression?
Linear Regression predicts a numeric outcome based on linear relationships between dependent and independent variables. It fits a straight line that minimizes the sum of squared errors between actual and predicted values.

4. What is Random Forest Algorithm?
Random Forest is an ensemble method that creates many decision trees on random subsets of data and averages their predictions. It handles non-linear patterns well and reduces overfitting.

5. Explain pandas and numpy.
pandas is a Python library for data handling using DataFrames (rows & columns). *numpy* is used for mathematical computations on arrays and matrices, providing fast numerical operations.

📘 Assignment Summary

- Clean and preprocess the Uber dataset.
- Calculate distances (Haversine formula).
- Apply Linear Regression and Random Forest.
- Compare their R^2 and RMSE values.
- Random Forest usually performs better due to its non-linearity handling.

Assignment 2 – Email Spam Classification

🧠 Understanding the Assignment

We must build a **binary classifier** that identifies whether an email is *Spam (1)* or *Not Spam (0)*.
Each email is represented as a row with word frequencies as columns (features).

⚙️ Algorithms Used

1️⃣ K-Nearest Neighbors (KNN)

- A *lazy learning* algorithm.
- To classify a new email, it looks at the *k* nearest emails in feature space and assigns the majority label.
- Example: If 4 of 5 nearest emails are spam, the new email is classified as spam.
- Distance metric: usually **Euclidean distance**.

2️⃣ Support Vector Machine (SVM)

- Finds a *hyperplane* that best separates spam and non-spam emails.
- It maximizes the margin (distance) between the two classes.
- Works well for high-dimensional text data.

🧩 Other Concepts

- **Binary Classification:** output = {0, 1}.
- **Train/Test Split:** dividing data for training and evaluation.
- **Feature Scaling:** ensures all features contribute equally.

💬 Viva Questions

1. What is binary classification?
It is a type of supervised learning where the output variable has two possible categories (e.g., Spam or Not Spam). The model learns to distinguish between the two classes based on training data.

2. What is KNN algorithm?
KNN classifies data points based on their neighbors. It computes distances to other points and assigns the most common label among the *k* closest samples. It's simple but computationally expensive for large datasets.

3. What is SVM algorithm?
SVM finds an optimal decision boundary that separates classes with maximum margin. It can also handle non-linear data using kernel functions like RBF or polynomial.

4. Why do we need data preprocessing in classification?
Because real-world data contains noise and missing values. Preprocessing normalizes the features, converts text to numeric form (e.g., word counts), and improves model accuracy.

5. What is feature scaling and why is it important?
Feature scaling transforms features to a similar range (e.g., 0–1). It prevents algorithms like KNN and SVM from being dominated by variables with larger numeric ranges.

📘 Assignment Summary

- Preprocess the email dataset (remove stop words, convert to word counts).
- Split into train/test sets.
- Train KNN and SVM models and compare accuracy and confusion matrix.
- SVM generally outperforms KNN in speed and accuracy for text data.

Assignment 3 – Neural Network for Bank Customer Churn

🧠 Understanding the Assignment

We must predict whether a bank customer will **leave (churn)** or **stay**.
This is a binary classification problem solved using **Artificial Neural Networks (ANNs)**.

⚙️ Algorithms Used

1️⃣ Artificial Neural Network (ANN)

- A computational model inspired by the human brain.
- Consists of **layers** of neurons: Input → Hidden → Output.
- Each neuron computes:

$$y = f(WX + b)$$

  where f is an activation function like ReLU or Sigmoid.
- The network learns by updating weights via **backpropagation** to minimize error.

2️⃣ Keras / TensorFlow

- Keras is a high-level library built on TensorFlow for easy ANN creation.
- It allows defining layers, activations, and optimizers in a few lines of code.

🧩 Other Concepts

- **Normalization:** scaling data so inputs are between 0 and 1.
- **Confusion Matrix:** evaluates predicted vs. actual results.
- **Accuracy & Precision:** measure model performance.

💬 Viva Questions

1. What is Normalization?
Normalization scales numerical values to a fixed range (usually 0–1) so that all features contribute equally to learning and prevent domination by larger values.

2. What is Standardization?
Standardization transforms data to have zero mean and unit variance (z-score). It is used when features

follow a Gaussian distribution.

3. Explain Confusion Matrix.
A Confusion Matrix is a 2×2 table that summarizes model predictions versus actual values: TP, FP, TN, FN. It helps compute metrics like accuracy, precision, and recall.

