	ANDHRA PRADESH STATE COUNCIL OF HIGHER EDUCATION

Model Syllabus for 4-Year UG Honours in B.Sc. (Artificial Intelligence) as Major in consonance with Curriculum framework w.e.f. AY 2025-26

Prepared by Adikavi Nannaya University, Rajahmundry

COURSE STRUCTURE (for Semester I to VI)

Year	Semester	Course	Title of the Course	No. of Hrs /Week	No. of Credits
I	I	1	Computer Fundamentals and Office Automation	3	3
			Computer Fundamentals and Office Automation Lab	2	1
		2	Mathematical Foundation for AI	4	4
	II	3	Python Programming and Data Structures	3	3
			Python programming and Data Structure Lab	2	1
		4	Artificial & Computational Intelligence	3	3
			Artificial & Computational Intelligence Lab	2	1
II	III	5	Statistical Foundation of AI	3	3
			Statistical Foundation of AI Lab	2	1
		6	DBMS	3	3
			DBMS Lab	2	1
		7	Exploratory Data Analysis & Data Visualization	3	3
			Exploratory Data Analysis & Data Visualization Lab	2	1
	IV	8	Data Science with R	3	3
			Data Science with R Lab	2	1
		9	Foundation of ML & Supervised Learning	3	3
			Foundation of ML & Supervised Learning lab	2	1
		10	Robotics Principles & Embedded systems	3	3
			Robotics Principles & Embedded systems Lab	2	1
III	V	11	Unsupervised & Reinforcement Learning	3	3
			Unsupervised & Reinforcement Learning Lab	2	1

Year	Semester	Course	Title of the Course	No. of Hrs /Week	No. of Credits
		12 A	Neural networks & Deep Learning	3	3
			Neural networks & Deep Learning Lab	2	1
		OR			
		12 B	Big Data Technologies	3	3
			Big Data Technologies Lab	2	1
		OR			
		12 C	Robotics Kinematics & Dynamics	3	3
			Robotics Kinematics & Dynamics Lab	2	1
		13 A	Natural Language Processing	3	3
			Natural Language Processing Lab	2	1
		OR			
		13 B	Cloud computing for Data Science	3	3
			Cloud computing for Data Science Lab	2	1
		OR			
		13 C	Additive Manufacturing & IoT	3	3
			Additive Manufacturing & IoT Lab	2	1
	VI	14 A	Conversational AI	3	3
			Conversational AI Lab	2	1
		OR			
		14 B	Time Series Analysis & Forecasting	3	3
			Time Series Analysis & Forecasting Lab	2	1
		OR			
		14 C	Robot Operating System	3	3
			Robot Operating System Lab	2	1
		15 A	Applications of Natural Language Processing	3	3
			Applications of NLP Lab	2	1
		OR			
		15 B	Data Engineering & MLOps	3	3
			Data Engineering & MLOps	2	1
		OR			
		15 C	Advanced AI Automation & Robotics	3	3
			Advanced AI Automation & Robotics Lab	2	1

Note: In the III Year (during the V and VI Semesters), students are required to select a pair of electives from one of the **THREE** specified domains. **For example: if set 'A' is chosen, courses 12 to 15 to be chosen as 12 A, 13 A, 14 A and 15 A.** To ensure in-depth understanding and skill development in the chosen domain, students must continue with the same domain electives in both the V and VI Semesters.

SEMESTER-I

COURSE 1: COMPUTER FUNDAMENTALS AND OFFICE AUTOMATION

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. **Understand foundational computing concepts**, including number systems, the evolution of computers, block diagrams, and generational progress.
2. **Develop knowledge of computer architecture**, focusing on system organization and networking fundamentals.
3. **Acquire practical skills in document creation**, formatting, and digital presentations using word processing tools.
4. **Gain proficiency in spreadsheet operations**, such as data entry, formulas, functions, and charting techniques.
5. **Introduce data visualization and basic modelling principles**, fostering analytical thinking in structuring and interpreting data sets.

Course Outcomes

1. At the End of the Course, The Students will be able to **explain different number systems**, the historical evolution of computers, and identify key components in a block diagram.
2. Learners will demonstrate **basic blocks of a computer and fundamental networking knowledge.**
3. Learners will create professional-level documents and **design visually appealing presentations** using word processing software and presentation software.
4. Learners will manipulate data within spreadsheets, apply formulas, and **generate accurate summaries and visualizations.**
5. Learners will apply data modelling techniques to **analyze, organize, and represent data effectively** in various scenarios.

Unit 1. Number Systems, Evolution , Block Diagram and Generations:

Number Systems: Binary, Decimal, Octal, Hexadecimal; conversions between number systems.

Evolution of Computers: History from early mechanical devices to modern-day systems.

Block Diagram of a Computer: Components like Input Unit, Output Unit, Memory, CPU (ALU + CU).

Generations of Computers: First to Fifth Generation – technologies, characteristics, examples.

Unit 2. Basic organization and N/W fundamentals:

Computer Organization: Functional components – Input/Output devices, Storage types, Memory hierarchy.

Types of Computers: Micro, Mini, Mainframe, and Supercomputers.

Networking Fundamentals: Definition, need for networks, types (LAN, WAN, MAN), topology (Star, Ring, Bus).

Internet Basics: IP Address, Domain Name, Web Browser, Email, WWW.

Unit 3. Word Processing and presentations:

Word Processing Basics: Using MS Word/Google Docs – formatting, styles, tables, mail merge.

Presentation Tools: Using PowerPoint/Google Slides – slide design, animations, transitions.

Applications: Creating resumes, reports, brochures, and presentations.

Keyboard Shortcuts

Unit 4. Spreadsheet Basics:

Spreadsheet Concepts: Understanding rows, columns, cells in tools like MS Excel/Google Sheets, cell referencing.

Functions and Formulae: SUM, AVERAGE, IF, COUNT.

Charts and Graphs: Creating visual representations

Data Handling: Sorting, filtering, conditional formatting.

Text Functions: LEFT, RIGHT, MID, LEN, TRIM, CONCAT, TEXTJOIN

Advanced Functions: **Logical**: IF, AND, OR, IFERROR, **Lookup**: VLOOKUP, HLOOKUP, XLOOKUP, INDEX, MATCH

Unit 5. Data Analysis and Visualization:

Conditional Formatting: Custom rules, Color scales, Icon sets, Data bars

Data Analysis Tools: Pivot Tables and Pivot Charts, Data Validation (Drop-downs, Input Messages, Error Alerts), What-If Analysis: Goal Seek, Scenario Manager, Data Tables

Charts and Dashboards: Creating Interactive Dashboards, Using slicers with Pivot Tables,Combo Charts and Sparklines

Productivity Tips: Using Named Ranges, Freeze Panes, Split View

Textbooks:

1. **Fundamentals of Computers,** Reema Thareja, Oxford University Press, Second Edition
2. **Fundamentals of Computers**, V. Rajaraman – PHI Learning
3. **Introduction to Computers** by Peter Norton – McGraw Hill
4. **Microsoft Office 365 In Practice** by Randy Nordell – McGraw Hill Education

References:

1. **Excel 2021 Bible** by Michael Alexander, Richard Kusleika – Wiley
2. **Networking All-in-One For Dummies** by Doug Lowe – Wiley
3. **Microsoft Official Docs and Training**: https://learn.microsoft.com
4. **Google Workspace Learning Center**: https://support.google.com/a/users/

Activities:

Outcome: At the End of the Course, The Students will be able to **explain different number systems**, the historical evolution of computers, and identify key components in a block diagram.

Activity: Create a digital poster or infographic comparing number systems (binary, decimal, octal, hexadecimal) and illustrating the timeline of computer generations with key innovations.

Evaluation Method: Rubric-based assessment of the poster presentation on a 10-point scale focusing on:

- Accuracy of number system conversions
- Correct identification of block diagram components
- Visual organization and creativity

Outcome: Learners will demonstrate **basic blocks of a computer and fundamental networking knowledge.**

Activity: Design a concept map showing the internal architecture of a computer and types of networks (LAN, WAN, MAN), including devices and topologies.

Evaluation Method: Checklist-based peer review and instructor validation:

- Completeness of the map
- Correctness of networking concepts

- Use of appropriate terminology
- Logical flow and structure of the map

Outcome: Learners will create professional-level documents and **design visually appealing presentations** using word processing software and presentation software.

Activity: Prepare a formal report (e.g., project proposal) in a word processor and present it using a slide deck with transitions, embedded media, and design elements.

Evaluation Method: Performance-based evaluation using a 10-point scoring scale:

- Formatting and structure of the document
- Presentation aesthetics and clarity
- Communication skills during presentation

Outcome: Learners will manipulate data within spreadsheets, apply formulas, and generate accurate summaries and visualizations.

Activity: Analyze a dataset (e.g., student scores or sales data) using spreadsheet software. Apply formulas (SUM, AVERAGE, IF, VLOOKUP) and create relevant charts.

Evaluation Method: Practical test with a rubric:

- Correct use of formulas
- Accuracy of data summarie

Outcome: Learners will apply data modelling techniques to **analyze, organize, and represent data effectively** in various scenarios.

Activity: Prepare an interactive dashboard for a given data set using EXCEL.

Evaluation Method: Evaluation of the dashboard on a 10-point scoring scale:

- Presentation aesthetics and clarity
- Interactiveness
- Communication skills during presentation

SEMESTER-I

COURSE 1: COMPUTER FUNDAMENTALS AND OFFICE AUTOMATION

Practical **Credits: 1** **2 hrs/week**

List of Experiments:

1. Demonstration of Assembling and Dessembling of Computer Systems.
2. Identify and prepare notes on the type of Network topology of your institution.
3. Prepare your resume in Word.
4. Using Word, write a letter to your higher official seeking 10-days leave.
5. Prepare a presentation that contains text, audio and video.
6. Using a spreadsheet, prepare your class Time Table.
7. Using a Spreadsheet, calculate the Gross and Net salary of employees(Min 5) considering all the allowances.
8. Generate the class-wise and subject-wise results for a class of 20 students. Also generate the highest and lowest marks in each subject.
9. Using IF, AND, OR, and IFERROR to Automate Grade Evaluation.
    a. Create a table of student scores in different subjects.
    b. Use IF to assign grades (A/B/C/Fail).
    c. Use IFERROR to handle missing scores or invalid data.
10. *Employee Database Search Using VLOOKUP, HLOOKUP, XLOOKUP, INDEX, and MATCH*
    a. Create a database of employees (Name, ID, Department, Salary).
    b. Implement VLOOKUP to search by employee ID.
    c. Use HLOOKUP to extract department heads by role.
    d. Apply XLOOKUP for more flexible searches.
    e. Use INDEX + MATCH as an alternative to VLOOKUP.
11. Sales Report Analysis Using Pivot Tables and Charts
    a. Use a dataset of product sales (Product, Region, Date, Quantity, Revenue).
    b. Create Pivot Tables to summarize data by region/product.
    c. Insert Pivot Charts for visual analysis (e.g., bar, line).
    d. Add slicers to make the dashboard interactive.
12. Designing a Data Entry Form with Drop-downs and Input Rules
    a. Create a student registration form.
    b. Add drop-down lists for course selection using Data Validation.

c. Add input messages to guide users.
d. Add error alerts for wrong entries.

13. Monthly Budget Planning using Goal Seek and Scenario Manager
    a. Create a simple personal budget (income, expenses, savings).
    b. Use Goal Seek to determine income needed to save a desired amount.
    c. Use Scenario Manager to compare different budgeting scenarios (best/ worst/ realistic case).
    d. Create a one-variable Data Table to analyze how different expenses affect savings.
14. Dashboard Creation Using Combo Charts, Sparklines & Slicers
    a. Use existing sales or attendance data.
    b. Insert combo charts (e.g., column + line).
    c. Add sparklines to show trends.
    d. Use slicers with Pivot Tables to control dashboard elements.
    e. Finalize and format for interactivity.

SEMESTER-I

COURSE 2: MATHEMATICAL FOUNDATION FOR AI

Theory **Credits: 4** **4 hrs/week**

Course Objectives

1. Develop a strong foundation in linear algebra, set theory, and functions essential for AI.
2. Understand and solve systems of linear equations using matrix methods.
3. Gain knowledge of eigenvalues, eigenvectors, and matrix diagonalization.
4. Learn fundamental concepts of probability and statistics for data analysis.
5. Explore functions and their applications relevant to AI problem-solving.

Course Outcomes

1. Solve complex linear algebra problems including matrix properties.
2. Apply set theory rigorously and compute eigenvalues/eigenvectors for intermediate examples.
3. Understand differentiation rules and solve constrained optimization problems.
4. Calculate probabilities in varied scenarios and explore discrete random variables.
5. Analyze data using comprehensive statistical measures and interpret visualizations.

Unit 1: Basic Linear Algebra and Systems of Linear Equations

Vectors and matrices: basics and operations (addition, multiplication, transpose, inverse)
Elementary row operations: row swapping, scalar multiplication, row addition
Row Echelon Form (REF), Reduced Row Echelon Form (RREF), Rank of matrix
System of linear equations: coefficient and augmented matrix representation
Types of solutions: unique, infinite, no solution
Gaussian elimination method using REF and back substitution

Unit 2: Set Theory and Eigen Concepts

Sets, subsets, set operations (union, intersection, difference, complement), Venn diagrams, Cartesian products
Relations and functions: definitions and properties, linear transformations, Eigenvalues, eigenvectors, characteristic polynomial, Diagonalization of matrices and symmetric matrices

Unit 3: Functions and their Properties

Definition, types of functions (polynomial, rational, exponential, logarithmic), Domain, range, and inverses of functions, Composition of functions, Continuity and basic limits, Graphical representation of functions, Maxima & Minima of functions.

Unit 4: Vector Differentiation

Vector differentiation –ordinary – derivatives of vectors – Differentiability – Gradient – Divergence - Curl operators - Directional derivatives of functions

Unit 5: Fundamentals of Probability & Basic Statistics

Probability: Concept of Uncertainty, Axioms and rules of probability, Conditional probability and independence, Law of total probability and Bayes' theorem
Measures of central tendency: Mean, Median, Mode
Measures of dispersion: range, interquartile range, variance, standard deviation
Introduction to correlation and covariance
Data representation: histograms, bar charts, scatter plots

Textbooks and References

1. Mathematics for Machine Learning, M. P. Deisenroth, A. A. Faisal, C. S. Ong, Cambridge University Press, 2020.
2. Introductory Linear Algebra, Howard Anton, Wiley.
3. Probability and Statistics for Engineers and Scientists, Ronald E. Walpole, Wiley.
4. Discrete Mathematics and its Applications, Kenneth H. Rosen, McGraw Hill.
5. Online Resources: Khan Academy, MIT OpenCourseWare (Linear Algebra, Probability, Statistics, Functions).

Activities:

Unit 1

Activity: Solve advanced linear equation systems using elementary row operations; explore matrix rank with concrete examples; interpret solutions graphically.
Evaluation Method: Assess problem-solving accuracy, clarity of solution process, and ability to classify solution types.

Unit 2

Activity: Practice set theory problems including Venn diagrams, unions, intersections; compute eigenvalues/eigenvectors for 3x3 matrices; perform matrix diagonalization exercises.

Evaluation Method: Evaluate completeness of set operations, correctness of eigen computations, and accuracy in diagonalization.

Unit 3

Activity: Plot and analyze various types of functions (polynomial, exponential, logarithmic); solve problems on function composition and inverses; perform simple graphical interpretations.

Evaluation Method: Assess quality of function plots, conceptual clarity of compositions and inverses, and accuracy of graphical analyses.

Unit 4

Activity: Calculate conditional probabilities; simulate discrete probability distributions; apply Bayes' theorem in practical scenarios (e.g., medical testing, reliability analysis).

Evaluation Method: Evaluate correct application of probability laws and rules, and logical use of Bayesian inference.

Unit 5

Activity: Analyze sample datasets to calculate central tendency measures and dispersion; compute correlation coefficients; create histograms and scatter plots; interpret data insights.

Evaluation Method: Assess accuracy of statistical calculations, clarity and correctness of visual data representation, and quality of interpretation.

SEMESTER-II

COURSE 3: PYTHON PROGRAMMING AND DATA STRUCTURES

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. To introduce the fundamentals of Python programming, including environment setup, syntax, and core concepts.
2. To develop problem-solving skills using control flow, functions, and modules.
3. To provide knowledge of Python data structures, file handling, and exception handling for effective programming.
4. To impart object-oriented programming concepts and GUI development skills for building applications.

Course Outcomes (COs)

After successful completion of the course, students will be able to:

1. Explain the basic features, syntax, data types, and operators of Python programming.
2. Apply control flow constructs, functions, and modules to develop structured Python programs.
3. Demonstrate the use of sequences, sets, and dictionaries for effective data handling and manipulation.
4. Implement file handling techniques and apply exception handling mechanisms for robust applications.
5. Develop object-oriented and GUI-based applications using Python.

Unit 1: Basics of Python Programming:

Introduction to Python, Features of Python, Programming Modes - Interactive Mode & Script Mode, Identifiers, Naming Conventions, Keywords (Reserved Words), Built-in Data Types, Literals - Integer, Float, Complex, Boolean, String, Variables, Operators, Expressions, Assignment Statements, Input/Output Statements, Python Syntax (Lines, Comments, Indentation),

Operators & Operands, Classification of Operators - Arithmetic Operators, Relational Operators, Logical Operators, Bitwise Operators, Assignment, Augmented Assignment, Identity Operators, Expressions & Precedence Rules

Unit 2: Control Flow, Functions & Modules:

Control Flow - if Statement, if-else, if-elif-else. Iterative Statements – while, for, Nested Loops, Loop Control Statements – break, continue, pass; else with loops

Need for Functions, Defining & Invoking User-defined Functions, Return Statement, Function Input/Output Cases, Scope of Variables - Local, Global, Nested Functions, Function Arguments - Required, Positional, Default, Variable-length, main() Function, Documentation Strings, Recursive Functions, Anonymous Functions (Lambda), Library Functions

Modules - Import, from..import, Creating & Using Modules, Namespaces

Unit 3: Core Data Types and Python Collections

Strings: Representation, Indexing, Slicing, Immutability, Operators, Methods, Formatting

Lists: Creation, Indexing, Slicing, Mutability, Common Methods, List Comprehension

Tuples: Immutability, Operations, Tuple Assignment

Sets and Frozensets: Methods, Mathematical Operations, Comprehension

Dictionaries: Key-Value Structure, Methods, Traversal, Nested Dictionaries

Unit 4: File Handling, Exception Management & Object-Oriented Programming

File Handling: Types, Opening, Reading, Writing, Closing, CSV Files, OS/Pathlib

Error Types, Exception Handling: try-except, raise, User-defined Exceptions, Assertions

OOP Concepts: Classes, Objects, Attributes, Methods, Constructor and Destructors

Encapsulation: Private and Public Members

Inheritance: Single, Multilevel, Multiple, Method Overriding

Unit 5: Abstract Data Structures and GUI Programming

Abstract Data Structures (ADTs): Concepts and Importance

Linked Lists: Definition, Types- Singly, Doubly, Circular; Node Structure, Insertion, Deletion, Traversal (Single Linked list implementation only)

Stacks: LIFO Principle, Implementation using List, Applications

Queues: FIFO Principle, Implementation using List, Priority Queues

GUI Programming with Tkinter: Widgets (Label, Button, Entry, Menu, Listbox, Canvas etc.), Event Handling, Building Simple GUI Apps

Textbooks:

1. Python Programming-An Object Oriented approach, Anita Goel, Universities Press
2. Python Programming using Problem Solving Approach Reema Thareja Oxford University Press 2020
3. Exploring Python, Budd T A, McGraw-Hill Education, 1st Edition, 2011.

Reference Book:

1. Python: The Complete Reference, Martin C. Brown, Mc Graw-Hill, 2018
2. Fundamentals of Python, Kenneth A. Lambert. (2019), First Programs,2nd Edition, CENGAGE Publication.

Activities:

Outcome: Explain the basic features, syntax, data types, and operators of Python programming.

Activity: Conduct a "Python Basics Lab" where students write small programs to demonstrate literals, variables, data types, and operators (e.g., swapping numbers, simple calculator).

Evaluation Method:

- Lab performance checklist (execution of 3 mini tasks)
- Short quiz with multiple-choice and fill-in-the-blanks on syntax, data types, and operators

Outcome: Apply control flow constructs, functions, and modules to develop structured Python programs.

