NEP BASED

SCHEME OF LEARNING

and

SYLLABUS FOR

B.TECH CSE(AI)

IV SEMESTER

| 1. **FOURTH SEMESTER B.TECH (CSAI) Scheme Of Learning …………………..3** |
| --- |
| 1. **IV SEMESTER B.TECH (CSAI) Detailed Syllabus of Core Courses ……………4** |
| **2.1 Theory of Automata & Formal Languages………………………… …………….4**  **2.2 Artificial Intelligence…………………………………………………………………9**  **2.3 Machine Learning…………………………………………………………………...14**  **2.4 Microprocessors and Microcontrollers ……………………………………………… 18**  2.5 **Optimization Techniques** …………………………………………… **………… 18** |
|  |
|  |

**1. B. Tech (CSAI) Semester-IV**

## 

| B. Tech -SEMESTER IV | | | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Course Code | Type | Course | L | T | P | Credits | Evaluation Scheme | | | | | Offering Dept. | Course  Type | |
| Theory | | | Practical | |
| AICTE | NEP-2020 |
| CA | MS | ES | CA | ES |
| CA CSC 4 01 | CC | Theory of Automata & Formal Languages | 3 | 1 | 0 | 4 | 30 | 20 | 50 | - | - | CSE | Program Core | Discipline Specific |
| CA CSC 4 02 | CC | Artificial Intelligence | 3 | 0 | 2 | 4 | - | 20 | 50 | 30 | - | CSE | Program Core | Discipline Specific |
| CA CSC 4 03 | CC | Machine Learning | 3 | 0 | 2 | 4 | - | 20 | 50 | 30 | - | CSE | Program Core | Discipline Specific |
| CA ECC 4 04 | CC | Microprocessors and Microcontrollers | 3 | 0 | 2 | 4 | - | 20 | 50 | 30 | - | ECE | Engg. Sc. | Interdisciplinary |
| CA ICC 4 05 | CC | Optimization Techniques | 3 | 1 | 0 | 4 | 30 | 20 | 50 | - | - | ICE | Engg. Sc. | Interdisciplinary |
| VAXXxxx\*\* | VAC | -- | - | - | - | NIL | 100 | - | - | - | - | - | Mandatory | VAC |
|  |  | 23 Contact Hours\*\* | | | | 20 |  | | | | | | | |
| \*\* Any one course from the approved list of VAC courses. Course code, L-T-P and evaluation scheme are given in the list of VAC courses. Actual teaching hours shall depend on LTP of the VAC course registered by the student. | | | | | | | | | | | | | | |

## 

**2. IV SEMESTER B.TECH (CSAI) Detailed Syllabus of Core Courses**

**2.1 Theory of Automata & Formal Languages**

1. **OVERVIEW OF THE COURSE:**

| Name of Course | Theory of Automata & Formal Languages |
| --- | --- |
| Offering Department | COMPUTER SCIENCE & ENGINEERING |
|  |  |

**OVERVIEW:**

The Course on the **Theory of Automata & Formal Languages** is offered to second-year students. Theory of Automata & Formal Languages course is to introduce the fundamental mathematical and computational principles that are the foundation of computer science. These include topics such as Turing machines, Automata, grammar and formal languages, decidability, halting problems, the P = NP question, and NP-completeness reductions. A secondary objective is to address students’ misconceptions that the Theory of Computation is disconnected from state-of-the-art applications and today’s open problems; once they complete a theory course, however, they understand that the fundamental theorems and proofs of theory provide the foundation material for all specialties of computer science. For example, the Theory of Automata & Formal Languages introduces conceptual tools that practitioners use in compiler and programming languages. A final and related objective of a TOC course is to prepare students to be either well-rounded practitioners or potential candidates for computer science graduate programs. Determining whether a Theory of Automata & Formal Languages course prepares students for graduate school is straightforward: approximately 1/3 of the problems on the GATE test for computer science require an understanding of the Theory of Automata & Formal Languages.