4. Define Classification Accuracy, Misclassification Rate, Precision.
Accuracy = (TP + TN) / Total samples.
Misclassification Rate = (FP + FN) / Total samples.
Precision = TP / (TP + FP).

5. Give one example of Confusion Matrix.
Example: If for 100 predictions, TP=50, TN=30, FP=10, FN=10:
Accuracy = (50+30)/100 = 80%, Misclassification Rate = 20%.

📘 Assignment Summary

- Load the bank dataset and normalize numeric features.
- Build an ANN with input, hidden, and output layers.
- Train and test the model using Keras/TensorFlow.
- Evaluate using accuracy and confusion matrix.
- Adjust layers or neurons to improve accuracy (typically 80–85%).

Assignment 4 – K-Nearest Neighbors on Diabetes Dataset

🧠 Understanding the Assignment

We predict if a patient has diabetes (based on age, BMI, glucose levels, etc.) using the KNN algorithm. It is a **classification** problem with target variable = Outcome (1 = diabetic, 0 = non-diabetic).

⚙️ Algorithm Used

1️⃣ K-Nearest Neighbors (KNN)

- Non-parametric and instance-based algorithm.
- To classify a new patient:
  - Compute distance to all training points (Euclidean).
  - Pick *k* closest points.
  - Assign the class most common among those *k* neighbors.
- Example: if 3 neighbors = [1, 1, 0], then prediction = 1 (diabetic).

🧩 Evaluation Metrics

- **Confusion Matrix** (TP, FP, TN, FN).

- **Accuracy = (TP+TN)/Total**
- **Precision = TP/(TP+FP)**
- **Recall = TP/(TP+FN)**
- **Error Rate = 1 – Accuracy**

💬 Viva Questions

1. What is K-Nearest Neighbors algorithm?
KNN is a supervised learning algorithm that classifies new samples based on similarity to known samples. It is simple, non-parametric, and effective for small datasets.

2. What is the use of Confusion Matrix?
It evaluates model performance by comparing actual and predicted labels, helping us calculate accuracy, precision, and recall.

3. What is ROC Curve and AUC?
ROC plots True Positive Rate vs. False Positive Rate. AUC (Area Under Curve) quantifies how well the model distinguishes classes; 1 indicates perfect classification.

4. What is the main advantage and disadvantage of KNN?
Advantage – easy to understand and no training time. Disadvantage – slow prediction for large datasets and sensitive to irrelevant features.

5. Why do we split data into training and testing sets?
To evaluate how well the model generalizes to new, unseen data. Training is for learning, testing for performance validation.

📘 Assignment Summary

- Load diabetes.csv and preprocess missing values.
- Split dataset into train/test sets.
- Train KNN model (e.g., k = 3).
- Compute Confusion Matrix, Accuracy, Precision, Recall, and Error Rate.
- Interpret results – accuracy ≈ 70–75%, which can improve by tuning *k* or feature scaling.

🧩 Final Takeaways from Group B

Concept	Algorithm	Problem Type	Key Metric	Example
Regression	Linear Regression, Random Forest	Predict numeric value	R^2, RMSE	Uber Fare
Classification	KNN, SVM	Binary (0/1)	Accuracy, Precision	Email Spam
Deep Learning	ANN	Binary (0/1)	Confusion Matrix	Bank Churn
Instance	KNN	Binary (0/1)	Accuracy, Recall	Diabetes

Concept	Algorithm	Problem Type	Key Metric	Example
Learning				Prediction

Would you like me to create a **one-page combined revision sheet** (formulas + workflow + key points) for all these ML experiments — perfect for viva and lab submissions?

> do for group C also blockcahin

Excellent — you're doing great 👍

Now let's move to **Group C: Blockchain Technology** (from your **LP3 Lab Manual**).
We'll go through every **experiment step-by-step** with:

- 🧠 Concept explained simply
- ⚙️ Algorithms / architecture used
- 💬 Viva & Assignment questions (3–5 lines each)
- 📘 Summary / takeaway

□ Group C — Blockchain Technology

Assignment 1 – Introduction to Blockchain and Basic Concepts

🧠 Concept

A **Blockchain** is a distributed, immutable digital ledger that stores transactions securely across many computers (nodes).
Each record is a *block* linked cryptographically to the previous one, forming a *chain*.
This ensures data transparency, integrity, and tamper resistance — no single party controls it.

⚙️ Core Structure

Each block typically contains:

- **Block Header:** hash of previous block, timestamp, nonce.
- **Transactions:** list of verified data entries.
- **Hash:** unique cryptographic fingerprint of block contents.