Activity: Group activity - "Python Problem Solving Challenge": Students solve real-life problems (e.g., finding prime numbers, grade calculator, menu-driven calculator) using control structures, functions, and importing standard modules.

Evaluation Method:

- Code submission with proper use of functions/modules (20%)
- Viva-voce to explain logic and flow of control (40%)
- Unit test with scenario-based programming questions (40%)

Outcome: Demonstrate the use of sequences, sets, and dictionaries for effective data handling and manipulation.

Activity: Hands-on mini project – "Student Data Manager": Students create a program using lists, tuples, sets, and dictionaries to store and manipulate student records (e.g., marks, courses, hobbies).

Evaluation Method:

- Practical demo of program with at least 5 data operations (add, search, delete, update, traverse)
- Evaluation rubric for correctness, efficiency, and use of appropriate data structure

Outcome: Implement file handling techniques and apply exception handling mechanisms for robust applications.

Activity: Individual assignment - "File-Based Address Book": Students create a program to store, update, and retrieve data from files, with exception handling for invalid inputs or missing files.

Evaluation Method:

- Assessment of program correctness (file read/write, append, delete, exception handling)
- Short quiz with error-tracing and debugging questions (given code with errors, students identify and correct)

Outcome: Develop object-oriented and GUI-based applications using Python.

Activity: Mini Project – "Student Information System with GUI": Students design a simple Tkinter-based application with classes/objects for handling student data, including basic GUI widgets (Entry, Button, Listbox).

Evaluation Method:

- Project demo and presentation (50%)
- Rubric-based evaluation for OOP concepts (classes, inheritance, encapsulation) and GUI design (widgets, event handling) (30%)
- Peer review/feedback on usability (20%)

SEMESTER-II

COURSE 3: PYTHON PROGRAMMING AND DATA STRUCTURES

Practical **Credits: 1** **2 hrs/week**

1. Basic Python Programs:
    a. Write a program to display basic details (name, roll number, department) using print() and demonstrate different literal types (int, float, string, boolean, complex).
    b. Write a program to perform arithmetic, relational, logical, bitwise, and assignment operations on given inputs.
2. Control Flow Practice
    a. Write a program to find the largest of three numbers using if-elif-else.
    b. Write a program to check whether a number is prime or not using loops.
    c. Write a program to illustrate the use of loop control statements (break, continue, pass).
3. Functions and Recursion
    a. Write a program to define a function to calculate factorial of a number (using recursion).
    b. Write a program to demonstrate different types of function arguments (default, positional, keyword, variable-length).
4. Write a program to illustrate string slicing, concatenation, repetition, and built-in methods.
5. Write a program to create a list of numbers, perform insertion, deletion, searching, sorting, and list comprehension.
6. Write a program to demonstrate tuple packing, unpacking, and immutability.
7. Write a program to implement set operations (union, intersection, difference, subset, superset).
8. Write a program to create a dictionary of student roll numbers and marks, and perform add, update, delete, and traversal operations.
9. Write a program to read and display count of vowels, consonants, digits, and spaces of a text file.
10. Write a program to copy the contents of one file into another file.
11. Write a program to read and process student marks from a CSV file (calculate average, highest, lowest).

12. Write a program to demonstrate exception handling using try-except-finally.
13. Write a program to create a class Student with attributes and methods to display details.
14. Write a program to demonstrate single and multilevel inheritance.
15. Implement stack (LIFO) and queue (FIFO) using lists.
16. Implement singly linked lists: node creation, insertion, deletion, traversal.
17. Write a Tkinter program with Label, Entry, and Button widgets to take user input and display it.
18. Write a Tkinter program to create a simple calculator application.

SEMESTER-II

COURSE 4: ARTIFICIAL & COMPUTATIONAL INTELLIGENCE

Theory **Credits: 3** **3 hrs/week**

Course Objectives

- Introduce foundational concepts and history of Artificial Intelligence (AI).
- Teach the PEAS framework and agent-based problem solving.
- Develop understanding of uninformed and informed search methods.
- Provide basic knowledge of machine learning concepts and computational intelligence.
- Instill awareness of ethical considerations in AI.
- Introduce logic programming fundamentals using Prolog as a practical tool.

Course Outcomes

Students completing this course will be able to:

- Explain basic AI terminology, agent models, and the PEAS framework.
- Formulate AI problems using search strategies and implement basic algorithms conceptually.
- Understand core machine learning categories and their applications.
- Describe computational intelligence approaches and their role in AI.
- Analyze ethical issues related to AI technologies.
- Write and execute simple Prolog programs demonstrating logic programming fundamentals.

Unit 1: Introduction to Artificial Intelligence and PEAS Framework

Introduction to AI: Definition, history, applications, and scope

The PEAS framework: Performance Measure, Environment, Actuators, Sensors, Examples of PEAS in real-world AI systems

Intelligent agents: Intelligent agents and their environments, **Types of intelligent agents:** Simple reflex, model-based, goal-based, utility-based, Agent architectures and rationality

Unit 2: Expert Systems

Definition and components of Expert Systems (Knowledge Base, Inference Engine, User Interface), Rule-based systems and knowledge representation, Examples of expert systems: medical diagnosis, decision support, Limitations and comparison with AI agents, Role and significance of expert systems in AI evolution

Unit 3: Search Strategies in AI

Problem-solving as search: problem formulation, states, actions, goal test, Traveller's problem

Uninformed (Blind) Search: Breadth-first search, Depth-first search, Uniform-cost search

Informed (Heuristic) Search: Greedy best-first search, A* algorithm

Applications of search in AI problems

Unit 4: Introduction to Machine Learning

What is machine learning, definitions, Types of learning: Supervised, Unsupervised, Reinforcement learning (basic ideas), classification, Regression , clustering and Association, Basic learning algorithms overview and applications

Unit 5: Computational Intelligence and Ethics in AI

Overview of computational intelligence (Basics of fuzzy logic, neural networks), Role of computational intelligence in AI, Ethics and societal challenges in AI, Responsible AI, fairness, transparency, and safety concerns

Unit 1: Introduction to AI, PEAS Framework & Intelligent Agents

Activities:

Interactive lecture with real-world AI examples

Group discussion on PEAS framework for different environments

Case study analysis of intelligent agents in everyday AI applications

Quiz on AI fundamentals and agent types

Outcomes:

Students will explain AI basics and describe the PEAS components.

Students will identify types of intelligent agents and their roles.

Evaluation Method:

Quiz and participation: 10%

Assignment on PEAS and agent modeling: 10%

Unit 2: Expert Systems

Activities:

Lecture with multimedia explaining components of expert systems

Hands-on group activity designing rule-based systems for simple decision problems

Case study review of medical diagnosis expert systems

Class debate on limitations and advantages of expert systems

Outcomes:

Understand expert system architecture and knowledge representation.

Develop simple rule-based systems for decision making.

Analyze real-world expert system examples critically.

Evaluation Method:

Group assignment designing rule base: 15%

Written test on expert system concepts: 10%

Unit 3: Search Strategies in AI

Activities:

Demonstration of uninformed search algorithms

Interactive exercises formulating search problems

Simulation of heuristic search and A* algorithm

Problem-solving sessions applying search to puzzles

Outcomes:

Formulate search problems and apply uninformed and heuristic algorithms conceptually.

Explain the working and use cases of different search strategies.

Evaluation Method:

Problem formulation assignment: 10%

Class test on search algorithms: 10%

Unit 4: Basics of Machine Learning

Activities:

Video lectures on different machine learning paradigms

Simple data classification exercises/classification demos

Group presentations on various ML types and applications

Quiz on ML types and terminology

Outcomes:

Explain supervised, unsupervised, and reinforcement learning basics.

Identify sample use cases for machine learning.

Evaluation Method:

Quiz and presentation: 10%

Written assignment on ML overview: 5%

Unit 5: Computational Intelligence & Ethics

Activities:

Lecture introducing fuzzy logic, neural networks overview

Ethical dilemma discussions and case studies in AI technology

Role play on AI responsibility and fairness

Group project: Develop guidelines for responsible AI use

Outcomes:

Understand computational intelligence basics and ethical AI challenges.

Propose responsible AI practices for diverse scenarios**.**

Evaluation Method:

Participation in ethics discussions: 5%

Group project report and presentation: 15%

Recommended Textbooks and References

- Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 4th Edition
- Elaine Rich, Kevin Knight, Artificial Intelligence, 3rd Edition
- Michael Negnevitsky, Artificial Intelligence: A Guide to Intelligent Systems
- Ivan Bratko, Prolog Programming for Artificial Intelligence, 4th Edition
- Online resources: AI course materials from Coursera, NPTEL, GeeksforGeeks AI tutorials.

SEMESTER-II

COURSE 4: ARTIFICIAL & COMPUTATIONAL INTELLIGENCE

Practical **Credits: 1** **2 hrs/week**

***Note**: Experiments have to be conducted using Prolog*

1. Introduction to Prolog Environment and Syntax
   - Setting up Prolog, understanding facts, rules, and queries.
2. Defining Simple Facts and Queries
   - Write and test simple facts like family relationships, likes/dislikes.
3. Creating Rules in Prolog
   - Define logical rules with conditions and test queries.
4. List Handling in Prolog
   - Write programs to manipulate lists (head, tail, concatenation).
5. Recursion in Prolog
   - Implement recursive relations such as factorial and Fibonacci.
6. Search and Backtracking
   - Demonstrate Prolog's backtracking with sample queries and control cuts.
7. Family Relationship Programs
   - Model family trees and query relationships like siblings, ancestors.
8. Solve the Eight Queens Problem
   - Classic AI problem solved using backtracking.
9. Implement Simple Arithmetic Operations
   - Addition, subtraction, multiplication using Prolog predicates.
10. Monkey and Banana Problem
    - Logic problem modeling and solution.
11. Basic Expert System Prototype
    - Write rules for a simple medical diagnosis or recommendation system.
12. Implement Search Algorithms (Conceptual)
    - Demonstrate basic search algorithms like best-first search using Prolog rules.

SEMESTER-III

COURSE 5: STATISTICAL FOUNDATIONS FOR AI

Theory **Credits: 3** **3 hrs/week**

Course Objectives

- Develop a strong grasp of statistical concepts and their role in AI modeling.
- Equip students to describe and analyze random variables, expectation, and variance.
- Enable mastery over various discrete and continuous probability distributions relevant to AI.
- Teach methods for evaluating relationships and making predictions through correlation and regression.
- Instill skills for drawing valid conclusions via inference, estimation, and hypothesis testing.
- Introduce students to advanced topics like multivariate and time series analysis, connecting statistical foundations to AI applications.
- Foster data analysis proficiency using Advanced Excel for simulation and model building.

Course Outcomes

- By the end of this course, students will be able to:
- Analyze and interpret random variables using expectation and variance.
- Identify and apply discrete and continuous probability distributions to AI problems.
- Evaluate relationships between variables using correlation and regression analyses.
- Perform statistical inference, estimate parameters, and conduct hypothesis testing.
- Apply multivariate and time series analysis for complex, real-world AI scenarios.
- Utilize Advanced Excel tools for data exploration, visualization, and statistical modeling suited for AI tasks.

Unit 1: Random Variables, Expectation, and Variance

Random variables: definition, types (discrete & continuous), and properties, Probability mass function (PMF) and probability density function (PDF), Cumulative distribution function (CDF), Mathematical expectation (mean), variance, and standard deviation, Moments and moment-generating functions

Unit 2: Probability Distributions

Discrete distributions: Binomial, Poisson, Geometric, Negative Binomial distributions — definitions, properties, and examples

Continuous distributions: Uniform, Normal (Gaussian), Exponential, Gamma distributions — definitions, properties, and applications

Joint, marginal, and conditional distributions, Introduction to Central Limit Theorem

Unit 3: Correlation and Regression

Bivariate data and scatter plots

Correlation: Pearson and Spearman coefficients, interpretation

Simple linear regression: model, estimation, properties, and analysis of variance

Multiple linear regression basics (conceptual understanding)

Residuals and goodness of fit

Unit 4: Statistical Inference, Estimation, and Hypothesis Testing

Population and sample, parameters and statistics, Sampling distributions, Point and interval estimation (confidence intervals), Tests of significance: z-test, t-test, chi-square test, and F-test, p-values and errors (Type I & II), Power of a statistical test

Unit 5: Multivariate and Time Series Analysis

Introduction to multivariate data and covariance matrices, Principal Component Analysis (PCA) basics and dimensionality reduction (conceptual), Introduction to clustering (k-means concept), Time series: Trend, seasonal, cyclical, and irregular components, Autocorrelation and lag, moving averages, forecasting basics

Activities:

Unit 1: Random Variables, Expectation, and Variance

Activities:

Solve problems involving calculation of expectation and variance using real-world datasets.

Advanced Excel exercise: Compute mean, variance, and standard deviation for sample data.

In-class quiz: Define and distinguish discrete/continuous random variables.

Outcome:

Students can identify, classify, and compute key properties (mean, variance) for random variables using formulas and Excel.

Evaluation Method:

Quiz (written/in-class): 5%

Excel assignment submission: 5%

Unit 2: Probability Distributions

Activities:

Practical Excel session: Simulate and visualize binomial, Poisson, and normal distributions.

Group worksheet: Draw and describe PMFs, PDFs, and CDFs for common distributions.

Homework: Problems on marginal and joint distributions.

Outcome:

Students can model and visualize discrete and continuous distributions for AI-relevant scenarios using Excel.

Evaluation Method:

Excel lab practical: 5%

Homework/problem set: 5%

Unit 3: Correlation and Regression

Activities:

Excel activity: Prepare scatter plots, calculate Pearson/Spearman coefficients, apply linear regression tool.

Mini-project: Analyze relationships between variables in an AI dataset (e.g., prediction, feature analysis).

Class discussion: Interpret regression outputs and residuals.

Outcome:

Students can analyze and interpret relationships and predictions using correlation and regression tools in Excel.

Evaluation Method:

Mini-project or report: 8%

Excel lab practical/lab observation: 4%

Unit 4: Statistical Inference, Estimation, and Hypothesis Testing

Activities:

In-class demonstration: Confidence intervals and hypothesis testing in Excel (t-test, z-test, chi-square).

Lab: Conduct sample analysis and interpret p-values, errors.

Group case study: Real or simulated AI problem involving inference.

Outcome:

Students can perform estimation and hypothesis testing, and make statistically valid decisions using Excel analysis.

Evaluation Method:

Written/lab-based test: 8%

Group/case study submission: 4%

Unit 5: Multivariate and Time Series Analysis

Activities:

Excel-based PCA (Principal Component Analysis) on a sample dataset (using add-ins or manual approach).

Time series exercise: Calculate and plot moving averages, trends, and seasonal decomposition.

Presentation: Apply a multivariate or forecasting approach to a real AI dataset.

Outcome:

Students can summarize multivariate relationships and analyze time series patterns using Excel.

Evaluation Method:

Excel analysis/presentation: 8%

Final test/project component: 7%

Recommended Textbooks & References

1. Sheldon M. Ross, Introduction to Probability and Statistics for Engineers and Scientists
2. Douglas C. Montgomery & George C. Runger, Applied Statistics and Probability for Engineers
3. D.C. Agarwal, Statistics for Data Science and AI
4. Larry J. Stephens, Excel Data Analysis: Your visual blueprint for analyzing data, statistics, and AI
5. Online: Khan Academy Statistics & Probability, Coursera Statistics for Data Science, Microsoft Excel Help

SEMESTER-III

COURSE 5: STATISTICAL FOUNDATIONS FOR AI

Practical **Credits: 1** **2 hrs/week**

Advanced Excel Lab

1. Random Variable Simulation

   - Simulate and visualize discrete/continuous random variables using Excel functions and Data Analysis Toolpak.

2. Expectation & Variance Calculation

   - Use Excel formulas to compute expected value, variance, and standard deviation from given datasets.

3. Modeling Discrete Probability Distributions

   - Generate and plot Binomial and Poisson distributions; analyze probabilities and mean/variance.

4. Modeling Continuous Distributions

   - Simulate Normal and Exponential distributions; use NORM.DIST, NORM.INV, EXPON.DIST functions.

5. Correlation Analysis

   - Calculate Pearson/Spearman correlation coefficients; visualize with scatter plots and trendlines.

6. Linear Regression in Excel

   - Fit a linear regression model, interpret coefficients, predict new values; use REGRESSION tool.

7. Statistical Inference & Estimation

   - Create confidence intervals using Excel formulas; visualize sampling distributions.

8. Hypothesis Testing

   - Perform z-test, t-test, chi-square tests in Excel; interpret p-values and results.

9. Multivariate Analysis

   - Conduct simple PCA (manual or via add-ins); analyze covariance matrices for multidimensional datasets.

10. Time Series Analysis

    - Organize time series data, create line charts, calculate moving averages, detect trends and seasonality, basic forecasting.

SEMESTER-III

COURSE 6: DATABASE MANAGEMENT SYSTEMS

Theory **Credits: 3** **3 hrs/week**

Course Objectives

- Understand fundamentals of data, file systems, and database systems architecture.
- Develop skills in conceptual data modeling through ER/EER diagrams.
- Learn relational database principles including keys, constraints, relational algebra, and normalization.
- Gain proficiency in SQL for data definition, manipulation, and basic procedural programming with PL/SQL.
- Understand NoSQL concepts with a focus on MongoDB architecture, CRUD operations, data modeling, indexing, and replication.

Course Outcomes

At the end of the course, students will be able to:

- Explain database concepts, architectures, and compare file-based systems to DBMS.
- Design conceptual data models using ER/EER diagrams and map to relational schemas.
- Apply SQL queries to perform data definition, manipulation, aggregation, and control PL/SQL blocks.
- Demonstrate fundamental NoSQL concepts and perform CRUD and query operations in MongoDB.
- Model data effectively in MongoDB and understand optimization techniques such as indexing and replication.

Unit 1: Overview of Database Management System

Introduction to data, information, database, and DBMS, File-based system and its drawbacks, Database approach and advantages, Classification of Database Management Systems, Various Data Models and Components of DBMS, Three-schema architecture of database, Costs and risks of database approach

Unit 2: Entity-Relationship Model and Relational Model

Introduction to ER Model: entities, attributes, relationships, Classification of entity sets and attributes, Relationship degree and classification, Reducing ER diagram to tables, Enhanced

ER model: generalization, specialization, IS-A relationships, Relational model: CODD Rules, concepts of keys, relational integrity, Relational algebra operations, Functional dependencies and normal forms

Unit 3: Structured Query Language and PL/SQL

Introduction to SQL and data types, Data Definition Language (DDL) and Data Manipulation Language (DML), Selection, projection, join, set operations, Aggregate functions, views, subqueries, Table modification commands, Introduction to PL/SQL: language elements, procedures, functions, triggers

Unit 4: Introduction to NoSQL and MongoDB Basics

What is NoSQL? Features and types of NoSQL databases, CAP theorem and BASE properties, Difference between RDBMS and NoSQL, MongoDB Architecture: database, collection, BSON format, MongoDB Datatypes (String, Number, Date, Boolean, Array, Embedded Docs), Installation and Setup: Mongo shell and GUI, Database and collection management

Unit 5: MongoDB Operations, Data Modeling, and Optimization

CRUD operations: insertOne, find, updateOne, deleteOne, bulk operations, Query operators and working with arrays, Data modeling in MongoDB: embedded vs referenced documents, Aggregation framework: pipelines, stages, operators, Indexing (single, compound, multikey, text indexes), Replication concepts: replica sets, failover, consistency

Activities

Unit 1: Overview of Database Management System

Activity: Study different database models and classify DBMS architectures.

Outcome: Understand database systems fundamentals and architectures.

Evaluation Method: Written quiz on database models and classification; short assignment analyzing file-based vs DBMS approach.

Unit 2: Entity-Relationship Model and Relational Model

Activity: Create ER and EER diagrams for a sample application.

Outcome: Develop skills in conceptual modeling and understand relational integrity and normalization.

Evaluation Method: Project submission of ER/EER diagrams; quiz on relational algebra and normalization.

Unit 3: Structured Query Language and PL/SQL

Activity: Write and execute SQL queries on sample databases; develop PL/SQL procedures and triggers.

Outcome: Ability to manipulate database data and write procedural code blocks.

Evaluation Method: Practical exam on SQL query writing and PL/SQL program development.

Unit 4: Introduction to NoSQL and MongoDB Basics

Activity: Install and configure MongoDB; perform basic CRUD operations in shell/GUI.

Outcome: Understand NoSQL principles and MongoDB architecture.

Evaluation Method: Lab assignments on basic MongoDB commands and database/collection management.

Unit 5: MongoDB Operations, Data Modeling, and Optimization

Activity: Design MongoDB schema using embedded and referenced documents; implement aggregation pipelines and indexing.

Outcome: Model and query document data and optimize performance.

Evaluation Method: Project on MongoDB data modeling and aggregation; written test on indexing and replication concepts.

Recommended Textbooks & References

1. Abraham Silberschatz, Henry F. Korth, S. Sudarshan, “Database System Concepts”, McGraw Hill Education.
2. Ramez Elmasri and Shamkant B. Navathe, “Fundamentals of Database Systems”, Pearson.
3. Kristina Chodorow, “MongoDB: The Definitive Guide”, O’Reilly Media.
4. MongoDB Manual & Official Documentation (https://docs.mongodb.com/)
5. C.J. Date, “An Introduction to Database Systems”, Pearson.

SEMESTER-III

COURSE 6: DATABASE MANAGEMENT SYSTEMS

Practical **Credits: 1** **2 hrs/week**

Experiment 1 : Database: Inventory Management

Table 1: Products

Structure:

Column Name	Data Type	Constraints
product_id	INT	PRIMARY KEY
product_name	VARCHAR(50)	NOT NULL
price	DECIMAL(10,2)	CHECK(price > 0)
stock_qty	INT	CHECK(stock_qty >= 0)

Sample Data:

product_id	product_name	price	stock_qty
1	Pen	10.00	100
2	Notebook	50.00	200
3	Stapler	120.00	50
4	Marker	25.00	80
5	File Folder	60.00	150

Table 2: Suppliers

Structure:

Column Name	Data Type	Constraints
supplier_id	INT	PRIMARY KEY
supplier_name	VARCHAR(50)	NOT NULL
contact_no	VARCHAR(20)	UNIQUE
product_id	INT	FOREIGN KEY REFERENCES Products(product_id)

Sample Data:

supplier_id	supplier_name	contact_no	product_id
101	StationeryMart	9876543210	1
102	PaperWorld	9876500000	2
103	OfficeSupplies	9876512345	3
104	MarkerHub	9876522222	4
105	FileDepot	9876533333	5

Section A: DDL (Data Definition Language)

1. Create a database called InventoryDB.
2. Create a table Products and table Suppliers with the specified columns and constraints:

Section B: DML (Data Manipulation Language)

4. Insert at least 5 rows into the Products table.
5. Insert at least 5 rows into the Suppliers table.
6. Update the stock quantity of product 'Pen' to 120.
7. Delete a supplier with a specific supplier_id.
8. Write a query to rename 'Notebook' to 'NoteBook A4'

Section C: DQL (SELECT Queries)

9. Display all records from the Products table.
10. Display only product_name and price of all products.
11. List all products that have a stock quantity less than 100.
12. Show all products between 20 and 100 price range.
13. Find all suppliers whose contact number starts with '98765'.
14. Find the average price of products.
15. Display the total number of products in the inventory.
16. Show the maximum and minimum stock quantities.
17. Count how many suppliers supply each product.
18. Show all products where price > 50 AND stock_qty > 100.
19. Show all products where price < 20 OR stock_qty < 80.
20. Display suppliers whose supplier_name contains the word 'Mart'
21. List all suppliers along with the product they supply (use INNER JOIN).
22. Display suppliers whose name starts with 'S'.
23. Find products whose name has exactly 5 characters
24. Find suppliers who supply products costing more than 100.

Experiment 2 : ONLINE BOOKSTORE DB

An online book store wants to implement a BOOKSTORE DB for managing their online transactions by using the following tables.

Authors Table

Column Name	Data Type	Constraints
author_id	INTEGER	PRIMARY KEY
first_name	VARCHAR	NOT NULL
last_name	VARCHAR	NOT NULL
nationality	VARCHAR	NULL allowed

Books Table

Column Name	Data Type	Constraints
book_id	INTEGER	PRIMARY KEY

Title	VARCHAR	NOT NULL
author_id	INTEGER	FOREIGN KEY REFERENCES Authors
publication_year	INTEGER	
Price	DECIMAL	

Customers Table

Column Name	Data Type	Constraints
customer_id	INTEGER	PRIMARY KEY
first_name	VARCHAR	NOT NULL
last_name	VARCHAR	NOT NULL
Email	VARCHAR	UNIQUE, NOT NULL
Address	VARCHAR	NOT NULL

Orders Table

Column Name	Data Type	Constraints
order_id	INTEGER	PRIMARY KEY
customer_id	INTEGER	FOREIGN KEY REFERENCES Customers
book_id	INTEGER	FOREIGN KEY REFERENCES Books
order_date	DATE	NOT NULL
quantity	INTEGER	NOT NULL

SAMPLE DATA SET for BOOKSTORE DB

Authors Table

author_id	first_name	last_name	nationality
1	Jane	Austen	British
2	George	Orwell	British
3	Gabriel	Garcia Marquez	Colombian
4	Toni	Morrison	American
5	Mark	Twain	American
6	Harper	Lee	American
7	Fyodor	Dostoevsky	Russian

Books Table

book_id	**Title**	**author_id**	**publication_year**	**price**
101	Pride and Prejudice	1	1813	12.99
102	1984	2	1949	9.50
103	One Hundred Years of Solitude	3	1967	15.00
104	Beloved	4	1987	11.25
105	Animal Farm	2	1945	8.75
106	Adventures of Huckleberry Finn	5	1884	10.50
107	To Kill a Mockingbird	6	1960	14.00

Customers Table

customer_id	first_name	last_name	Email	address
201	Alice	Smith	alice.s@example.com	12 Oak St, London
202	Bob	Johnson	bob.j@example.com	45 Pine Ave, Oxford
203	Charlie	Brown	charlie.b@example.com	78 Maple Rd, Bristol
204	Diana	Prince	diana.p@example.com	34 Queen St, York
205	Edward	Norton	edward.n@example.com	22 River Ln, Leeds
206	Fiona	Hall	fiona.h@example.com	56 Lake Dr, Bath
207	Greg	Miller	greg.m@example.com	89 Park Ave, Glasgow

Orders Table

order_id	customer_id	book_id	order_date	Quantity
301	201	101	2025-07-20	1
302	202	102	2025-07-21	2
303	201	105	2025-07-22	1
304	203	103	2025-07-23	1
305	204	106	2025-07-24	1
306	205	107	2025-07-25	3
307	206	104	2025-07-26	2

Section A: DDL (Schema Design & Constraints)

1. Write SQL statements to create all 4 tables (Authors, Books, Customers, Orders) with:
    - Primary Keys
    - Foreign Keys
    - Appropriate data types
    - NOT NULL constraints where necessary.

2. Alter the Books table to add a constraint that price must be greater than 0.
3. Add a new column phone_number to the Customers table (VARCHAR(15)) and ensure it is unique.
4. Drop the phone_number column from the Customers table.

Section B: DML (Data Manipulation)

5. Insert at least 7 records for each table (use sample dataset above).
6. Update the price of the book titled *Animal Farm* by increasing it by 10%.
7. Delete all orders made before 2025-07-21.
8. Change the nationality of Gabriel Garcia Marquez to "Latino-American".

Section C: SELECT Queries (Data Querying)

9. List all books published between 1900 and 2000.
10. Find all customers whose email contains "example.com".
11. Retrieve books whose price is between 10 and 15 and published before 1950.
12. Show authors who are either ‘British’ or ‘American’.
13. Find books that have a price less than 10 or are published after 1980.
14. Display all orders placed after 2025-07-22.
15. List all books written by author with author_id = 2.
16. Find customers whose last name starts with B.
17. Show all books with a price NOT between 9 and 13.
18. Display books whose publication_year is in (1813, 1945, 1987).
19. Find authors whose nationality is NOT ‘British’.
20. List customers whose address contains the word Park.
21. Show all books sorted by price in descending order.
22. List authors in alphabetical order by last_name.
23. Display orders sorted by order_date (latest first).

Use of Date Functions

24. Show all orders placed in July 2025.
25. Show all orders with an estimated delivery date (5 days after order date).
26. Show customers who placed an order on a weekend.
27. Calculate how many days have passed since the last order was placed.

Aggregate Functions (COUNT, SUM, AVG, MIN, MAX)

28. Count the total number of books in the database.
29. Find the average price of all books.
30. Show the highest-priced book.
31. Count how many orders each customer has placed.
32. Calculate the total sales (price × quantity) for each customer.

GROUP BY and HAVING

33. Count how many books are written by each author.
34. Group orders by customer_id and display total quantity ordered.
35. Show customers who have ordered more than 2 books in total (use HAVING).
36. Find the total number of books sold per author (GROUP BY author).

Experiment 3: EMPLOYEE DB

An enterprise wants to automate its employee management process by implementing an Employee Database.The goal is to replace manual record-keeping with a centralized system that stores employee, department, and project details. Use the following table structures and data set to implement Employee DB.

EmployeeDB - Table Structures

1. Departments Table

Column	**Type**	**Constraints**
dept_id	INT	PRIMARY KEY
dept_name	VARCHAR	UNIQUE, NOT NULL
location	VARCHAR	NOT NULL

2. Employees Table

Column	Type	Constraints
emp_id	INT	PRIMARY KEY
first_name	VARCHAR	NOT NULL
last_name	VARCHAR	NOT NULL
email	VARCHAR	UNIQUE, NOT NULL
phone	VARCHAR	CHECK (phone LIKE '--____')
hire_date	DATE	NOT NULL
job_title	VARCHAR	NOT NULL
salary	DECIMAL	CHECK (salary > 0)
dept_id	INT	FOREIGN KEY REFERENCES Departments(dept_id)
manager_id	INT	FOREIGN KEY REFERENCES Employees(emp_id) (self-referential)

3. Projects Table

Column	**Type**	**Constraints**
project_id	INT	PRIMARY KEY
project_name	VARCHAR	NOT NULL
start_date	DATE	NOT NULL
end_date	DATE	NULL
dept_id	INT	FOREIGN KEY REFERENCES Departments(dept_id)

4. Employee_Project Table (Many-to-Many)

Column	Type	Constraints
emp_id	INT	FOREIGN KEY REFERENCES Employees(emp_id), PRIMARY KEY(emp_id, project_id)
project_id	INT	FOREIGN KEY REFERENCES Projects(project_id)
hours_allocated	INT	CHECK (hours_allocated > 0)

Sample Data Set

Departments Table

dept_id	dept_name	Location
1	HR	New York
2	IT	San Francisco
3	Finance	Chicago
4	Marketing	Boston
5	Operations	Seattle
6	Legal	Washington D.C.
7	Sales	Dallas
8	R&D	Austin
9	Procurement	Denver
10	Customer Care	Miami

2. Employees Table

emp_id	first_name	last_name	Email	phone	hire_date	job_title	salary	dept_id	manager_id
101	Alice	Johnson	alice.j@corp.com	123-456-7890	2020-03-15	HR Manager	75000	1	NULL
102	Bob	Smith	bob.s@corp.com	234-567-8901	2019-05-20	IT Analyst	65000	2	104
103	Charlie	Brown	charlie.b@corp.com	345-678-9012	2021-01-10	Finance Executive	58000	3	106

104	Diana	Prince	diana.p@corp.com	456-789-0123	2018-07-12	IT Manager	90000	2	NULL
105	Ethan	Hunt	ethan.h@corp.com	567-890-1234	2022-02-25	Marketing Lead	62000	4	NULL
106	Fiona	Hall	fiona.h@corp.com	678-901-2345	2017-11-01	Finance Manager	85000	3	NULL
107	Greg	Miles	greg.m@corp.com	789-012-3456	2023-04-15	IT Support	45000	2	104
108	Hannah	White	hannah.w@corp.com	890-123-4567	2021-09-05	HR Executive	50000	1	101
109	Ian	Scott	ian.s@corp.com	901-234-5678	2020-11-20	Operations Analyst	56000	5	NULL
110	Julia	Adams	julia.a@corp.com	012-345-6789	2019-12-18	Legal Advisor	70000	6	NULL

3. Projects Table

project_id	**project_name**	**start_date**	**end_date**	**dept_id**
201	Payroll System	2023-01-01	NULL	3
202	Website Upgrade	2023-02-10	NULL	2
203	Recruitment Drive	2023-03-05	NULL	1
204	Ad Campaign	2023-05-20	NULL	4
205	New CRM Tool	2023-04-15	NULL	7
206	Compliance Portal	2023-06-10	NULL	6
207	Inventory System	2023-07-01	NULL	5
208	AI Research	2023-08-05	NULL	8
209	Customer Feedback	2023-09-10	NULL	10
210	Procurement System	2023-10-01	NULL	9

4. Employee_Project Table

emp_id	project_id	hours_allocated
102	202	120

104	202	80
103	201	100
106	201	150
101	203	50
105	204	70
107	202	60
109	207	90
110	206	110
108	203	40

Section A: DDL (Schema Creation & Modification)

1. Write SQL statements to create the above tables with the specified constraints
2. Alter the Employees table to add a column bonus DECIMAL(8,2) with default value0.
3. Drop the column bonus from Employees.

Section B: DML (Insert, Update, Delete)

4. Insert at least 10 rows into Departments, Employees, Projects, and Employee_Project.(use the above data set)
5. Try inserting an employee with a negative salary (should fail due to CHECK constraint).
6. Update the salary of the employee with emp_id = 103 by 15%.
7. Delete an employee record who has resigned (choose any emp_id).
8. Increase all employees' salaries in the IT department by 5%.
9. Change the department of an employee to "Research".(should fail due to FK constraint)

Section C: DQL (Select Queries)

10. List all employees and their details.
11. Show all employees in the "HR" department.
12. Find employees with salaries between 50,000 and 80,000.
13. Retrieve employees hired after 2020.
14. Show employees who are in either the IT or Finance department.
15. Find employees whose email ends with "@corp.com".
16. List all employees with salary > 60,000 AND located in "New York".
17. Display employees in descending order of salary.
18. Count the number of employees in each department.
19. Show the average salary of employees department-wise.
20. Display departments where the average salary is greater than 70,000.
21. Find the number of employees in each project.
22. Display departments with more than 3 employees.
23. Show the sum of all salaries department-wise.
24. List all distinct department IDs from the Employees table.

25. Show employee names with the year they were hired.
26. Show employees grouped by the year of hire.
27. List employees hired in the last 90 days.
28. List the no of years of experience of all the employees

Section D: Joins

29. List all employees with their department names (INNER JOIN).
30. Display all departments along with employees, including those departments without employees (LEFT JOIN).
31. Show employees and the projects they are working on (JOIN 3 tables: Employees, Employee_Project, Projects).
32. List projects along with total hours allocated by employees.
33. Write a query to find employees who are working on more than one project.
34. Show all projects handled by the 'Finance' department.

Experiment 4 : PL/SQL Programming

1. Write a procedure GetEmpInfo that takes emp_id as input and displays name, salary, and department.
2. Write a PL/SQL block that checks if an employee's salary is above 50,000. If yes, print "High Salary" ;Otherwise print "Standard Salary".
3. Write a PL/SQL program to display the top 10 rows in the Emp table based on their job and salary
4. Write a stored procedure GiveBonus that takes department ID and a designation as input, along with a bonus amount, and updates the salary of all employees in that department who have the specified designation by adding the bonus amount to their current salary.
5. Create a trigger to prevent inserting employees with a salary less than 30,000.
6. Create a trigger to avoid any transactions(insert, update, delete) on the EMP table on Saturday & Sunday.

Experiment 5: MongoDB Setup and Basic Operations

- Install, configure MongoDB and Mongo Compass
- Create databases and collections, basic insertOne and insertMany operations

Experiment 6: MongoDB CRUD Operations

- Query documents using find() with filter operators
- Update and delete documents using updateOne(), updateMany(), deleteOne(), deleteMany()

Experiment 7: Data Modeling in MongoDB

- Design embedded document models for student-course system
- Design normalized data model with document references and implement

Experiment 8: Aggregation Framework

- Create aggregation pipelines with $match, $group, $project, $sort

- Use advanced operators like $lookup (joins), $unwind, $bucket

Experiment 9: Indexing and Query Optimization

- Create single field, compound, multikey, and text indexes
- Analyze query performance and optimize indexes

Experiment 10: Replication and Transactions in MongoDB

- Configure replica sets and test failover scenarios
- Implement multi-document transactions

SEMESTER-III

COURSE 7: EXPLORATORY DATA ANALYSIS & DATA VISUALIZATION

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. To introduce the concepts, importance, and workflow of exploratory data analysis.
2. To impart practical skills in data preprocessing, cleaning, and manipulation using Python.
3. To equip students with techniques in univariate, bivariate, and multivariate data analysis.
4. To develop proficiency in data visualization and interpretation using standard Python libraries.
5. To enable students to build reports and dashboards to communicate data insights effectively.

Course Outcomes

After completing this course, students will be able to:

1. Explain the importance and process of EDA in the data science pipeline.
2. Perform robust data preprocessing including missing value handling, outlier detection, transformation, and encoding.
3. Manipulate, filter, and reshape datasets using Python libraries like Pandas and NumPy.
4. Conduct data analysis and visualize results using Matplotlib, Seaborn, and Plotly.
5. Design interactive visualizations and dashboards to communicate findings in real-world scenarios.

Unit 1: Introduction to Exploratory Data Analysis (EDA)

Significance and objectives of EDA, Types and scales of data (nominal, ordinal, interval, ratio), Concepts of variability and central tendency, Role of EDA in the data science pipeline, Introduction to data quality and data issues, Differences between EDA, classical, and Bayesian analysis, Basic terminology and concepts

Unit 2: Data Preprocessing and Cleaning

Overview of data preprocessing, Handling missing types of missingness (MCAR, MAR, MNAR), Techniques for missing data imputation (mean, median, mode, forward/backward fill), Detection and treatment of duplicates, Outlier detection methods (Z-score, IQR method),

Data transformation: normalization and standardization, Encoding categorical variables: label encoding, one-hot encoding, Data integration and reduction techniques, Importance of preprocessing for machine learning models

Unit 3: Data Manipulation using Python with Pandas and NumPy

NumPy Array Creation: Use `np.array()`, `np.zeros()`, `np.ones()` to create arrays of various dimensions. **Array Manipulation:** Reshape, transpose, flatten arrays for flexible data formats. Element-wise Arithmetic Operations, Statistical Functions on NumPy Arrays

Indexing & Slicing: Access array elements via slicing, masking, and fancy indexing for selective data analysis.

Pandas DataFrames: Series and Data frames, Create from dictionaries, Series, CSV, Excel, and JSON files using Dataframe or reader functions

Data Frame Manipulation: Indexing & Selection - Label-based and Position-based indexing; conditional filtering with boolean masks, Aggregation & Grouping- Group data by columns with `.groupby()`, summarize using `.sum()`, `.mean()`, `.count()`.

Data Reshaping & Pivoting and stack/unstack hierarchical index data.

Merging & Joining Data Frames

Datetime & Categorical Data: Handle and transform temporal data with `pd.to_datetime()`, categorical conversions with `.astype('category')`.

Unit 4: Basic Data Analysis and Visualization

What is visualization, what is it's importance? Types: histograms, bar charts, Boxplots and violin plots for data distribution, Scatter plots and correlation analysis (Pearson, Spearman), Cross-tabulation and contingency tables, Analyzing relationships: correlation vs causation, Visualizing categorical vs numerical data

Unit 5: Advanced Visualization and Reporting

Principles of effective graphical representation, Introduction to visualization libraries: Matplotlib, Seaborn, Plotly, Advanced plots: heatmaps, pair plots, joint plots, Interactive visualizations and dashboards, Geographical data visualization basics, Storytelling with using visualization for communication, Introduction to reporting tools and best practices

Activities

Unit 1: Introduction to EDA

Activity:

Analyze the Titanic dataset to explore data types, central tendency, and variability.

Outcome:

Develop a foundational understanding of data structures and EDA significance.

Evaluation Method:

Lab notebook submission and oral quiz.

Unit 2: Data Preprocessing and Cleaning

Activity:

Clean a health survey dataset: identify/impute missing values, standardize columns, encode categories, remove duplicates/outliers.

Outcome:

Demonstrate proficiency in data cleaning and preprocessing techniques in Python.

Evaluation Method:

Script submission and report on preprocessing steps with before/after summary.