When a new transaction occurs:

1. It's broadcast to the network.
2. Nodes validate it (consensus).
3. A new block is created and added to the chain.

💬 Viva Questions

1. What is Blockchain?
Blockchain is a decentralized ledger that records transactions across multiple computers in a secure, transparent, and tamper-proof way using cryptographic hashing.

2. What is a block in Blockchain?
A block stores a group of verified transactions along with metadata such as timestamp, nonce, and hash of the previous block, ensuring immutability.

3. What is a distributed ledger?
A distributed ledger is a database replicated across multiple nodes in a network, where each node maintains an identical copy to ensure transparency and fault tolerance.

4. What is a hash function in Blockchain?
A hash function converts input data into a fixed-length alphanumeric code. It ensures integrity since even a small change in input drastically changes the hash output.

5. Why is Blockchain secure?
It's secure because it uses cryptography, consensus mechanisms, and decentralization—making tampering or altering data computationally impractical.

💬 Assignment Questions

1. Explain working of Blockchain with diagram.
A user initiates a transaction → transaction is broadcast → network nodes validate it → verified transactions are grouped into a block → block is added to the chain → all nodes update their copy.

2. What are key characteristics of Blockchain?
Decentralization, transparency, immutability, security, and consensus are core traits that make Blockchain reliable and trustworthy.

3. Differentiate between public, private, and consortium Blockchain.

- **Public:** anyone can join and validate (e.g., Bitcoin).
- **Private:** controlled by a single organization.
- **Consortium:** semi-decentralized, managed by multiple organizations.

4. What are applications of Blockchain?
Used in cryptocurrency, supply-chain tracking, digital identity, smart contracts, and healthcare record management.

5. What are main components of Blockchain architecture?
Blocks, nodes, transactions, hash functions, and consensus algorithms.

Assignment 2 – Consensus Mechanisms

🧠 Concept

A **consensus mechanism** ensures that all nodes agree on the validity of transactions and maintain a single version of truth.
Without a central authority, consensus allows trustless cooperation among nodes.

⚙️ Popular Algorithms

Mechanism	Principle	Used In
Proof of Work (PoW)	Miners solve complex puzzles to validate blocks.	Bitcoin
Proof of Stake (PoS)	Validators are chosen based on coin ownership.	Ethereum 2.0
Delegated PoS (DPoS)	Coin holders vote for delegates who validate.	EOS
Practical Byzantine Fault Tolerance (PBFT)	Nodes vote to reach majority consensus, tolerating malicious nodes.	Hyperledger

💬 Viva Questions

1. What is consensus in Blockchain?
Consensus is the process through which distributed nodes agree on the validity of a block before adding it to the chain.

2. Explain Proof of Work.
In PoW, miners compete to solve cryptographic puzzles. The first to solve adds a new block and earns a reward. It's secure but energy-intensive.

3. What is Proof of Stake?
Validators are selected based on how many coins they hold ("stake"). It's energy-efficient and discourages attacks through financial risk.

4. Why is consensus important?
It prevents double-spending, ensures consistency, and builds trust in a decentralized network.

5. Compare PoW vs PoS.
PoW uses computational power to secure the network, while PoS uses token ownership. PoS is faster and consumes less energy.

💬 Assignment Questions

1. Explain different types of consensus algorithms.
PoW, PoS, DPoS, PBFT, and Proof of Authority (PoA) are popular. Each balances speed, decentralization, and security differently.

2. Which consensus mechanism does Bitcoin use? Why?
Bitcoin uses PoW for high security and resistance to attacks through computational effort.

3. What are challenges in consensus algorithms?
Energy consumption, scalability, latency, and risk of centralization due to resource inequality.

4. What is Sybil attack in Blockchain?
A Sybil attack occurs when one entity creates multiple fake nodes to gain control of the network's consensus process.

5. How is consensus achieved without central authority?
Through cryptographic validation, incentives, and majority voting where honest nodes outnumber malicious ones.

Assignment 3 – Smart Contracts using Ethereum / Solidity

🧠 Concept

A **Smart Contract** is a self-executing program stored on the Blockchain that automatically enforces rules and transactions once predefined conditions are met.
They remove intermediaries and ensure transparency.

⚙️ Algorithm / Workflow

1. Write a smart contract in **Solidity**.
2. Compile it using **Remix IDE** or **Solc**.
3. Deploy it to the **Ethereum Blockchain** using a wallet (e.g., MetaMask).
4. Once deployed, contract functions execute automatically upon condition fulfillment.