Unit 3: Data Manipulation using Python

Activity:

Manipulate a retail transaction dataset by indexing, filtering, grouping, merging, and handling date/time columns.

Outcome:

Apply multiple Pandas/NumPy techniques for dataset transformation.

Evaluation Method:

Graded Jupyter notebook, peer code review.

Unit 4: Basic Data Analysis and Visualization

Activity:

Explore demographic data with descriptive statistics, create histograms, boxplots, scatter plots, and cross-tabulations.

Outcome:

Visualize and interpret univariate and bivariate data insights.

Evaluation Method:

Visualization notebook and critical analysis write-up.

Unit 5: Advanced Visualization and Reporting

Activity:

- Build a dashboard on sales or air quality data using Matplotlib/Seaborn/Plotly.
- Incorporate interactive and geospatial visualizations.Outcome:
- Develop professional dashboards and communicate insights through visual storytelling.

Evaluation Method:

Final project dashboard evaluation and presentation.

Preferred Textbooks

1. Suresh Kumar Mukhiya, Usman Ahmed, "Hands-On Exploratory Data Analysis with Python", Packt Publishing, 2020.
2. Jake Vander Plas, "Python Data Science Handbook: Essential Tools for Working with Data", O'Reilly, 2017.
3. Catherine Marsh, Jane Elliott, "Exploring Data: An Introduction to Data Analysis for Social Scientists", Wiley, 2008.

References

1. Eric Pimpler, "Data Visualization and Exploration with R", GeoSpatial Training Service, 2017.
2. Claus O. Wilke, "Fundamentals of Data Visualization", O'Reilly, 2019.
3. Matthew O. Ward, Georges Grinstein, Daniel Keim, "Interactive Data Visualization: Foundations, Techniques, and Applications", CRC Press, 2015.

SEMESTER-III

COURSE 7: EXPLORATORY DATA ANALYSIS & DATA VISUALIZATION

Practical **Credits: 1** **2 hrs/week**

***Note**: This lab has to be executed using interactive computing environments like Jupyter Notebook or Google Colab.*

1. Importing Data and Basic Exploration
   - Load datasets from CSV/Excel/JSON
   - Use `.head()`, `.info()`, `.describe()`, `.shape` to examine structure and summary statistics
2. Examining Data Types and Missing Values
   - Identify data types for each column
   - Detect and quantify missing values with `.isnull()` and `.sum()`.
3. Handling Missing Data Techniques
   - Impute missing values using mean, median, mode, forward, and backward fill
   - Drop rows/columns with missing data and compare results
4. Dealing With Duplicates and Outliers
   - Find and remove duplicate entries
   - Detect outliers using Z-score or IQR methods
   - Treat or remove outliers and compare distributions
5. Data Transformation and Scaling
   - Normalize, standardize, log-transform columns
   - Encode categorical variables (Label Encoding, One-Hot Encoding)
6. Indexing, Filtering, and Slicing DataFrames
   - Select specific rows/columns using index and Boolean filters
   - Conditional selections and advanced slicing
7. Aggregating, Grouping, and Pivoting Data
   - Use `.groupby()` for aggregate statistics
   - Create pivot tables for categorical summaries
8. Merging and Concatenating Multiple Datasets
   - Join, merge, concatenate DataFrames
   - Handle merging keys and missing arguments
9. Univariate Visualization Techniques
   - Create histograms, boxplots, violin plots for distributions

- Visualize frequency and central tendency

10. Bivariate and Multivariate Visualization
    - Display scatter plots, correlation matrices, pair plots
    - Visualize relationships and patterns
11. Time Series Data Exploration
    - Parse and index datetime columns
    - Visualize trends and seasonality
12. Interactive Visualizations and Dashboards
    - Build simple interactive graphs with Plotly
    - Design a dashboard showing multiple aspects of a dataset

SEMESTER-IV

COURSE 8: DATA SCIENCE WITH R

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. Introduce the data science process, lifecycle, and applications in real-world domains.
2. Build proficiency in R programming for data manipulation, exploration, and visualization.
3. Train students in handling structured, unstructured, and time-based data effectively.
4. Familiarize with basic machine learning and statistical modeling using R.
5. Develop awareness of ethical, interpretability, and responsible use of data science.

Course Outcomes

At the end of the course, students will be able to:

1. Explain the Data Science process and perform EDA (Exploratory Data Analysis).
2. Write R programs using variables, functions, loops, and packages for basic analytics.
3. Perform data wrangling, cleaning, and visualization with R libraries (dplyr, tidyr, ggplot2).
4. Build and evaluate basic machine learning models such as regression and clustering.
5. Apply data science techniques to practical case studies.

Unit 1. Introduction to Data Science Process:

Introduction- Definition - Data Science in various fields - Examples - Impact of Data Science - Data Analytics Life Cycle - Data Science Toolkit - Data Scientist - Data Science Team, Exploratory Data Analysis (EDA), Feature Engineering & Data Transformation

Unit 2. Basics of R Programming:

Introduction to R and RStudio, Data Types, Variables, Operators, Control Structures (if, loops, apply), Functions and Packages, Data Input/Output (CSV, Excel, XML, JSON).

Unit 3. Data Handling & Visualization in R:

Data Frames, Lists, Matrices, Data Wrangling with dplyr and tidyr, Handling Missing Data, Working with Date/Time in R. Visualization with ggplot2: grammar of graphics, aesthetics, geometries, scales.
Faceting and layering techniques,Visualizing categorical and numerical data, Customizing and exporting plots

Unit 4. Applications & Case Studies in Data Science:
Simple Linear Regression, Multiple Regression
Model Evaluation Method: Accuracy, Confusion Matrix, ROC.
K-Means Clustering, Text Mining & Word Clouds, Recommender Systems Basics, Ethical Issues in Data Science

Unit 5. Advanced Topics in Data Science with R :
Introduction to Time Series Analysis in R (ARIMA basics)- Concept of time series (trend, seasonality, noise), Time series objects in R (ts, zoo, xts), Plotting and decomposing time series, Stationarity and differencing, Autocorrelation & Partial Autocorrelation (ACF/PACF), AR, MA, ARIMA model basics, Forecasting using forecast package
Creating interactive visualizations with plotly packages-Converting ggplot2 plots to interactive plots
Animations and sliders in plotly

R Shiny: Building interactive web applications-Introduction to Shiny framework, UI and server functions, Reactive expressions and reactivity in Shiny, Input and output widgets (sliders, dropdowns, text), Layouts and dashboard design

Textbooks

1. An Introduction to Statistical Learning with Applications in R, Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, Springer, 2nd Edition, 2021
2. R for Data Science, Hadley Wickham and Garrett Grolemund, O'Reilly Media, 2017.

Reference Books

1. The Art of R Programming, Norman Matloff,, No Starch Press, 2011.
2. Modern Applied Statistics with S, W.N. Venables & B.D. Ripley, Springer, 2002.
3. Introduction to Data Science: Data Analysis and Prediction Algorithms with R, Rafael A. Irizarry, CRC Press, 2020.
4. Data Science from Scratch: First Principles with Python (for conceptual clarity only), Joel Grus,

Activities:
Outcome: Explain the Data Science process and perform EDA (Exploratory Data Analysis).
Activity: Use a real-world dataset (e.g., Titanic or COVID data) to:

- Outline the steps of the Data Science workflow
- Perform EDA using summary statistics and visualizations (histograms, boxplots, scatterplots)

Evaluation Method: Presentation and checklist (10-point scale):

- Clear explanation of workflow stages
- Quality of EDA insights
- Use of appropriate plots and summaries

Outcome: Write R programs using variables, functions, loops, and packages for basic analytics.

Activity: Write an R script that:

- Reads a CSV file
- Uses if, for, and while loops
- Defines and calls custom functions with arguments and return values

Evaluation Method: Code review and execution test to verify (10-point scale):

- Correctness of the syntax and logic
- Functionality of control structures
- Output accuracy and modularity

Outcome: Perform data wrangling, cleaning, and visualization with R libraries (dplyr, tidyr, ggplot2).

Activity: Clean a messy dataset using:

- dplyr for filtering, selecting, and mutating
- tidyr for reshaping and handling missing values
- Time-based operations (e.g., filling gaps, formatting dates)

Evaluation Method: Before-and-after comparison (10 point score):

- Completeness of cleaning steps
- Use of appropriate functions
- Handling of missing/time data

Outcome: Implement basic machine learning models and evaluate performance using appropriate metrics and visual tools.

Activity: Build a simple classification model (e.g., logistic regression or decision tree) using R:

- Train/test split
- Predict outcomes
- Evaluate using confusion matrix, accuracy, precision, recall

Evaluation Method: Model report and demo (10 point scale):

- Correct implementation of model
- Use of evaluation metrics

SEMESTER-IV

COURSE 8: DATA SCIENCE WITH R

Practical **Credits: 1** **2 hrs/week**

List of Practicals:

1. Compute Mean, Median, Mode, Variance, and Standard Deviation
2. Visualize Binomial, Normal, and Poisson Distributions
3. Perform t-test and Chi-Square Test in R
4. Calculate Correlation and Build a Simple Linear Regression Model
5. Conduct Exploratory Data Analysis (EDA) on a Real-World Dataset
6. Apply Feature Engineering: Scaling, Normalization, and Encoding
7. Practice R Programming: Variables, Control Structures, and Functions
8. Read and Write Data from CSV, Excel, JSON, and XML Files
9. Use dplyr and tidyr for Data Wrangling Tasks
10. Handle Missing Data and Detect Outliers
11. Work with Dates and Times in R
12. Visualize Data Using ggplot2 (Bar, Scatter, Histogram, Boxplot)
13. Perform K-Means Clustering and Visualize Clusters
14. Evaluate Models Using Confusion Matrix, Accuracy, and ROC Curve
15. Perform Text Mining and Create a Word Cloud
16. Time Series Forecasting with ARIMA on a real dateset (e.g., monthly airline passengers, stock prices, or temperature data).
17. Create interactive bar, line, and scatter plots using plotly. On a real dataset (e.g., COVID-19 cases, sales data, or student marks).
18. Develop a Shiny app that lets users upload a CSV file.

SEMESTER-IV

COURSE 9: FOUNDATION OF ML & SUPERVISED MACHINE LEARNING

Theory **Credits: 3** **3 hrs/week**

Course Objectives

- Provide a comprehensive understanding of supervised machine learning algorithms and their theoretical foundations.
- Enable students to formulate machine learning problems and implement algorithms in Python.
- Introduce fundamental learning theory concepts such as hypothesis, VC dimension, PAC learning, Find-S, and candidate elimination.
- Develop practical skills in model evaluation, tuning, and application to real-world datasets.
- Instill ethical awareness and explainability considerations in supervised learning.

Course Outcomes

On successful completion, students will be able to:

- Describe core supervised learning algorithms and their mathematical foundations.
- Implement and evaluate linear and nonlinear supervised learning models using Python.
- Apply learning theory fundamentals to analyze model complexity and generalization.
- Perform model selection, cross-validation, and hyperparameter tuning.
- Develop end-to-end supervised learning solutions for real-life problems.
- Uphold ethical standards and interpret model results critically.

Unit 1: Introduction to Supervised Machine Learning

What is Machine Learning & Supervised Machine Learning ? - Definitions

Machine learning pipeline: data collection, cleaning, splitting, feature selection

Loss functions, cost functions, bias-variance tradeoff

Performance metrics: accuracy, precision, recall, F1 score, ROC-AUC

Introduction to Python libraries (scikit-learn, pandas, matplotlib) for ML

Learning Theory Fundamentals: Hypothesis and hypothesis space, Concept learning: types of concepts, Find-S algorithm, Candidate Elimination algorithm: overview and intuition, VC dimension, PAC (Probably Approximately Correct) learning

Unit 2: Linear Models for Regression and Classification

Introduction to regression problems and applications

Simple linear regression: theory, assumptions, ordinary least squares

Gradient Descent for training linear models,

Multiple and polynomial regression models

Regularization techniques: Ridge, Lasso, Elastic Net

Logistic regression: sigmoid function, decision boundary, training using gradient descent

Evaluation metrics: RMSE, MAE, R^2, confusion matrix

Unit 3: Classification Algorithms

Concept of classification and categorical data, Decision Trees, Naive Bayes algorithm, Instance based learning, K-Nearest Neighbors (KNN), Support Vector Machines (SVM) with kernel tricks, **Evaluation metrics:** confusion matrix, ROC curves, cross-validation

Unit 4: Model Selection, Tuning, and Real-World Applications

Data preprocessing: imputation, scaling, encoding categorical variables, Feature engineering and selection techniques, Train-test split, k-fold cross-validation, Hyperparameter tuning: grid search, random search, Case studies in finance, healthcare, retail, Ethics and explainability in supervised learning

Unit 5: Ensemble Learning Methods

Bagging, boosting, stacking, Random Forests: theory, applications, feature importance , Gradient Boosting Machines (GBM), AdaBoost, XGBoost, Comparing ensembles versus single models

Avoiding overfitting: cross-validation, hyperparameter tuning

Activities:

Unit 1: Introduction and Learning Theory Fundamentals

Activities:

- Lecture and discussion on ML fundamentals and pipeline
- Concept learning exercises: Hypothesis space, Find-S, Candidate Elimination
- Concept quizzes on VC Dimension and PAC Learning
- Python intro: Data loading and basic evaluation metrics

Outcomes:

- Understand ML basics and key learning theory concepts.
- Illustrate concept learning algorithms and hypothesis characterization.

Evaluation Method:

- Quiz on theory concepts: 8%
- Assignment on implementing Find-S algorithm: 7%

Unit 2: Linear Models for Regression and Classification

Activities:

- Coding lab: Linear regression and logistic regression models in Python
- Visualization of model fits and decision boundaries
- Experiment with gradient descent and regularization parameters

Outcomes:

- Implement and interpret linear regression and logistic regression.
- Explain optimization and regularization for generalization.

Evaluation Method:

- Practical lab report: 10%
- Coding assignment (regression/classification): 7%

Unit 3: Classification Algorithms

Activities:

- Implement KNN, Decision Trees, SVM, and Naive Bayes classifiers
- Performance evaluation via confusion matrix, ROC, and cross-validation
- Group discussion on algorithm strengths and weaknesses

Outcomes:

- Train and evaluate various classification algorithms.
- Select appropriate classifiers for varied contexts.

Evaluation Method:

- Lab submission and class presentation: 12%

Unit 4: Model Selection, Tuning, and Applications

Activities:

- Feature engineering workshop and data preprocessing in Python
- Hyperparameter tuning using GridSearch and RandomizedSearch
- Mini project based on real-world dataset with ethical evaluation
- Presentation and peer review of project findings

Outcomes:

- Master data preparation, model tuning, and performance assessment.
- Demonstrate a full ML pipeline with ethical considerations.

Evaluation Method:

- Mini project and presentation: 16%
- Final written exam: 20%

Unit 5: Ensemble Learning Methods

Activities:

- Ensemble methods lab: Random Forest, AdaBoost, and XGBoost implementation
- Hyperparameter tuning exercises and cross-validation applications
- Case study: Comparing ensemble vs base models

Outcomes:

- Develop ensemble models and understand boosting/bagging concepts.
- Apply model tuning to improve performance.

Evaluation Method:

- Practical lab and report: 10%

Recommended Textbooks & References

- Tom M. Mitchell, Machine Learning, McGraw Hill International Edition
- Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer
- Aurélien Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, O'Reilly
- Richard O. Duda, Peter E. Hart, David G. Stork, Pattern Classification, Wiley
- Online: scikit-learn documentation, Coursera Machine Learning courses, Kaggle tutorials

SEMESTER-IV

COURSE 9: FOUNDATION OF ML & SUPERVISED MACHINE LEARNING

Practical **Credits: 1** **2 hrs/week**

***Note**: This lab has to be executed using interactive computing environments like Jupyter Notebook or Google Colab.*

1. Data Loading, Exploration, and Preprocessing
   - Load datasets (e.g., Iris, Boston Housing) using pandas
   - Perform exploratory data analysis: summary statistics, visualization
   - Handle missing data, categorical encoding, feature scaling
2. Implement Simple Linear Regression
   - Fit linear regression to sample data
   - Visualize regression line and residuals
   - Compute evaluation metrics (RMSE, R^2)
3. Gradient Descent from Scratch
   - Implement gradient descent algorithm for linear regression
   - Experiment with learning rate and convergence
4. Logistic Regression for Classification
   - Train logistic regression model on binary classification dataset
   - Evaluate classification metrics: confusion matrix, accuracy, ROC curve
5. Decision Tree Classifier
   - Train decision tree on classification datasets
   - Visualize tree structure
   - Analyze feature importance and pruning effects
6. Naive Bayes Classifier
   - Train Naive Bayes on text or classification datasets
   - Performance comparison with other classifiers
7. K-Nearest Neighbors (KNN) Algorithm
   - Implement KNN classification using scikit-learn
   - Experiment with different values of k
   - Visualize decision boundaries
8. Support Vector Machine (SVM) Classifier

- Train SVM with linear and RBF kernel
- Understand margin and support vectors
- Parameter tuning and performance evaluation

9. Model Evaluation and Hyperparameter Tuning
   - Perform train-test split and K-fold cross-validation
   - Use GridSearchCV or RandomizedSearchCV
   - Evaluate final model performance on unseen data
10. Random Forest and Ensemble Methods
   - Train Random Forest classifier
   - Understand bagging and feature randomness
   - Compare ensemble performance with base classifiers
11. Gradient Boosting Machine (GBM)
   - Train GBM model using popular libraries (XGBoost or sklearn)
   - Visualize feature importance

SEMESTER-IV

COURSE 10: ROBOTICS PRINCIPLES AND EMBEDDED SYSTEMS

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. To introduce fundamental concepts of robotics including robot classification, kinematics, and mechanisms.
2. To explain the roles and types of actuators, drives, and sensors, including piezoelectric technologies.
3. To develop an understanding of robot control systems, feedback mechanisms, and intelligent control methods.
4. To expose students to embedded systems, focusing on microcontrollers and interfacing with hardware components.
5. To provide practical experience in Arduino programming and hardware interfacing for robotics applications.

Course Outcomes

Upon successful completion of this course, students will be able to:

1. Classify different types of robots and explain their kinematic principles and degrees of freedom.
2. Describe actuators, drives, and sensor technologies used in robotics, including piezoelectric actuators and sensors.
3. Design and analyze control systems for robotic joints with feedback and intelligent control strategies.
4. Understand embedded systems concepts and program microcontroller-based systems (Arduino) for robotic applications.
5. Develop and implement practical robotics solutions using Arduino hardware and software tools.

Unit 1: Introduction to Robotics

Classification of Robots, Components and Characteristics of Robots, Applications of Robotics, Degrees of Freedom: Definition, importance, calculation, examples

Fundamentals of Kinematics and Position Analysis, Robots as Mechanisms, Matrix Representation and Transformation Matrices, Forward and Inverse Kinematics

Unit 2: Actuators and Drives

Actuating Systems: Features, selection criteria, **Actuators:** Types- electric, hydraulic, pneumatic, **Drives:** Types and characteristics-electric, hydraulic, pneumatic, stepper, servo; comparison and usage, **Piezoelectric actuators:** Principle, applications in microrobotics, advantages (precision, quick response, energy harvesting) , Use of Reduction Gears, Actuating Devices and Control

Unit 3: Grippers and Sensors

Grippers: Types (mechanical, magnetic, vacuum, adhesive), applications, Criteria for gripper selection, Sensor characteristics, Description of Vision, Force, Proximity, Tilt Sensors, Piezoelectric sensors (Working principle, parts, & applications), types (force, vibration, pressure), applications of robotics, Principles and practical applications of key sensor types

Unit 4: Robot Controls

Point-to-Point and Continuous Path Control, Intelligent Robot Control Methods, Control Systems for Robot Joints, Control Actions, **Feedback Devices:** Concepts and significance, Integration of sensors in feedback and control loops (e.g., vibration damping, tactile feedback, health monitoring), Closed-loop vs. Open-loop Control Systems

Unit 5: Embedded Systems

Embedded Systems: Definition, characteristics, and significance in robotics, Microcontrollers and Microprocessors, Arduino microcontroller architecture and applications, Interfacing sensors, actuators, drives, and grippers with embedded systems, Embedded programming essentials (C/C++, input/output, analog/digital signals), Embedded communication protocols (serial, I2C basics), Integration of sensor data in embedded applications.

Unit 1: Introduction to Robotics

Activity: Analyze and classify various types of robots; calculate degrees of freedom for given robot models.

Outcome: Students will understand robot types, components, and the significance of degrees of freedom.

Evaluation Method: Written assignments and quizzes on robot classification and kinematics concepts.

Unit 2: Actuators and Drives

Activity: Investigate characteristics of electric, hydraulic, pneumatic actuators and drives; study piezoelectric actuator demonstrations.

Outcome: Ability to compare different actuators and drives and understand piezoelectric actuator applications.

Evaluation Method: Lab demonstration reports and presentation on actuators.

Unit 3: Grippers and Sensors

Activity: Examine different types of grippers; conduct hands-on experience with piezoelectric, proximity, force, and vision sensors.

Outcome: Understand gripper types, sensor characteristics, and apply piezoelectric sensing principles.

Evaluation Method: Practical sensor interfacing reports and sensor data analysis.

Unit 4: Robot Controls

Activity: Implement point-to-point and continuous path control algorithms; integrate sensor feedback for closed-loop control.

Outcome: Develop understanding of control methods and feedback systems in robotics.

Evaluation Method: Lab assignments implementing control algorithms on robotic joints; tests on control theory.

Unit 5: Embedded Systems

Activity: Program Arduino microcontrollers to interface sensors, actuators, and drives; develop embedded control applications.

Outcome: Ability to design embedded systems and program Arduino for real-world robotics applications.

Evaluation Method: Project work based on Arduino robotics applications, demonstration and code review.

Recommended Textbooks & References

1. "Robotics: Mechanics and Control" by Guruprasad K. Eastern Economy Edition
2. "Modern Robotics: Mechanics, Planning, and Control" by Kevin M. Lynch and Frank C. Park, Cambridge University Press.

3. “Introduction to Autonomous Mobile Robots” by Roland Siegwart et al. (Free e-Resource)
4. “Arduino Programming in 24 Hours” by Richard Blum, Sams Teach Yourself
5. “Springer Handbook of Robotics” edited by Bruno Siciliano and Oussama Khatib.

SEMESTER-IV

COURSE 10: ROBOTICS PRINCIPLES AND EMBEDDED SYSTEMS

Practical **Credits: 1** **2 hrs/week**

Note: This lab has to be completed using Arduino Micro controller and Arduino programming.

1. Introduction to Arduino and Basic Programming
   - Setting up Arduino IDE
   - Blinking an LED — basic digital output
2. Reading Digital Inputs
   - Interfacing and reading button presses/switches
3. Interfacing Analog Sensors
   - Reading values from a potentiometer or light-dependent resistor (LDR)
4. Ultrasonic Distance Sensor Interfacing
   - Measuring distance using HC-SR04 sensor and displaying data
5. Servo Motor Control
   - Controlling servo motor angles for robotic joint simulation
6. DC Motor and Motor Driver Interfacing
   - Running a DC motor with L298N motor driver
7. Stepper Motor Control
   - Basic control of stepper motors for precise movements
8. Line Following Robot Implementation
   - Using IR sensors to detect and follow a line on the ground
9. Obstacle Avoidance Robot
   - Using ultrasonic sensors and motor control to avoid obstacles
10. Force/Tactile Sensor Interfacing (including Piezo sensors)
    - Reading piezoelectric sensor signals as force or vibration inputs
11. Temperature Sensor Interfacing
    - Using sensors like LM35 or DHT11 for environmental sensing
12. Bluetooth Module (HC-05) Controlled Robot
    - Controlling robot movement wirelessly using smartphone commands
13. Servo-Driven Robotic Gripper
    - Building and controlling a simple gripper mechanism
14. PWM Motor Speed Control
    - Controlling motor speed using Pulse Width Modulation techniques

SEMESTER-V

COURSE 11: UNSUPERVISED & REINFORCEMENT LEARNING

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. To understand the fundamental concepts and techniques of unsupervised learning.
2. To study advanced unsupervised learning models including clustering and generative models.
3. To grasp the foundational theory of reinforcement learning, including Markov Decision Processes.
4. To explore model-free reinforcement learning algorithms and the Multi-Armed Bandit problem.
5. To learn advanced reinforcement learning techniques and their real-world applications.

Course Outcomes

By the end of the course, students will be able to:

1. Explain key concepts and algorithms in unsupervised learning and perform clustering and dimensionality reduction.
2. Implement advanced unsupervised learning techniques like Gaussian Mixture Models and autoencoders.
3. Formulate reinforcement learning problems using MDPs and apply dynamic programming methods.
4. Apply model-free reinforcement learning algorithms including Q-learning and SARSA, and understand Multi-Armed Bandit solutions.
5. Analyze advanced reinforcement learning methods such as Actor-Critic and policy gradients, recognizing their applications.

Unit 1: Introduction to Unsupervised Learning

Overview and motivation of unsupervised learning, Types of unsupervised learning: clustering, dimensionality reduction, Basic clustering algorithms: k-means, hierarchical clustering, Evaluation of clustering results, Singular value decomposition, Introduction to principal component analysis (PCA), Applications and challenges of unsupervised learning

Unit 2: Advanced Unsupervised Learning Techniques

Gaussian Mixture Models and Expectation-Maximization, Density-based clustering (DBSCAN), Intro to autoencoders for feature extraction, Anomaly detection methods, Introduction to generative models (GANs, VAEs)

Unit 3: Foundations of Reinforcement Learning

Reinforcement learning basics: agent, environment, states, actions, rewards, Markov Decision Process (MDP), Policy, value functions, and Bellman equations, Exploration vs exploitation, Dynamic programming methods: policy evaluation and iteration

Unit 4: Model-Free Reinforcement Learning and Multi-Armed Bandits

Monte Carlo methods, Temporal Difference Learning, Q-learning, SARSA, Introduction to Multi-Armed Bandit Problem, Exploration-exploitation strategies: epsilon-greedy, UCB, Thompson Sampling, Application of bandit algorithms in real-world problems, Relation of Multi-Armed Bandit to Reinforcement Learning

Unit 5: Advanced Reinforcement Learning Concepts and Applications

Actor-Critic algorithms and policy gradients, Hierarchical and multi-agent reinforcement learning (overview), Inverse reinforcement learning and imitation learning, Real-world applications and case studies

Unit 1: Introduction to Unsupervised Learning

Activity: Apply k-means and hierarchical clustering on datasets; evaluate clustering results and perform PCA.

Outcome: Ability to implement and analyze basic unsupervised learning algorithms.

Evaluation Method: Practical assignment on clustering and PCA; quiz on concepts.

Unit 2: Advanced Unsupervised Learning Techniques

Activity: Implement Gaussian Mixture Models, DBSCAN, and simple autoencoders; explore anomaly detection with datasets.

Outcome: Understand and apply advanced unsupervised learning methods and feature extraction.

Evaluation Method: Lab reports on GMM, DBSCAN, and autoencoders; conceptual tests.

Unit 3: Foundations of Reinforcement Learning

Activity: Model MDPs for simple problems; solve policy evaluation problems; discuss exploration-exploitation.

Outcome: Grasp reinforcement learning framework and MDP formulation fundamentals.

Evaluation Method: Written exercises on MDPs; online quizzes; class discussions.

Unit 4: Model-Free Reinforcement Learning & Multi-Armed Bandits

Activity: Implement Q-learning and SARSA algorithms on grid world; simulate multi-armed bandit strategies like epsilon-greedy and UCB.

Outcome: Apply model-free RL methods and bandit algorithms in practice.

Evaluation Method: Code submissions; comparative analysis reports; quizzes.

Unit 5: Advanced Reinforcement Learning Concepts and Applications

Activity: Implement simple Actor-Critic algorithms; case studies on multi-agent RL or inverse reinforcement learning; present application reports.

Outcome: Gain knowledge of advanced RL methods and their practical relevance.

Evaluation Method: Project presentations; peer review; final exam questions.

Recommended Textbooks & References

- "Reinforcement Learning: An Introduction" by Richard S. Sutton and Andrew G. Barto, 2nd Edition, MIT Press, 2018.
- "Pattern Recognition and Machine Learning" by Christopher M. Bishop, Springer, 2006 (for unsupervised learning).
- "Machine Learning: A Probabilistic Perspective" by Kevin P. Murphy, MIT Press, 2012.
- Relevant research papers and articles on Multi-Armed Bandits and Deep RL (provided by instructor).
- Online resources and tutorials (e.g., OpenAI Gym, TensorFlow Agents).

SEMESTER-V

COURSE 11: UNSUPERVISED & REINFORCEMENT LEARNING

Practical **Credits: 1** **2 hrs/week**

1. Data Exploration and Preprocessing for Unsupervised Learning
   - Perform data normalization, scaling, and visualize distributions for clustering tasks.
2. K-Means Clustering Implementation
   - Apply k-means clustering algorithm on a dataset and analyze cluster quality.
3. Hierarchical Clustering and Dendrogram Analysis
   - Perform agglomerative and divisive hierarchical clustering and interpret dendrograms.
4. DBSCAN Clustering for Density-Based Grouping
   - Use the DBSCAN algorithm on real-world data to identify clusters and noise points.
5. Principal Component Analysis (PCA)
   - Apply PCA for dimensionality reduction and visualize data in reduced dimensions.
6. Autoencoder for Feature Extraction
   - Build a simple autoencoder using neural networks for unsupervised feature learning.
7. Gaussian Mixture Models and Expectation-Maximization
   - Implement GMM clustering and compare with k-means results.
8. Multi-Armed Bandit Problem Simulation
   - Simulate epsilon-greedy, UCB, and Thompson Sampling algorithms to balance exploration and exploitation.
9. Q-Learning Algorithm Implementation
   - Implement Q-learning for a simple environment (e.g., Grid World) and find optimal policies.
10. SARSA Algorithm for Reinforcement Learning
    - Compare SARSA with Q-learning on the same environment.
11. Policy Gradient Method Exercise
    - Implement a basic policy gradient method with a simple environment.
12. Reinforcement Learning Case Study using gym library
    - Apply RL algorithms to a case study like Tic-Tac-Toe, CartPole, or FrozenLake.

SEMESTER-V

COURSE 12 A: NEURAL NETWORKS & DEEP LEARNING

Theory **Credits: 3** **3 hrs/week**

Course Objectives:

1. Introduce the fundamental concepts of Artificial Neural Networks and Deep Learning, along with their historical and biological inspirations.
2. Provide an in-depth understanding of different neural network architectures including Perceptron, DNN, CNN, RNN, and advanced models.
3. Develop hands-on skills to design, train, and evaluate deep learning models using popular frameworks such as TensorFlow and Keras.
4. Expose students to applications of deep learning in computer vision, natural language processing, and generative modeling.
5. Enable students to critically analyze challenges in deep learning such as overfitting, bias, and ethical concerns.

Course Outcomes:

After successful completion of this course, students will be able to:

1. Explain the principles of neural networks, perceptrons, activation functions, and the evolution of deep learning.
2. Apply concepts of forward/backward propagation, weight initialization, and optimization techniques to train deep neural networks.
3. Design and implement convolutional neural networks (LeNet, AlexNet, VGG) for image classification tasks.
4. Build and analyze recurrent neural networks (RNN, LSTM, GRU) for sequential data and natural language processing applications.
5. Experiment with advanced deep learning concepts such as transfer learning, generative models, and transformers using pre-trained models.

Unit 1. Foundations of Deep Learning:

What is Artificial Intelligence, Machine Learning, and Deep Learning? History and applications of deep learning, Biological vs. Artificial Neurons

Introduction to Neural Networks, Perceptron and activation functions (Linear, ReLU, Sigmoid, Tanh, Softmax), Types of Neural Networks (shallow vs. deep, feedforward vs.

recurrent), Gradient descent and backpropagation (conceptual only), Concept of loss functions (MSE, cross-entropy) at intuitive level

Unit 2. Deep Neural Networks:

Forward and backward propagation, Weight initialization, learning rate, and optimization algorithms (SGD, Adam, RMSProp), Overfitting & underfitting: Regularization, Dropout, Batch normalization, Activation functions in deep networks, Loss functions in detail (MSE, cross-entropy, hinge loss) Introduction to Keras/TensorFlow framework

Unit 3. Convolutional Neural Networks (CNNs):

Introduction to images and pixels, Filters/kernels, padding, and pooling, CNN architecture and layers (Conv, Pooling, Fully Connected, Softmax), Classical CNN architectures: LeNet-5 (digit recognition - first CNN model), AlexNet (ImageNet breakthrough - deeper CNN), VGG (concept of depth, simplicity)

Applications in image classification, object detection, facial recognition

Unit 4. Recurrent Neural Networks (RNNs) and NLP

Sequences and time series data, Introduction to RNNs: vanishing/exploding gradient issue LSTM and GRU (intuitive and architectural view), Word embeddings: Word2Vec, GloVe, introduction to contextual embeddings (BERT at high level)

Applications: Sentiment analysis, text generation, simple time-series forecasting

Unit 5. Advanced & Emerging Topics:

Generative models: GANs (Generator & Discriminator intuition), VAEs (introduction only), Transformers: attention mechanism (intuitive), BERT, GPT family (overview), Transfer learning & fine-tuning pre-trained models (vision & NLP), AI ethics: Bias, fairness, privacy, safety, explainability

Textbooks:

1. Chollet, F. (2018). *Deep Learning with Python* (1st ed.). Manning Publications.
2. Nielsen, M. A. (2015). *Neural Networks and Deep Learning*. Determination Press. *(Available free online: http://neuralnetworksanddeeplearning.com)*

Reference Books:

1. Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* (2nd ed.). O'Reilly Media.
2. Howard, J., & Gugger, S. (2020). *Practical Deep Learning for Coders*. O'Reilly Media. *(Based on the fast.ai course)*
3. Shane, J. (2019). *You Look Like a Thing and I Love You: How AI Works and Why It's Making the World a Weirder Place*. Voracious / Hachette Book Group.

Activities:

Outcome: Explain the principles of neural networks, perceptrons, activation functions, and the evolution of deep learning.

Activity: *Concept Mapping Exercise* - Students work in small groups to draw a timeline-based concept map linking biological neurons → perceptron → multilayer perceptron → activation functions → deep learning.

Evaluation Method: Rubric-based evaluation of concept maps (clarity, correctness, and depth of connections).

Outcome: Apply concepts of forward/backward propagation, weight initialization, and optimization techniques to train deep neural networks.

Activity: *Hands-on Coding Task* - Students implement a simple 3-layer neural network from scratch in Python/NumPy (without Keras/TensorFlow) to observe propagation and optimization effects.

Evaluation Method: Code demonstration + oral viva where students explain how changing initialization/learning rate affects training.

Outcome: Design and implement convolutional neural networks (LeNet, AlexNet, VGG) for image classification tasks.

Activity: *Mini Project (Image Classification)* - Students form teams to train CNNs (LeNet, AlexNet, VGG) on a dataset like MNIST/CIFAR-10 and compare results.

Evaluation Method: Project report + presentation with accuracy comparison, confusion matrix, and discussion of design choices.

Outcome: Build and analyze recurrent neural networks (RNN, LSTM, GRU) for sequential data and natural language processing applications.

Activity: *Case Study on Sentiment Analysis* - Students use IMDB reviews dataset to build RNN/LSTM/GRU models for sentiment classification and analyze performance.
Evaluation Method: Submission of case study report with experimental setup, evaluation metrics (accuracy/F1-score), and reflection on differences among models.

Outcome: Experiment with advanced deep learning concepts such as transfer learning, generative models, and transformers using pre-trained models.
Activity: *Model Exploration & Demonstration* - Students choose one advanced technique (Transfer Learning on ResNet, GAN for image generation, or Transformer for text classification) and prepare a live demo in class.
Evaluation Method: Evaluation of demo + short reflective note (1–2 pages) on challenges, benefits, and application potential of the chosen technique.

SEMESTER-V

COURSE 12 A: NEURAL NETWORKS & DEEP LEARNING

Practical **Credits: 1** **2 hrs/week**

1. Build a perceptron from scratch in Python
2. Use Google Teachable Machine or TensorFlow Playground
3. Visualize various Activation Functions and their Gradients
4. Build and train a deep neural network for classification (e.g., MNIST digits)
5. Experiment with dropout, batch normalization, and different activations
6. Train a CNN to classify fashion images (Fashion-MNIST)
7. Visualize filters and feature maps
8. Fine-tune a pre-trained CNN (MobileNet, VGG) for a small dataset
9. Build an LSTM model for movie review sentiment analysis (IMDb dataset)
10. Generate text using a simple character-level RNN
11. Use a pre-trained model (like MobileNet or BERT) for a simple task
12. Use Huggingface to deploy a Sentiment Analysis App for Swiggy Reviews

SEMESTER-V

COURSE 12 B: BIG DATA TECHNOLOGIES

Theory **Credits: 3** **3 hrs/week**

Course Objectives:

1. Introduce students to the concepts, characteristics, and challenges of Big Data.
2. Familiarize students with the Hadoop ecosystem and its core components (HDFS, YARN, MapReduce).
3. Develop practical knowledge of distributed storage and parallel processing in Hadoop.
4. Provide hands-on exposure to data ingestion tools (Sqoop, Flume) and serialization techniques.
5. Enable students to explore NoSQL databases (HBase), coordination services (ZooKeeper), and Hadoop–Spark integration for large-scale data analysis.

Course Outcomes:

At the end of this course, students will be able to:

1. Explain Big Data concepts and challenges along with the role of the Hadoop ecosystem.
2. Demonstrate understanding of HDFS and YARN architectures and their functions in distributed data management.
3. Apply MapReduce and high-level tools (Hive, Pig, Spark) to process and analyze large datasets.
4. Design and implement data ingestion workflows using Sqoop, Flume, and serialization formats like Avro and Parquet.
5. Utilize NoSQL databases and ecosystem enhancements such as HBase, ZooKeeper, and Hadoop–Spark integration for scalable big data solutions.

Unit 1. Foundations of Big Data & Hadoop Ecosystem

Introduction to Big Data: characteristics (volume, variety, velocity, veracity, value), Hadoop Ecosystem Overview: HDFS, MapReduce, YARN, Hadoop Common, Hadoop architecture and use cases

Unit 2. Hadoop Distributed File System (HDFS) & YARN:

Deep dive into HDFS architecture: blocks, NameNode, DataNodes, HDFS file operations, fault tolerance, replication

YARN architecture: ResourceManager, NodeManager, application scheduling

Unit 3. MapReduce & High-Level Tools

MapReduce programming model: map, shuffle, reduce phases, Writing MapReduce applications in Hadoop

High-level abstractions: Hive, Pig, Crunch, and introduction to Spark integration

Unit 4. Data Ingestion & Serialization

Data ingestion pipelines: Sqoop (for RDBMS), Flume (streaming), Data formats & serialization: Avro, Parquet, SequenceFile, Practical ingestion workflows-batch and streaming

Unit 5. NoSQL & Ecosystem Enhancements

Overview of NoSQL within Hadoop ecosystem: HBase, Configuration and usage of ZooKeeper for coordination, Hadoop integration with Spark for data processing

Textbooks

1. Hadoop: The Definitive Guide, Tom White, 4th Edition, O'Reilly
   Free Resource available at piazza-resources.s3.amazonaws.comO'Reilly Media
2. Learning Spark, 2nd Edition, Jules S. Damji, Brooke Wenig, Tathagata Das, Denny Lee, O'Reilly

Reference Books

3. BIG DATA, Black Book TM, DreamTech Press, 2016 Edition.
4. BIG DATA and ANALYTICS, Seema Acharya, SubhasniChellappan , Wiley publications, 2016

Activities:

Outcome: Explain Big Data concepts and Hadoop ecosystem

Activity: Short seminar / presentation on Big Data applications

Evaluation Method: Oral presentation + concept quiz

Outcome: Demonstrate HDFS and YARN architectures

Activity: Hands-on lab to configure HDFS & analyze NameNode/DataNode logs

Evaluation Method: Lab performance + viva

Outcome: Apply MapReduce and high-level tools

Activity: Mini-project implementing MapReduce job and Hive queries

Evaluation Method: Project report + execution demo

Outcome: Design and implement data ingestion workflows

Activity: Lab task to ingest data using Sqoop/Flume and serialize with Avro/Parquet

Evaluation Method: Lab record + output validation

Outcome: Utilize NoSQL and ecosystem enhancements

Activity: Case study on HBase–Spark integration with example dataset

Evaluation Method: Case study report + written test

SEMESTER-V

COURSE 12 B: BIG DATA TECHNOLOGIES

Practical **Credits: 1** **2 hrs/week**

1. Installation & setup of Hadoop single-node cluster
2. Explore Hadoop directory structure and basic commands (hadoop fs operations)
3. Demonstration of Hadoop architecture components (HDFS, YARN, MapReduce) using sample logs
4. Store and retrieve large files from HDFS (block distribution, replication factor demo)
5. Simulate NameNode/DataNode failure and observe fault tolerance & recovery
6. Configure YARN and run sample applications, observe ResourceManager and NodeManager roles
7. Write a simple MapReduce program for word count
8. Develop a MapReduce job for inverted index creation
9. Perform data analysis using Pig Latin scripts
10. Execute Hive queries for structured data analysis (tables, partitions)
11. Import data from RDBMS into Hadoop using Sqoop
12. Capture and store log/streaming data using Flume
13. Serialize and store datasets in Avro and Parquet formats
14. Build an end-to-end ingestion workflow combining batch (Sqoop) and streaming (Flume)
15. Create and manage tables in HBase (CRUD operations)
16. Demonstrate coordination with ZooKeeper
17. Process HBase datasets using Spark integration with Hadoop

SEMESTER-V

COURSE 12 C: ROBOTICS KINEMATICS & DYNAMICS

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. Introduce fundamental concepts of robot kinematics and dynamics.
2. Develop ability to analyze robot motion through forward and inverse kinematics.
3. Understand velocity and acceleration relations via Jacobian matrices.
4. Derive robot dynamics equations using Newton-Euler and Lagrangian methods.
5. Learn trajectory planning and control techniques for robotic manipulators.