💬 Viva Questions

1. What is a smart contract?
A smart contract is code deployed on the Blockchain that runs automatically when conditions are met, ensuring trustless automation.

2. What language is used to create Ethereum smart contracts?
Solidity — a high-level, object-oriented programming language similar to JavaScript.

3. What is Gas in Ethereum?
Gas is the computational cost paid in Ether to execute transactions or contracts, preventing abuse of resources.

4. What is the Ethereum Virtual Machine (EVM)?
The EVM is the runtime environment that executes smart contracts securely and deterministically on all Ethereum nodes.

5. Can a smart contract be changed after deployment?
No, once deployed it's immutable. Any changes require redeployment of a new contract.

💬 Assignment Questions

1. Write and deploy a simple Solidity contract.
Example:

```
solidity


pragma solidity ^0.8.0;
contract HelloWorld {
```

```
    string public message = "Hello Blockchain!";
}
```

Deploy on Remix and verify output using MetaMask.

2. What are advantages of smart contracts?
Automation, transparency, security, and elimination of intermediaries reduce cost and human error.

3. How does Ethereum differ from Bitcoin?
Bitcoin focuses on peer-to-peer currency, while Ethereum enables programmable contracts and decentralized applications (dApps).

4. What are limitations of smart contracts?
They are immutable, can contain bugs, depend on accurate external data, and consume Gas for every operation.

5. Give real-life examples of smart contract usage.
Insurance claim automation, supply-chain tracking, digital identity management, and real-estate agreements.

Assignment 4 – Blockchain Applications & Case Study

🧠 Concept

This assignment explores *real-world* use cases of Blockchain such as finance, supply chain, healthcare, and voting systems.
You analyze how Blockchain adds transparency, traceability, and security.

⚙️ Application Examples

Domain	Use Case	Benefit
Finance	Cryptocurrency payments, cross-border transfers	Fast, low-fee, trustless transactions
Supply Chain	Product tracking	Transparency, fraud reduction
Healthcare	Patient record storage	Privacy & interoperability
Voting	E-voting systems	Tamper-proof, transparent election process

---

💬 Viva Questions

1. What are common Blockchain applications?
Cryptocurrency, smart contracts, supply-chain tracking, digital identity, and decentralized finance (DeFi).

2. How does Blockchain enhance supply chain management?
By providing real-time, immutable tracking of goods, ensuring authenticity and reducing counterfeiting.

3. What are limitations of Blockchain technology?
High energy use, slow transaction speed, scalability issues, and regulatory uncertainty.

4. What is Hyperledger?
Hyperledger is an open-source enterprise Blockchain framework by Linux Foundation supporting private and permissioned ledgers.

5. What are tokens and cryptocurrencies?
Tokens represent digital assets or utilities on a Blockchain. Cryptocurrencies like Bitcoin or Ether act as digital money for transactions and rewards.

💬 Assignment Questions

1. Discuss any one real-world Blockchain use case.
Example: *Supply-chain transparency* — each step of product movement is recorded on Blockchain, ensuring authenticity from manufacturer to consumer.

2. How does Blockchain support healthcare systems?
It enables secure, shareable, and tamper-proof patient records accessible only by authorized parties.

3. What are the advantages of using Blockchain in banking?
It reduces transaction time, eliminates intermediaries, enhances auditability, and improves data security.

4. Explain role of smart contracts in business automation.
Smart contracts automatically enforce agreements when conditions are met, reducing manual processing and disputes.

5. Discuss future scope of Blockchain.
Blockchain is expanding into IoT, metaverse, NFTs, decentralized AI, and government data systems, driving transparency and efficiency.

🧩 Group C – Key Takeaways

Concept	Summary	Example Technology
Blockchain	Distributed ledger of immutable transactions	Bitcoin, Ethereum
Consensus	Mechanism for network agreement	PoW, PoS, PBFT
Smart Contracts	Self-executing code for automation	Solidity + EVM
Applications	Real-world implementations	Finance, Health, Supply Chain
Tools Used	Remix IDE, MetaMask, Ganache	For developing & testing contracts

---

✅ **Group C (Blockchain)** complete.

Would you like me to prepare a **final consolidated revision sheet** that covers **Groups A + B + C** (DAA + ML + Blockchain) — including definitions, formulas, algorithms, and viva-ready key points in a 2-page summary format?