Course Outcomes

By the end of the course, students will be able to:

1. Identify robot components, degrees of freedom, and use Denavit-Hartenberg convention for kinematic modeling.
2. Solve forward and inverse kinematics problems for serial manipulators.
3. Compute Jacobians and analyze velocity kinematics and singularities.
4. Apply Newton-Euler and Lagrangian formulations to derive dynamic equations of robot manipulators.
5. Develop trajectory plans and implement basic control strategies for robot motion.

Unit 1: Introduction to Robot Kinematics

Introduction to robotics: types and applications, Robot components and coordinate frames, Degrees of freedom and workspace, Homogeneous transformations and spatial transformations, Denavit-Hartenberg (D-H) convention for link representation

Unit 2: Forward and Inverse Kinematics

Forward kinematics: computing end-effector position and orientation

Inverse kinematics: analytical and numerical solutions

Workspace analysis and tool configurations, Redundancy and multiple solutions in inverse kinematics

Unit 3: Velocity and Jacobian Analysis

Linear and angular velocities of rigid bodies, Differential motions and velocity propagation through robot links, Jacobian matrix: formulation and interpretation, Singularity analysis and dexterity measures, Kinematic duality and manipulator velocity control

Unit 4: Robot Dynamics

Basics of kinetics: work-energy, momentum principles, Newton-Euler formulation of robot dynamics, Lagrangian mechanics and derivation of equations of motion, Dynamic modeling of serial manipulators, Inclusion of actuator dynamics and nonrigid effects

Unit 5: Trajectory Planning and Control

Trajectory planning in joint and Cartesian spaces, Path interpolation and smooth motion generation, Control strategies: PD control, computed torque, inverse dynamics control, Force and hybrid control, Introduction to robot programming languages and simulation

Suggested Textbooks & References

1. John J. Craig, Introduction to Robotics: Mechanics and Control, Pearson
2. Mark W. Spong, Seth Hutchinson, M. Vidyasagar, Robot Modeling and Control, Wiley
3. Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, Giuseppe Oriolo, Robotics: Modelling, Planning and Control, Springer
4. Robert J. Schilling, Fundamentals of Robotics: Analysis and Control, Prentice Hall
5. Haruhisa Kawasaki, Robot Kinematics and Dynamics, Encyclopedia of Life Support Systems
6. MATLAB Robotics Toolbox Documentation and Tutorials for practical work

Activities:

Unit 1: Introduction to Robot Kinematics

Activities:

- Study robot types and components
- Setup coordinate frames and assign Degrees of Freedom
- Practice homogeneous and spatial transformations
- Apply Denavit-Hartenberg parameters modeling on simple mechanisms

Outcomes: Ability to model robots in terms of coordinate frames and D-H parameters.

Evaluation: Quiz and assignment on D-H parameter assignments and transformations.

Unit 2: Forward and Inverse Kinematics

Activities:

- Compute end-effector position and orientation forward kinematics
- Solve inverse kinematics analytically and numerically for basic robots
- Analyze robot workspace and tool configurations

Outcomes: Skill in solving forward/inverse kinematic problems and workspace analysis.

Evaluation: Problem-solving assignment and class test.

Unit 3: Velocity and Jacobian Analysis

Activities:

- Calculate linear and angular velocities of robot links
- Derive Jacobian matrices and interpret their significance
- Study singularity and robot dexterity measures

Outcomes: Understanding of velocity propagation and kinematic influence on manipulator control.

Evaluation: Lab exercises on Jacobian calculation and singularity analysis.

Unit 4: Robot Dynamics

Activities:

- Learn basic kinetics principles relevant to robotics
- Derive equations of motion with Newton-Euler and Lagrangian methods
- Model dynamics of serial manipulators including actuator effects

Outcomes: Ability to formulate dynamic models for robot motion analysis.

Evaluation: Assignment on dynamics derivation and mid-term written exam.

Unit 5: Trajectory Planning and Control

Activities:

- Generate and simulate robot trajectories in joint and Cartesian spaces
- Explore control schemes: PD, computed torque, inverse dynamics control
- Understand force and hybrid control approaches; introduction to programming and simulation tools

Outcomes: Skills in planning trajectories and designing control strategies for robot manipulators.

Evaluation: Project work and final exam practicals.

SEMESTER-V

COURSE 12 C: ROBOTICS KINEMATICS & DYNAMICS

Practical **Credits: 1** **2 hrs/week**

Practical 1: MATLAB Basics for Robotics

- Learn basic MATLAB operations and plotting
- Visualize points, vectors, and coordinate frames in 2D and 3D

Practical 2: Coordinate Transformations

- Create rotation and translation matrices
- Build homogeneous transformation matrices
- Apply transformations to points and visualize the results

Practical 3: Denavit-Hartenberg Parameters and Forward Kinematics

- Assign DH parameters for a simple 2-DOF planar arm
- Compute forward kinematics using DH transformation
- Plot end-effector positions for given joint angles

Practical 4: Inverse Kinematics of Planar Manipulator

- Solve analytical inverse kinematics for 2-DOF arm
- Verify solutions by forward kinematics plotting

Practical 5: Jacobian and Velocity Kinematics

- Calculate Jacobian matrix for 2-DOF arm
- Compute end-effector velocity from joint velocities
- Analyze singular configurations

Practical 6: Basic Trajectory Planning

- Generate joint trajectories with linear interpolation
- Plot joint angle, velocity, acceleration profiles
- Simulate end-effector path

Practical 7: Simple Dynamic Calculations

- Calculate torque requirements for joint movements
- Plot torque vs. time for simple motion profiles

Practical 8: Robot Motion Animation

- Animate arm movement following generated trajectories
- Visualize workspace and reachable points

SEMESTER-V

COURSE 13 A: NATURAL LANGUAGE PROCESSING

Theory **Credits: 3** **3 hrs/week**

Course Objectives:

1. Introduce the foundations of Natural Language Processing and its applications in real-world tasks.
2. Familiarize students with text preprocessing, linguistic analysis, and parsing techniques.
3. Equip learners with methods for information extraction, word representations, and sentiment classification.
4. Explore deep learning techniques for NLP, including RNNs, LSTMs, GRUs, and Transformers.
5. Provide hands-on experience with modern NLP tools (NLTK, spaCy, Hugging Face) for implementing applications such as chatbots, summarization, and document classification.

Course Outcomes:

At the end of this course, students will be able to:

1. Explain the principles, challenges, and applications of NLP and use basic text processing tools.
2. Apply preprocessing techniques (tokenization, stemming, lemmatization) and parsing methods to analyze language structures.
3. Implement information extraction and text representation methods (NER, embeddings, classification pipelines).
4. Build and evaluate deep learning models (RNN, LSTM, GRU, Transformer) for NLP tasks.
5. Utilize pre-trained transformer models (BERT, GPT) with Hugging Face for advanced NLP applications such as summarization, chatbots, and document classification.

Unit 1. Introduction to NLP and Language Fundamentals:

Definition, Goals, and Scope of NLP, Real-world Applications (Assistants, Chatbots, Translation, Summarization, QA, Spam Detection), Fundamentals of Language Processing, Ambiguities in NLP (Lexical, Structural, Contextual)

Installations: Python setup, NLTK, spaCy basics

Regular Expressions (Essential patterns, findall, split, sub, matching tokens)

Unit 2. Text Preprocessing and Linguistic Analysis:

Key NLP Terminologies: Morphology, Lexicon, Orthographic Rules

Finite State Transducers

Text Preprocessing Techniques: Tokenization, Stopword Removal, Stemming, Lemmatization

Grammar and Context-Free Grammar

Parsing Techniques: Top-down, Bottom-up, CYK Algorithm

Semantic Analysis: Elements, Meaning Representation

Unit 3. Information Extraction and Representation:

Named Entity Recognition (NER): Concepts, Examples, Using spaCy & NLTK

Word Embeddings: Word2Vec (Skip-Gram, CBOW), Comparison, Implementations, Bag of Words and N-grams, Text Classification Pipeline, Sentiment Analysis Applications

Ethical considerations in preprocessing & classification

Unit 4. Deep Learning for NLP:

Recurrent Neural Networks (RNN): Basics, RNN vs CNN/Feedforward NN, LSTM and GRU for Sequence Modeling, Transformer Models: Introduction, Pretrained Models (BERT, GPT), Hugging Face Ecosystem

Unit 5. Transformers and Modern NLP:

Transformer architecture basics (self-attention, encoder-decoder), BERT: Pretraining, Fine-tuning,

GPT and Generative NLP, Hugging Face Ecosystem (using pre-trained models)

Text Summarization: Extractive, Abstractive, Hybrid Approaches

Applications: Document Classification, Chatbots, Virtual Assistants

Textbook:

1. Natural Language Processing, Sini Raj Pulari, Umadevi Maramreddy, Shriram k. Vasudevan, Oxford University Press
2. Speech and Language Processing, An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Daniel Jurafsky, James H. Martin, Pearson Education, 2023.

Reference Book:

1. Natural Language Processing and Information Retrieval, Tanveer Siddiqui, U.S. Tiwary , Oxford University Press.
2. 2. Natural Language Processing Recipes - Unlocking Text Data with Machine Learning and Deep Learning using Python, Akshay Kulkarni, Adarsha Shivananda, Apress, 2019.

Activities:

Outcome: Explain NLP fundamentals and basic text processing tools.

Activity: Quiz/ Assignment on Analyze ambiguities in Indian language sentences.

Evaluation Method: Quiz Score

Outcome: Apply preprocessing and parsing techniques to analyze language.

Activity: Case study: Building a simple grammar-based sentence parser.

Evaluation Method: Application of text preprocessing to raw text corpus, parsing

Outcome: Implement information extraction and text representation methods.

Activity: Hands-on NER using spaCy and NLTK.

Evaluation Method: Lab report submission on embeddings & NER.

Outcome: Build and evaluate deep learning models for NLP.

Activity: Group Discussion on Discussion: Compare RNN vs Transformers.

Evaluation Method: Depth of Understanding, Participation, Explanation

Outcome: Utilize pre-trained transformer models for advanced NLP applications.

Activity: Lab: Implement chatbot using GPT model.

Evaluation Method: Practical exam using Hugging Face models – Accuracy, Effectiveness

SEMESTER-V

COURSE 13 A: NATURAL LANGUAGE PROCESSING

Practical **Credits: 1** **2 hrs/week**

1. Install Python, NLTK, and spaCy. Write a sample program to print available NLP corpora and models.
2. Write regex patterns for extracting emails, phone numbers, hashtags, and dates from a text file.
3. Demonstrate lexical and structural ambiguity with example sentences. Use NLTK parse trees to visualize.
4. Implement sentence and word tokenization using NLTK and spaCy. Compare outputs.
5. Write a program to remove stopwords and analyze text length reduction.
6. Apply Stemming and Lemmatization on a dataset and compare differences.
7. Use NLTK to demonstrate top-down and bottom-up parsing of a simple grammar.
8. Use spaCy to extract entities (e.g., names, locations, organizations) from news text.
9. Implement text representation and calculate similarity between documents.(Bag of Words and N-grams)
10. Build a sentiment classifier using Scikit-learn (Naive Bayes / Logistic Regression).
11. Implement a simple RNN to generate sentences character by character.
12. Hugging Face to load a pretrained BERT model and perform masked word prediction.
13. Implement extractive and abstractive summarization using Hugging Face pipelines.
14. Build a simple FAQ-based chatbot using Transformer-based embeddings.

SEMESTER-V

COURSE 13 B: CLOUD COMPUTING FOR DATA SCIENCE

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. Introduce the fundamentals of cloud computing and its role in data science.
2. Provide understanding of virtualization, service, and deployment models.
3. Familiarize students with cloud storage, data management, and databases.
4. Expose students to cloud-based big data and machine learning platforms.
5. Train students in building, deploying, and monitoring ML pipelines on the cloud.

Course Outcomes

At the end of this course, students will be able to:

1. Explain cloud computing concepts including service models, deployment models, and virtualization.
2. Demonstrate cloud storage and database services for managing large-scale data.
3. Apply cloud-based platforms to run data science and machine learning workflows.
4. Build and deploy ML models on cloud services using AutoML and managed ML platforms.
5. Evaluate and monitor cloud-deployed solutions with respect to scalability, performance, and cost.

Unit 1. Introduction to Cloud Computing

Definition & Evolution of Cloud Computing, Service-Oriented Architecture (SOA) & Web Services, Utility & Grid Computing concepts, Characteristics of Cloud Computing

Cloud Computing Architecture: Front-end, Back-end, Networking, Delivery Models

Cloud Service Models: SaaS, PaaS, IaaS, Continuous Delivery using PaaS

Unit 2. Virtualization & Deployment Models

Concept & importance of Virtualization, Types of Virtualizations: Application, Network, Desktop, Storage, Server, Data Virtualization

Cloud Deployment Models: Public, Private, Community, Hybrid

Role of Cloud Computing in Data Science, Advantages of Cloud in Machine Learning

Unit 3. Cloud Storage & Data Management

Cloud Storage: Introduction, Benefits, Use Cases (Backup, Archiving, DR, Content Delivery)

Cloud Storage Systems: Block-based, File-based, Object-based storages

Key-Value Databases: Features & limitations

Batch vs. Streaming data for ML pipelines

Cloud Data Warehouses: AWS Redshift, Google BigQuery

Unit 4. Cloud Platforms for Data Science & ML

Machine Learning in the Cloud: Benefits & Limitations, Cloud-based ML Services: AIaaS, GPUaaS

Managed ML Platforms: Overview & advantages, Cloud ML Platforms: AWS SageMaker, Azure ML Studio, Google Cloud AutoML

Unit 5. Training & Deployment of ML on Cloud

Factors for selecting Cloud ML Platforms: ETL/ELT pipeline support, Scale-up/out training, ML frameworks, Pre-tuned services

Steps for Training ML Models in Cloud: Data source identification, Feature engineering, Training, Validation, Deployment, Monitoring, Monitoring & improving cloud-deployed ML models

Case studies & industry applications

Text / Reference Books

1. Handbook of Cloud Computing, Dr. Anand Nayyar, BPB Publications (2019)
2. Cloud Computing: A Practical Approach, Toby Velte, Anthony Velte, Robert C., McGraw Hill
3. Cloud Computing for Data Analysis, Noah Gift, Alfredo Deza, Pragmatic AI Labs
4. Data Science on the Google Cloud Platform, Valliappa Lakshmanan, O'Reilly
5. Machine Learning in the AWS Cloud: Amazon SageMaker, Abhishek Mishra, Wiley

Activities:

Outcome: Explain cloud computing concepts including service models, deployment models, and virtualization.

Activity: Prepare a comparative chart/report on cloud service and deployment models with real-world examples (AWS, Azure, GCP).

Evaluation Method: Report submission & viva (assess clarity, accuracy, and examples used).

Outcome: Demonstrate cloud storage and database services for managing large-scale data.
Activity: Perform lab experiments on block/file/object storage and execute queries in BigQuery / RDS.
Evaluation Method: Lab performance & practical exam (students demonstrate CRUD operations and explain storage use cases).

Outcome: Apply cloud-based platforms to run data science and machine learning workflows.
Activity: Implement a cloud-based ETL pipeline (e.g., using AWS Glue / Dataflow) for preparing a dataset.
Evaluation Method: Lab report + demo evaluation (workflow completeness, correctness of execution).

Outcome: Build and deploy ML models on cloud services using AutoML and managed ML platforms.
Activity: Train and deploy a classification/regression model using AWS SageMaker / Azure ML / Google AutoML.
Evaluation Method: Practical demo + oral viva (assess deployment success, prediction results, and understanding of pipeline).

Outcome: Evaluate and monitor cloud-deployed solutions with respect to scalability, performance, and cost.
Activity: Configure monitoring (e.g., CloudWatch, Stackdriver) for a deployed ML service and analyze resource usage.
Evaluation Method: Mini-project report & presentation (grading based on monitoring setup, analysis quality, cost optimization suggestions).

SEMESTER-V

COURSE 13 B: CLOUD COMPUTING FOR DATA SCIENCE

Practical **Credits: 1** **2 hrs/week**

1. Create Virtual Machine using VMware workstation for Windows/Linux
2. Install & configure WAMP/XAMPP/Apache on the VM and host a sample page
3. Install and configure a cloud account (AWS/Azure/GCP free tier).
4. Create and manage storage buckets; upload and access datasets.
5. Launch an instance and configure Block-based storage (EBS)
6. Create & configure File-based storage on cloud VM (EFS/Network FS).
7. Set up Jupyter Notebook/Colab on cloud VM.
8. Connect to cloud-hosted database services (AWS RDS, BigQuery, Cosmos DB).
9. Implement a batch ETL pipeline in the cloud.
10. Launch a SageMaker notebook, attach IAM role and S3 bucket, run sample notebook
11. Build a classification/regression model using AWS SageMaker / Azure ML Studio / GCP AI Platform.
12. Implement a simple ETL job: extract (RDS / CSV), transform, load into cloud DW (e.g., Redshift / BigQuery).
13. Use CloudWatch / Stackdriver to monitor endpoints, set alarms and auto-scale rules.
14. Use cloud AutoML services for dataset prediction tasks.
15. Deploy a trained ML model as a REST API endpoint in the cloud.

SEMESTER-V

COURSE 13 C: ADDITIVE MANUFACTURING & IOT

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. To provide a comprehensive understanding of additive manufacturing technologies, workflows, and design considerations.
2. To explore the fundamentals and advanced concepts of Industrial IoT enabling technologies and architectures in manufacturing.
3. To develop skills for integrating IoT systems in smart manufacturing and additive manufacturing environments.
4. To familiarize students with IoT devices such as ESP32 and Raspberry Pi for practical implementation in manufacturing contexts.
5. To introduce IoT security, data analytics, and emerging trends relevant to Industry 4.0 and additive manufacturing.

Course Outcomes

By the end of this course, students will be able to:

1. Explain the fundamental principles, workflows, and applications of additive manufacturing.
2. Analyze and differentiate between various additive manufacturing processes and their design parameters.
3. Describe IoT enabling technologies, architectures, communication protocols, and deployment models used in manufacturing.
4. Implement IoT applications using ESP32 and Raspberry Pi for real-time monitoring and data logging in manufacturing processes.
5. Identify and apply IoT security principles and emerging Industry 4.0 technologies in smart manufacturing.

Unit 1: Fundamentals and Workflow of Additive Manufacturing

Introduction to Additive Manufacturing (AM) and evolution, Manufacturing workflow in AM: design, material selection, printing, post-processing, and quality control, Classification of AM processes and materials used, Benefits and limitations of AM, Industrial applications and case studies of AM

Unit 2: Advanced Additive Manufacturing Processes and Design

Detailed study of extrusion-based, powder bed fusion, vat photopolymerization, and other AM technologies, Design for Additive Manufacturing (DfAM) principles and topology optimization, Process parameters, post-processing techniques, and quality assurance, Integration of AM into smart manufacturing systems

Unit 3: IoT Enabling Technologies and Architecture in Manufacturing

Overview of Industrial IoT (IIoT) and its role in manufacturing, IoT enabling technologies: sensors, actuators, RFID, wireless communication (Wi-Fi, ZigBee, 5G), IoT architecture layers: perception, network, processing, and application layers, IoT device interfaces and networking protocols (MQTT, CoAP, OPC-UA), IoT levels in industrial systems and deployment templates

Unit 4: IoT Integration and Applications in Smart Manufacturing

Integration of IoT with manufacturing workflows and Additive Manufacturing machines, IoT-enabled real-time monitoring, predictive maintenance, asset tracking, and quality control, Cloud computing, edge computing, and data management in IoT manufacturing setups, Digital twins, cyber-physical systems, and IoT platforms for smart manufacturing

Unit 5: IoT Security, Analytics, and Future Trends

Security challenges and best practices in Industrial IoT, Big data analytics, AI, and machine learning applications in IoT-enabled manufacturing, Cybersecurity frameworks and risk mitigation strategies for IIoT, Emerging trends: augmented reality, 5G, blockchain, and sustainable IoT solutions in manufacturing

Suggested Textbooks & References

1. Ian Gibson, David Rosen, Brent Stucker, Additive Manufacturing Technologies, Springer
2. Rajkumar Buyya, Amir Vahid Dastjerdi, Internet of Things: Principles and Paradigms, Morgan Kaufmann
3. Sabina Jeschke, Christian Brecher, Industrial Internet of Things: Cybermanufacturing Systems, Springer

4. Various, Industrial Internet of Things: Technologies, Design, and Applications, Routledge
5. Giacomo Veneri, Hands-On Industrial Internet of Things: Build Real-World IIoT Projects, Packt Publishing

Activities

Unit 1: Fundamentals and Workflow of Additive Manufacturing

Activities: Study AM processes and workflow steps including design, selection, printing, and post-processing. Explore case studies on industrial applications of AM

Outcomes: Understand different AM methods and their industrial relevance

Evaluation: Quiz and assignment on AM workflows and materials

Unit 2: Advanced Additive Manufacturing Processes and Design

Activities: Analyze details of different AM technologies and process parameters. Practice CAD modeling and topology optimization for AM

Outcomes: Ability to choose appropriate AM processes and design for manufacturing.

Evaluation: Project on CAD modeling and process selection

Unit 3: IoT Enabling Technologies and Architecture

Activities: Study sensors, RFID, and wireless protocols used in industrial IoT. Explore IoT architecture layers and deployment templates

Outcomes: Explain IoT enabling technologies and system architectures.

Evaluation: Assignment on IoT protocols and architectures

Unit 4: IoT Integration and Applications in Smart Manufacturing

Activities: Analyze integration of IoT with AM and manufacturing workflows. Setup practical IoT monitoring scenarios using ESP32/Raspberry Pi.

Outcomes: Implement IoT applications for manufacturing monitoring and control

Evaluation: Lab project on IoT-enabled real-time monitoring

Unit 5: IoT Security, Analytics, and Future Trends

Activities: Study security challenges and mitigation strategies in IIoT. Explore AI, big data analytics, and emerging Industry 4.0 trends

Outcomes: Evaluate security and technological trends in IoT manufacturing.

Evaluation: Presentation on IoT security and future developments

SEMESTER-V

COURSE 13 C: ADDITIVE MANUFACTURING & IOT

Practical **Credits: 1** **2 hrs/week**

Additive Manufacturing Lab

1. Introduction and Familiarization with AM Machines
   - Hands-on study of FDM, SLA, SLS type 3D printing machines
   - Safety and operational procedures
2. CAD Modeling and STL File Preparation
   - Creating 3D CAD models using CAD software
   - Generating and repairing STL files for 3D printing
3. Process Simulation and Optimization
   - Use of slicing software (e.g., CURA, Catalyst)
   - Simulation of build time, support structures, and material consumption
4. Fabrication of Prototype Parts
   - Printing parts with different AM processes
   - Post-processing and dimensional accuracy checking
5. Reverse Engineering and Physical Model Creation
   - Creating 3D models from scans or measurements
   - Printing and validating the models

IoT Laboratory (with ESP32 / Raspberry Pi)

1. Introduction to ESP32 and Raspberry Pi
   - Overview of board features, pin configuration, and development environment setup
   - Basic programming and flashing firmware
2. Sensor Interfacing and Data Acquisition
   - Interfacing digital and analog sensors (temperature, humidity, light, motion) with ESP32 and Raspberry Pi
   - Reading sensor data using GPIO, ADC, and I2C interfaces
3. Wireless Communication Setup
   - Connecting ESP32/Raspberry Pi to Wi-Fi networks
   - Simple data transmission over HTTP and MQTT protocols
4. Data Logging and Cloud Integration
   - Sending sensor data to cloud IoT platforms like ThingSpeak, Adafruit IO, or AWS IoT

- Creating dashboards for real-time visualization

5. IoT Security Basics and Deployment
    - Implementing basic security practices: Wi-Fi encryption, data encryption, and secure MQTT connections
    - Configuring deployment templates for common industrial IoT use cases
6. IoT-based Monitoring for Additive Manufacturing (Capstone)
    - Using ESP32/Raspberry Pi to simulate real-time monitoring of AM process parameters
    - Data logging, alerts, and remote monitoring setup

SEMESTER-VI

COURSE 14 A: CONVERSATIONAL AI

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. Understand the fundamentals and significance of conversational AI and chatbots.
2. Learn to design conversation flows and manage dialogs effectively.
3. Gain hands-on experience in building and training Natural Language Understanding (NLU) models.
4. Develop skills for creating multilingual and voice-enabled chatbots.
5. Testing, deployment, and maintenance of conversational AI systems.

Course Outcomes

Upon completion, students will be able to:

1. Describe core components and architecture of conversational AI systems.
2. Design and implement dialog flows and effective conversation management strategies.
3. Build and evaluate NLU models for intent classification and entity extraction.
4. Create multilingual and voice-enabled chatbots integrating speech technologies.
5. Conduct chatbot testing, deploy on multiple platforms, and implement monitoring for continuous improvement.

Unit 1: Introduction to Conversational AI and Chatbots

What is Conversational AI and its importance, Overview of chatbot use cases and domains, Key components: NLU, dialog management, NLG, Dialog systems and architecture, Introduction to chatbot platforms and tools, Differences between rule-based and AI chatbots, Challenges in conversational AI development

Unit 2: Designing Conversations and Dialog Management

Understanding conversation flows and user intents, Dialog management strategies: state machines, frame-based systems, Slot filling techniques for context maintenance, Handling multi-turn conversations and context switching, Managing conversation errors and fallback strategies, Role of session and user context in dialog personalization, Best practices in conversation design for UX

Unit 3: Natural Language Understanding (NLU) Fundamentals

Preparing training annotation and labeling, intent classification techniques and algorithms , Entity recognition and extraction methods, Using pre-trained models and transfer learning in NLU, Handling ambiguous and overlapping intents, Supporting multiple languages with NLU components, Evaluating NLU model performance

Unit 4: Building Multilingual and Voice-Enabled Chatbots

Techniques to support multilingual conversations, Speech recognition and synthesis basics , Integrating voice assistants with chatbots, text-to-speech (TTS) and speech-to-text (STT) , Designing voice user interfaces (VUIs), Testing voice chatbots with different accents and dialects, Privacy and ethical considerations in voice-enabled AI

Unit 5: Testing, Deployment, and Maintenance of Chatbots

Testing chatbots: functional, user acceptance, load testing, Metrics for chatbot performance and user satisfaction, Continuous learning, retraining, and updating models, Handling failures and graceful degradation, Deploying chatbots across platforms and integration with backend systems, Analytics and monitoring tools for conversational AI, Future trends in conversational AI and emerging technologies

Recommended Textbooks & References

- "Reinforcement Learning: An Introduction" by Richard S. Sutton and Andrew G. Barto, MIT Press
- "Conversational AI: Chatbots that work" by Andrew Freed, Packt Publishing
- "Handbook of Natural Language Processing" edited by Nitin Indurkhya and Fred J. Damerau
- "Conversational Interfaces: Principles and Practice" by Michael McTear
- "Deep Learning for Natural Language Processing" by Palash Goyal, Sumit Pandey, Karan Jain
- "Artificial Intelligence: A Modern Approach" by Stuart Russell and Peter Norvig
- Rasa Open Source Documentation (https://rasa.com/docs/rasa)
- "Building Chatbots with Python" by Sumit Raj, Apress
- Online tutorials and practical guides from Rasa community and GitHub repositories

Activities

Unit 1: Introduction to Conversational AI and Chatbots

Activity: Explore chatbot examples and dissect major components

Outcome: Understand conversational AI scope, architecture, and difference between rule-based and AI chatbots

Evaluation Method: Quiz on chatbot concepts and architecture; assignment on case studies of chatbot use cases

Unit 2: Designing Conversations and Dialog Management

Activity: Map conversation flows, build intents and slots, design fallback strategies using Rasa

Outcome: Ability to model user intents, manage context, and design resilient dialogs

Evaluation Method: Lab exercises designing conversation flows; peer review of chatbot dialogues

Unit 3: Natural Language Understanding (NLU) Fundamentals

Activity: Annotate training data, train NLU pipeline in Rasa, experiment with entity extraction and intent classification

Outcome: Develop skills in preparing data and training NLU models; evaluate model performance

Evaluation Method: Practical NLU model accuracy tests; report on training data preparation and challenges

Unit 4: Building Multilingual and Voice-Enabled Chatbots

Activity: Implement multilingual support using separate pipelines; integrate speech-to-text and text-to-speech APIs

Outcome: Create chatbots supporting multiple languages and voice interfaces

Evaluation Method: Demonstration of multilingual chatbot proficiency; voice bot interaction testing

Unit 5: Testing, Deployment, and Maintenance of Chatbots

Activity: Use Rasa X for interactive testing, deploy chatbot on cloud servers, monitor conversation analytics

Outcome: Master chatbot evaluation, deployment techniques, error handling, and continuous improvement

Evaluation Method: Final project deployment; analysis report of chatbot performance and user feedback

SEMESTER-VI

COURSE 14 A: CONVERSATIONAL AI

Practical **Credits: 1** **2 hrs/week**

Tools: Rasa Open Source

Experiment 1: Setting Up Rasa

- Install Rasa and prerequisite tools
- Understand folder structure and key files (domain.yml, nlu.yml, stories.yml)
- Create a basic rule-based chatbot and test on Rasa shell

Experiment 2: Defining Intents and Entities

- Create intents and entities in `nlu.yml`
- Annotate training data for intent classification and entity extraction
- Train and test the NLU model with sample queries

Experiment 3: Dialogue Management with Stories and Rules

- Write stories for conversation flows (user intents + bot responses)
- Define rules for slot filling and fallback handling
- Test multi-turn conversations and context handling

Experiment 4: Implementing Slots and Form Actions

- Use slots to capture and remember user inputs
- Create forms to fill multiple slots during conversations
- Manage slot validity and resetting

Experiment 5: Custom Actions for Dynamic Responses

- Write custom Python actions (in `actions.py`) for dynamic content
- Use APIs or simple logic to enhance chatbot responses
- Test bot with custom actions integrated

Experiment 6: Multilingual Chatbot Development

- Extend chatbot to support multiple languages via separate NLU pipelines
- Handle intents and responses per language
- Test language switching in conversations

Experiment 7: Integrating Voice with Rasa

- Connect Rasa chatbot with speech-to-text and text-to-speech services
- Build a simple voice-enabled chatbot demo
- Evaluate voice input accuracy and bot response

Experiment 8: Error Handling, Fallbacks, and Recovery

- Implement fallback policies for unclear user inputs
- Log fallback incidents for model improvement
- Design conversation recovery strategies

Experiment 9: Testing, Evaluation, and Debugging

- Use Rasa X for interactive learning and conversation reviews
- Evaluate intent classification, entity extraction, and dialogue success
- Debug and retrain models iteratively

Experiment 10: Deploying Rasa Chatbot

- Deploy Rasa chatbot locally and on cloud platforms (Heroku, Railway)
- Expose bot via chatbot UI or API endpoints
- Test chatbot in real-world web interface

SEMESTER-VI

COURSE 14 B: TIME SERIES ANALYSIS AND FORECASTING

Theory **Credits: 3** **3 hrs/week**

Course Objectives

The course aims to:

1. Provide fundamental understanding of time series data, components, and characteristics.
2. Train students in identifying, modeling, and forecasting using ARMA/ARIMA/SARIMA models.
3. Introduce state-space and multivariate approaches for complex data.
4. Familiarize students with modern forecasting methods, including spectral and evaluation techniques.
5. Enable hands-on practice with real-world datasets using R/Python statistical libraries.

Course Outcomes

By the end of the course, students will be able to:

1. Explain the concepts of time series, stationarity, and autocorrelation functions.
2. Apply ARMA/ARIMA/SARIMA models to real-world time series data.
3. Analyze multivariate and state-space time series using appropriate tools.
4. Implement forecasting workflows using R/Python for financial, business, and scientific datasets.
5. Evaluate forecast accuracy and select appropriate models using statistical criteria.

Unit 1. Fundamentals & Stationary Processes

Introduction to time series: types, components, forecasting process. Stationary processes: definitions, autocovariance, autocorrelation functions (ACF/PACF).

Model evaluation metrics. ACF/PACF example analyses.

Unit 2. ARMA & Forecasting with ARMA

ARMA(p,q) models: definition, estimation, forecasting approaches. Model identification: AIC, PACF/ACF, diagnostic checks. Practical examples of fitting ARMA and generating forecasts.

Unit 3. Non-Stationary & Seasonal Models

Non-stationary time series: differencing, unit roots. Seasonal models: SARIMA and multiplicative seasonal ARIMA. Identification, estimation, and diagnostic checks for seasonal models.

Unit 4. State-Space & Multivariate Time Series

Multivariate time series: Vector ARMA models (VARMA), estimation, forecasting. State-space representation: formulation, Kalman filter basics, forecasting in state-space models.

Unit 5. Advanced Topics & Forecast Evaluation

Spectral analysis: frequency-domain representation, spectral density. Forecast performance: measures, monitoring, choosing models.

Textbook:

1. Introduction to Time Series and Forecasting, Peter J. Brockwell & Richard A. Davis, 2nd Edition, Springer

Reference Books

1. Time Series Analysis: Forecasting and Control, George E. P. Box, Gwilym M. Jenkins & Gregory C. Reinsel
2. Introduction to Time Series Analysis and Forecasting, Douglas C. Montgomery, Cheryl L. Jennings, Murat Kulahci , (Wiley)
3. Time Series Analysis and Its Applications: With R Examples, R. H. Shumway & D. S. Stoffer

Activities:

Outcome: Explain the concepts of time series, stationarity, and autocorrelation functions

Activity: Students will prepare a seminar or short presentation explaining stationarity, ACF, PACF with a simple dataset example (like sales data).

Evaluation Method: Evaluated through presentation quality, understanding during viva, and a short concept quiz.

Outcome: Apply ARMA/ARIMA/SARIMA models to real-world time series data

Activity: Students will conduct a hands-on lab task to fit ARIMA and SARIMA models on stock price or rainfall data using Python/R.

Evaluation Method: Lab record submission, correctness of implementation, and a practical exam.

Outcome: Analyze multivariate and state-space time series using appropriate tools
Activity: Students will carry out a case study on macroeconomic datasets (like GDP, inflation, unemployment) using VAR or state-space modeling.
Evaluation Method: Case study report, results interpretation, and oral viva.

Outcome: Implement forecasting workflows using R/Python for financial, business, and scientific datasets
Activity: Students will design an end-to-end forecasting pipeline (data preprocessing → model building → forecasting → visualization). Example: forecasting COVID-19 daily cases or retail sales.
Evaluation Method: Project demo, code submission, and project report.

Outcome: Evaluate forecast accuracy and select appropriate models using statistical criteria
Activity: Students will compare multiple forecasting methods (e.g., ARIMA vs. Exponential Smoothing) on the same dataset and analyze performance using RMSE, MAE, and MAPE.
Evaluation Method: Written assignment, interpretation of metrics, and justification of chosen model.

SEMESTER-VI

COURSE 14 B: TIME SERIES ANALYSIS AND FORECASTING

Practical **Credits: 1** **2 hrs/week**

(Using R/Python statsmodels, pandas, forecast, or equivalent)

1. Import and visualize time series datasets (stock, weather, sales).
2. Perform decomposition of time series into trend, seasonal, residual components.
3. Compute and plot Autocorrelation Function (ACF) & Partial ACF (PACF).
4. Test stationarity using Augmented Dickey-Fuller (ADF) test.
5. Fit ARMA models and validate residuals.
6. Implement ARIMA and SARIMA models for seasonal data.
7. Perform model selection using AIC/BIC and cross-validation.
8. Forecast with ARIMA/SARIMA and plot prediction intervals.
9. Apply multivariate time series (VAR) to macroeconomic datasets.
10. Explore state-space models using Kalman filtering.
11. Conduct spectral analysis of a time series.
12. Compare forecasting methods: ARIMA vs. Exponential Smoothing vs. ML models.
13. Evaluate forecast performance with RMSE, MAPE, etc.

SEMESTER-VI

COURSE 14 C: ROBOT OPERATING SYSTEMS

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. To introduce students to the architecture and core concepts of Robot Operating System (ROS).
2. To develop skills in ROS installation, basic node programming, and use of ROS tools.
3. To enable robot modeling and simulation using URDF, rviz, and Gazebo with ROS integration.
4. To teach motion planning, navigation, and basic SLAM using ROS MoveIt! and navigation stack.
5. To expose students to advanced ROS applications including sensor data processing, action servers, and ROS2 features.

Course Outcomes

Upon completion of this course, students will be able to:

1. Understand and explain ROS architecture, middleware, and communication mechanisms.
2. Install ROS and develop basic publisher and subscriber nodes using Python and C++.
3. Model, visualize, and simulate robotic systems using URDF, rviz, and Gazebo tools.
4. Implement motion planning and robot navigation using ROS MoveIt!, SLAM, and navigation stacks.
5. Develop advanced ROS applications including asynchronous communication, hardware interfacing, and sensor integration.

Unit 1: Introduction to ROS and Middleware Concepts

Overview of ROS architecture and design principles, ROS communication mechanisms: Nodes, Topics, Services, and Actions, ROS Master, Parameter Server, and Package management, Understanding ROS graph and computation model.

Unit 2: ROS Installation, Tools, and Basic Programming

Introduction to ROS command line tools and GUI tools (rviz, rqt), Writing basic publisher and subscriber nodes in Python and C++, ROS messages and custom message creation

Unit 3: Robot Modeling and Simulation in ROS

Using URDF for robot description, Visualization with rviz, Simulation with Gazebo and integration with ROS, Sensor integration and simulation

Unit 4: Motion Planning and Navigation

Introduction to ROS MoveIt! Framework, Path planning algorithms and controllers, Robot localization and mapping (SLAM basics), Navigation stack overview and implementation

Unit 5: Advanced ROS Concepts and Applications

Action servers and clients, Integration with real robots and hardware interfaces, Using ROS with sensor data processing (e.g., LIDAR, cameras), Introduction to ROS2 and comparative features

Suggested Textbooks & References

1. Morgan Quigley, Brian Gerkey, William D. Smart, Programming Robots with ROS, O'Reilly
2. Lentin Joseph, Robot Operating System (ROS) for Absolute Beginners, Apress
3. Anis Koubaa, Robot Operating System (ROS): The Complete Reference, Springer
4. Aaron Martinez, Enrique Fernandez, Learning ROS for Robotics Programming, Packt Publishing
5. Patrick Goebel, Mastering ROS for Robotics Programming, Packt Publishing

Activities

Unit 1: Introduction to ROS and Middleware Concepts

Activities:

- Study ROS architecture, nodes, topics, services, and actions through lectures and tutorials
- Explore ROS Master, Parameter Server, and package management concepts using diagrams and real examples
- Run ROS core and basic nodes to understand communication flow

Outcomes: Explain ROS middleware components and ROS computation graph.

Evaluation: Quiz on ROS architecture and communication model.

Unit 2: ROS Installation, Tools, and Basic Programming

Activities:

- Install ROS distribution and setup workspace on Linux
- Write and execute simple publisher and subscriber nodes in Python and C++
- Practice using ROS tools such as rostopic, rosnode, rviz, and rqt for debugging and visualization
- Create and use custom ROS messages

Outcomes: Develop and debug ROS nodes using programming and built-in ROS tools.

Evaluation: Lab exercises and assignment on basic ROS programming

Unit 3: Robot Modeling and Simulation in ROS

Activities:

- Create robot models using Unified Robot Description Format (URDF)
- Visualize robot models and frames in rviz; manipulate robot state and TF (transform) trees
- Setup robot and sensor simulation environments using Gazebo with real-time integration
- Subscribe to simulated sensor data and process it within ROS nodes.

Outcomes: Model and simulate robotic systems along with sensor integration.

Evaluation: Practical lab on URDF modeling and Gazebo simulation

Unit 4: Motion Planning and Navigation

Activities:

- Configure and use MoveIt! framework for robot arm motion planning
- Implement basic path planning and execute robot motions in simulation
- Set up localization and mapping using ROS SLAM packages
- Execute navigation stack for robot movement in known and unknown environments

Outcomes: Implement motion planning and autonomous navigation using ROS tools.

Evaluation: Mini-project implementing motion planning and navigation tasks.

Unit 5: Advanced ROS Concepts and Applications

Activities:

- Develop and test action servers and clients for preemptible asynchronous tasks
- Interface with hardware or simulated sensors such as LIDAR and cameras

- Process real and simulated sensor data for perception and control
- Compare features of ROS and ROS2 through demonstrations and discussions.

Outcomes: Design advanced ROS applications integrating hardware, sensors, and asynchronous communication.

Evaluation: Final project demonstrating integrated ROS functionalities

SEMESTER-VI

COURSE 14 C: ROBOT OPERATING SYSTEMS

Practical **Credits: 1** **2 hrs/week**

1. ROS Setup and Basic Node Programming
   - Installing ROS and setting up the workspace
   - Writing simple publisher/subscriber examples in Python/C++
   - Running ROS core and nodes, using ROS tools
2. Robot Description and Visualization
   - Creating URDF models of simple robots
   - Visualizing robot models in rviz
   - Sensor simulation setup in Gazebo
3. Simulation and Sensor Data Handling
   - Launching Gazebo world and robot models
   - Subscribing to sensor topics and processing data
   - Writing custom messages and services
4. Motion Planning and Navigation Labs
   - Setting up MoveIt! for robot arm manipulation
   - Implementing simple navigation and path planning
   - Using SLAM packages for environment mapping
5. Integration and Project Work
   - Interfacing hardware with ROS nodes (optional hardware setups)
   - Mini-project implementing robot task using ROS
   - Debugging and visualization of robot behavior

SEMESTER-VI

COURSE 15 A: APPLICATIONS OF NLP

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. To gain practical experience in key NLP tasks including information extraction, sentiment analysis, knowledge graph construction, question answering, and machine translation.
2. To develop skills in designing and implementing NLP pipelines using LangFlow.
3. To understand evaluation techniques and performance metrics for NLP applications.
4. To prepare students for real-world NLP challenges through hands-on experimentation.
5. To enable students to build end-to-end NLP applications integrating multiple components.

Course Outcomes

By the end of the course, students will be able to:

1. Implement named entity recognition, relation extraction, and event/time extraction using NLP tools.
2. Build and evaluate sentiment analysis models incorporating various NLP features.
3. Create and query knowledge graphs for AI applications.
4. Design question answering systems and basic chatbots leveraging retrieval and neural methods.
5. Develop machine translation pipelines and evaluate translation quality effectively.

Unit 1: Information Extraction

Named Entity Recognition (NER) - concepts and techniques, Relation Extraction basics, Extracting events and temporal information, Tools and libraries overview (spaCy, HuggingFace), Evaluation metrics for information extraction

Unit 2: Sentiment Analysis

Sentiment analysis overview and use cases, Rule-based sentiment analysis, Machine learning methods for sentiment classification, Neural networks for sentiment analysis, Feature engineering and common NLP features

Unit 3: Knowledge Graph Applications

Introduction to knowledge graphs (KGs), use of KGs for AI tasks, Motivation and practical use cases of KGs, Querying and accessing knowledge graphs, Service-oriented dialog systems using KGs

Unit 4: Question Answering and Chatbots

IR-based factoid QA systems, Knowledge-based QA techniques, Neural network-based QA, Properties of human conversation and dialog flow, Basic dialog system design principles

Unit 5: Machine Translation

Statistical machine translation, Neural machine translation fundamentals, Translation for Indic languages, Popular MT tools and APIs, Evaluating translation quality

Recommended Textbooks & References

1. "Natural Language Processing with Python" by Steven Bird, Ewan Klein, and Edward Loper, O'Reilly Media
2. "Speech and Language Processing" by Daniel Jurafsky and James H. Martin, Prentice Hall
3. "Deep Learning for Natural Language Processing" by Palash Goyal, Sumit Pandey, and Karan Jain, Apress
4. LangFlow Documentation and Tutorials (https://www.langflow.org)
5. Research papers and articles on Knowledge Graphs and Neural Question Answering

Unit 1: Information Extraction

Activity: Build LangFlow pipelines for NER, relation extraction, and event/time extraction using sample datasets.

Outcome: Understand practical extraction techniques and evaluate results using standard metrics.

Evaluation Method: Lab assignment demonstrating working IE pipelines with performance reports.

Unit 2: Sentiment Analysis

Activity: Construct sentiment analysis workflows combining rule-based filters and ML/neural classifiers in LangFlow.

Outcome: Learn to engineer features and observe the impact on sentiment classification accuracy.

Evaluation Method: Practical test comparing different sentiment models; report analysis.

Unit 3: Knowledge Graph Applications

Activity: Extract entities and relationships to build knowledge graphs; perform graph queries via LangFlow-Neo4j integration.

Outcome: Master knowledge graph concepts and querying for NLP use cases.

Evaluation Method: Project submission of functional KG with query examples.

Unit 4: Question Answering and Chatbots

Activity: Develop retrieval-based and neural QA chains; design basic chatbot flows managing intents and replies.

Outcome: Ability to combine retrieval and generation methods for QA; implement conversational bots.

Evaluation Method: Lab demonstrations of QA accuracy and chatbot dialogue handling.

Unit 5: Machine Translation

Activity: Implement MT pipelines using neural models/APIs; evaluate translations on given language pairs including Indic languages.

Outcome: Grasp translation techniques and quality metrics.

Evaluation Method: Comparative analysis of translation output with BLEU scores.

SEMESTER-VI

COURSE 15 A: APPLICATIONS OF NLP

Practical **Credits: 1** **2 hrs/week**

Note: Perform this lab using Langflow platform

Experiment 1: Named Entity Recognition (NER)

- Configure LangFlow pipeline for Named Entity Recognition.
- Use pre-trained models for extracting entities from sample texts.
- Visualize entity recognition results and annotate new data in LangFlow.

Experiment 2: Relation Extraction

- Build a LangFlow workflow to extract relationships between entities.
- Use chaining of NLP components (entity detection + relation extraction).
- Test extraction on example datasets and review output quality.

Experiment 3: Event and Temporal Information Extraction

- Create a pipeline in LangFlow to extract events and time expressions.
- Combine entity extraction with rule-based temporal parsing nodes.
- Evaluate extracted event and temporal data for accuracy.

Experiment 4: Sentiment Analysis

- Design LangFlow workflow for sentiment classification using available language models.
- Incorporate rule-based filters and ML classifiers as components.
- Visualize sentiment polarity and intensity from sample inputs.

Experiment 5: Feature Engineering for Sentiment Analysis

- Use LangFlow nodes for text preprocessing and feature extraction (n-grams, POS, embeddings).
- Feed features into ML models in the pipeline.
- Analyze impact of differing features on sentiment prediction.

Experiment 6: Knowledge Graph Creation and Querying

- Use LangFlow to extract entities and relations to construct a knowledge graph.
- Integrate graph query components (e.g., using Neo4j API nodes).
- Execute and visualize graph queries within LangFlow.

Experiment 7: Question Answering Pipeline

- Build an IR-based question answering workflow using LangFlow's document retrieval and language model nodes.
- Add neural QA components for answer extraction from documents.
- Test QA system on real questions and review responses.

Experiment 8: Building Basic Chatbots

- Implement conversation flow using LangFlow chains and memory nodes.
- Manage intents, entities, and fallback mechanisms in the chatbot design.
- Test simple dialog scenarios within LangFlow.

Experiment 9: Machine Translation Workflow

- Use LangFlow connectors to translation APIs or neural MT models.
- Build a text translation pipeline supporting Indic languages.
- Evaluate translation quality and analyze errors.

Experiment 10: Model Evaluation and Optimization

- Create workflows to calculate evaluation metrics (Precision, Recall, F1, BLEU) using LangFlow components.
- Iterate model improvements and observe performance changes.

SEMESTER-VI

COURSE 15 B: DATA ENGINEERING & MLOPS

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. To introduce the lifecycle and roles in Data Engineering.
2. To explore data architecture principles, distributed systems, and technology choices.
3. To analyze MLOps features, risks, and challenges in developing ML systems.
4. To design CI/CD pipelines and deployment strategies for ML models.
5. To understand monitoring, governance, and Responsible AI compliance in production ML.

Course Outcomes

At the end of the course, students will be able to:

1. Explain Data Engineering and its organizational roles.
2. Analyze major concepts in data architecture and distributed systems.
3. Apply MLOps features and evaluate challenges in ML model development.
4. Design and implement CI/CD pipelines for ML deployment.
5. Evaluate governance and Responsible AI practices in MLOps.

Unit 1. Foundations of Data Engineering

Data Engineering: definition, lifecycle, skills, activities. Evolution and roles of Data Engineers: technical vs business responsibilities, internal vs external roles. Relationship between Data Engineering and Data Science. Data lifecycle vs Data Engineering lifecycle.

Unit 2. Data Architecture & Distributed Systems

Enterprise and Data Architecture definitions. Principles of good data architecture. Scalability, failure design, tiers, microservices, monolith vs modular. Event-driven architecture, hybrid cloud, multicloud, edge computing. Technology selection criteria: team size, interoperability, cost, TCO.

Unit 3. MLOps Fundamentals

MLOps challenges and risk mitigation. Responsible AI and scaling ML solutions. Key MLOps features: EDA, feature engineering, model training & evaluation, reproducibility. Deployment requirements, monitoring basics. Model versioning and experimentation tracking.

Unit 4. Model Deployment & CI/CD Pipelines

Preparing models for production. Runtime environments: dev to production adaptation.
CI/CD pipelines: building ML artifacts, testing pipelines. Deployment strategies: batch, online, A/B testing, canary releases. Containerization & scaling (Docker, Kubernetes).

Unit 5. Monitoring, Feedback Loops & Governance

Monitoring models in production: drift detection, ground truth evaluation.
Feedback loops: retraining workflows, online evaluation. Logging, monitoring frameworks.
Governance: regulations (GDPR, CCPA, GxP), Responsible AI principles.
Templates for governance, compliance, and model risk management.

Text/ Reference books

1. Fundamentals of Data Engineering, Joe Reis & Matt Housley, O'Reilly, 2022.
2. Introducing MLOps, Mark Treveil & Dataiku Team, O'Reilly, 2020

Web Resources:

https://www.ibm.com/think/topics/data-engineering
https://martinfowler.com/articles/microservices.html
https://towardsdatascience.com/a-gentle-introduction-to-mlops-7d64a3e890ff/

Activities

Outcome: Explain Data Engineering and roles
Activity: Prepare a concept map showing different Data Engineer roles in an organization.
Evaluation Method: Short presentation + written quiz.

Outcome: Analyze data architecture concepts
Activity: Case study on choosing between monolith, microservices, and event-driven architectures.
Evaluation Method: Case study report + viva.

Outcome: Apply MLOps features in ML development

Activity: Lab exercise on feature engineering & reproducibility using MLflow.

Evaluation Method: Lab record submission + demo.

Outcome: Design CI/CD pipelines for ML

Activity: Mini-project: build a simple ML CI/CD pipeline with GitHub Actions/Docker.

Evaluation Method: Project demo + evaluation rubric.

Outcome: Evaluate governance and Responsible AI practices

Activity: Group discussion & policy brief on GDPR/Responsible AI practices.

Evaluation Method: Written assignment + peer review.

SEMESTER-VI

COURSE 15 B: DATA ENGINEERING & MLOPS

Practical **Credits: 1** **2 hrs/week**

1. Install and configure a modern data engineering environment (Python, Jupyter, VSCode).
2. Explore and visualize the data lifecycle on a sample dataset (sales/weather).
3. ETL basics: extract data from CSV/JSON -> transform -> load into relational database.
4. Compare performance of batch vs event-driven ingestion using Apache Kafka or RabbitMQ.
5. Deploy a small dataset on Hadoop Distributed File System (HDFS) and perform simple operations.
6. Case study lab: design a microservices vs monolithic workflow for a mock business problem.
7. Perform Exploratory Data Analysis (EDA) and track experiments using **MLflow**.
8. Build a simple ML model (regression/classification) and enable **reproducibility** with version control.
9. Manage datasets and model versions using **DVC (Data Version Control)**.
10. Containerize an ML model with **Docker**.
11. Automate training and deployment with a **GitHub Actions CI/CD pipeline**.
12. Deploy an ML model as a REST API using **FastAPI / Flask**.
13. Implement model drift detection: monitor incoming data and compare with training data distribution.
14. Build a simple feedback loop: retrain a model automatically when drift exceeds a threshold.
15. Configure logging and monitoring using **Prometheus/Grafana** for a deployed ML model.
16. Case study: analyze GDPR/Responsible AI implications on a real dataset (e.g., facial recognition).

SEMESTER-VI

COURSE 15 C: ADVANCED AI AUTOMATION & ROBOTICS

Theory **Credits: 3** **3 hrs/week**

Course Objectives

- Understand advanced industrial automation systems and how AI integrates to enhance adaptive manufacturing.
- Gain expertise in autonomous navigation and robot perception using advanced SLAM and sensor fusion.
- Explore Robotic Process Automation (RPA) and Agentic Process Automation (APA) for intelligent workflow automation.
- Learn the architecture and applications of Agentic AI in cognitive robotics, emphasizing autonomous learning and decision-making.
- Study generative AI models and their applications in robotics design, simulation, and automation for adaptive and creative behavior.

Course Outcomes

Upon completion, students will be able to:

- Analyze and design AI-integrated industrial automation systems utilizing protocols and IIoT.
- Implement autonomous navigation strategies using advanced sensor fusion and SLAM techniques.
- Develop RPA and APA-based workflows using automation platforms, demonstrating AI-enhanced decision making.
- Apply Agentic AI principles to design cognitive robotic systems for autonomous learning and planning.
- Utilize generative AI models for robot design optimization, synthetic data generation, and adaptive automation.

Unit 1: Industrial Automation Systems and AI Integration

Advanced industrial automation architectures and systems, Automation protocols and communication networks (Profibus, Fieldbus, HART), Role of AI in adaptive automation and smart manufacturing, LEAN production systems and AI-driven process optimization, Cyber-physical systems and Industrial Internet of Things (IIoT)

Unit 2: Autonomous Navigation and Robot Perception

Localization and mapping with advanced SLAM algorithms, Sensor fusion techniques: LIDAR, camera, IMU integration, Dynamic obstacle detection and avoidance using AI, Path planning in dynamic environments with AI optimization, ROS2 and advanced simulation environments for autonomous robots

Unit 3: Robotic Process Automation (RPA) & Agentic Process Automation (APA)

What is RPA, Taskbot, process, API - RPA advanced workflow automation techniques, APA fundamentals: autonomous agent architectures and capabilities, AI-driven decision-making in APA workflows, Platforms and frameworks for developing APA applications

Unit 4: Agentic AI and Cognitive Robotics

What is Agentic AI - Architecture of cognitive robotic systems and agentic AI, Autonomous learning and adaptation in agentic systems, Reasoning, planning, and proactive decision-making in robotics, Human-robot collaboration with agentic AI support, Ethical, safety, and trust considerations in cognitive robotics

Unit 5: Generative AI Applications in Robotics and Automation

Overview of generative AI models and architectures, Robot design and task programming via generative AI techniques, Synthetic data generation for training robotic perception systems, Generative AI for real-time adaptive robotics behavior, Emerging applications: creative robotics, digital twins, and augmented automation

Reference Textbooks

1. Thomas R. Kurfess, Robotics and Automation Handbook, CRC Press
2. Roland Siegwart, Illah R. Nourbakhsh, Davide Scaramuzza, Autonomous Mobile Robots, MIT Press
3. Mary C. Lacity, Leslie P. Willcocks, Robotic Process Automation and Cognitive Automation, SB Publishing
4. Stuart Russell, Peter Norvig, Artificial Intelligence: A Modern Approach, Pearson
5. Morgan Quigley, Brian Gerkey, William D. Smart, Programming Robots with ROS, O'Reilly
6. Ian Goodfellow et al., Deep Learning (Generative Models section), MIT Press

Activities

Unit 1: Industrial Automation Systems and AI Integration

Activities: Study automation architectures, protocols, AI in manufacturing, LEAN systems, and IIoT applications.

Outcomes: Understand automation systems, analyze AI integration, explain protocols and IIoT.

Evaluation: Quiz, project report, and participation in discussions.

Unit 2: Autonomous Navigation and Robot Perception

Activities: Implement SLAM, sensor fusion, obstacle avoidance, path planning, and ROS2 simulation.

Outcomes: Apply SLAM and sensor fusion, develop navigation algorithms, and use ROS2 effectively.

Evaluation: Lab demo, case study, and theoretical quiz.

Unit 3: Robotic Process Automation (RPA) & Agentic Process Automation (APA)

Activities: Design RPA workflows, build APA agents, use AI for decision-making, explore platforms, and present case studies.

Outcomes: Develop RPA/APA bots, demonstrate autonomous decision making, and evaluate real applications.

Evaluation: Workflow submission, case study presentations, and lab practical.

Unit 4: Agentic AI and Cognitive Robotics

Activities: Design cognitive agents, implement reasoning and planning, simulate collaboration, discuss ethics and safety, and complete mini-projects.

Outcomes: Understand agentic AI, apply autonomous learning, collaborate with robots, and analyze ethical issues.

Evaluation: Project report, presentation, ethical discussion, and lab assignment.

Unit 5: Generative AI Applications in Robotics and Automation

Activities: Study generative models, build GAN/VAE, generate synthetic data, simulate adaptive behaviors, and research new applications.

Outcomes: Explain generative AI, apply to design and adaptation, create synthetic data, and present innovative use cases.

Evaluation: Coding assignments, research presentation, and practical demonstration.

SEMESTER-VI

COURSE 15 C: ADVANCED AI AUTOMATION & ROBOTICS

Practical **Credits: 1** **2 hrs/week**

1. Introduction to RPA Platform and Studio

- Installing and configuring UiPath Studio
- Exploring interface components and features
- Developing simple automation workflows (message boxes, input dialogs)

2. Workflow Design and Control Flows

- Building sequences and flowcharts for automation logic
- Implementing decision-making and looping constructs
- Using variables, arguments, and data types to manage data

3. UI Automation and Data Manipulation

- Recording and automating mouse and keyboard actions
- Using selectors and wildcards for reliable UI interaction
- Extracting and manipulating data from applications and Excel

4. Error Handling, Debugging, and Logging

- Implementing robust error handling using Try-Catch
- Adding logs and debugging workflows effectively
- Designing recovery mechanisms for workflow failures

5. Project Work: Browser Automation

- Automate web-based tasks: form filling, data scraping, navigation
- Handling dynamic web elements and pop-ups
- Integrate web automation with data input/output

6. Project Work: Message Automation

- Automate sending and receiving messages via platforms (supported APIs)
- Parse incoming messages and trigger responses
- Schedule messaging workflows for periodic communication

7. Project Work: Email Automation

- Automate email sending, reading, filtering, and attachments handling
- Integrate Outlook/Gmail with RPA tools
- Build email workflows for notifications, reports, and auto-responses

8. Project Work: Excel Automation

- Read/write Excel files; automate data entry and extraction
- Perform data cleaning, sorting, filtering, and formatting
- Generate reports, pivot tables, charts, and integrate with email automation
- Automate repetitive Excel tasks including formula application
- Utilize DataTables and Workbook activities for efficient Excel operations

9. Advanced Automation and Integration

- Using queues, orchestrator, and REFramework for enterprise bots
- Invoking APIs and integrating with external systems
- Deploying bots and monitoring execution