1. ***SYLLABUS***

| **COURSE NO** | **TITLE OF THE COURSE** | **COURSE STRUCTURE** | **PRE-REQUISITE** |
| --- | --- | --- | --- |
| CACSC 401 | Theory of Automata & Formal Languages | 3L - 1T - 0P | None |
| **COURSE OUTCOMES (COs)**  After completion of this course, the students are expected to be able to demonstrate the following knowledge, skills, and attitudes:  CO1: Understand formal languages, grammars, and automata, and analyze their relationships with computational models.  CO2: Apply concepts of finite automata and regular expressions to design and optimize regular language recognizers.  CO3: Construct context-free grammars and pushdown automata for context-free languages, and solve problems related to ambiguity and normal forms.  CO4: Analyze and design Turing machines to model complex computational problems and understand the limitations of computation.  CO5: Evaluate the decidability and computational complexity of problems, distinguishing between tractable and intractable problems, and explore P, NP, and NP-hard classifications. | | | |
| **COURSE CONTENTS**  **Unit I**  **Finite Automata and Regular Languages**: Deterministic FA, Non-deterministic FA, Regular expressions, Finite Automaton with €- moves, Regular Expression, Regular Languages, and Kleene’s theorem– Conversion of NFA to DFA, Equivalence of finite Automaton and regular expressions, Arden’s Theorem. Myhill Nerode Theorem, Minimization of DFA, Pumping Lemma for Regular sets, Problems based on Pumping Lemma  **Unit II**  **Context-Free Grammar and Languages:** Grammar, Types of Grammar, Context-Free Grammars and Languages, Derivations, Ambiguity, Relationship between derivation and derivation trees, Simplification of CFG, Elimination of Useless symbols - Unit productions - Null productions, Chomsky normal form (CNF), Greibach Normal form (GNF), Problems related to CNF and GNF  **Unit III**  **Pushdown Automata and Context-Free Languages:** Moves, Instantaneous descriptions, Deterministic/Non-deterministic, pushdown automata, Equivalence of Pushdown automata and CFL, pumping lemma for CFL, problems based on pumping Lemma, Conversion from CFG-to-PDA, Conversion from PDA-to-CFG  **Unit IV**  **Turing Machines and Computability:** Definition and Types of Turing machines, Computable languages and functions, Techniques for Turing machine construction, Multi-head and Multi tape Turing Machines, The Halting problem, Partial Solvability, Problems about Turing machine- Chomsky hierarchy of languages  **Unit V**  **Decidable and Undecidable Problems in Computability:** Unsolvable Problems and Computable Functions, Recursive and recursively enumerable languages, Universal Turing machine, Measuring and classifying complexity - Tractable and Intractable problems, P, NP and NP-hard, Decidable and Undecidable problems | | | |
| **SUGGESTED READINGS**   1. Hopcroft J.E., Motwani R., and Ullman J.D, “Introduction to Automata Theory, Languages, and Computations”, Second Edition, Pearson Education. 2. John C Martin, “Introduction to Languages and the Theory of Computation”, Third Edition, Tata McGraw Hill Publishing Company, New Delhi 3. Marvin L. Minsky “Computation: Finite and Infinite” – Prentice Hall, 1967 4. Michael Sipser “Introduction to the Theory of Computation”, Third Edition, 2012 Cengage Learning 5. Peter Lenz – An Introduction to Formal Languages and Automata – 3rd Edition Narosa, 2003 6. Thomas A. Sukamp – An introduction to the theory of computer science languages and machines – 3rd edition, Pearson Education, 2007. 7. G E Reevsz -“Introduction to Formal Languages” TMH, 2000. | | | |

***C. CO-PO & CO-PSO MAPPING TABLE***

| **CO\PO** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PSO1** | **PSO2** | **PSO3** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CO1** | 3 | 3 | 2 | 3 | 2 | 0 | 0 | 0 | 2 | 2 | 0 | 3 | 3 | 2 | 0 |
| **CO2** | 2 | 2 | 3 | 2 | 2 | 0 | 0 | 0 | 2 | 2 | 0 | 3 | 2 | 3 | 0 |
| **CO3** | 2 | 2 | 3 | 2 | 2 | 0 | 0 | 0 | 2 | 2 | 0 | 3 | 2 | 3 | 0 |
| **CO4** | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 3 | 3 | 0 | 3 | 3 | 3 | 1 |
| **CO5** | 3 | 3 | 2 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 3 | 3 | 2 | 2 |

***D. THEORY LECTURE PLAN***

| **S. No.** | **CONTENT** | **NUMBER**  **OF**  **LECTURES** | **Unit** |
| --- | --- | --- | --- |
| 1 | Introduction & Motivation of TOC, Infinite Sets, Closures, Alphabets, Languages & Representation | 1 | **Unit-I**  **(11)** |
| 2 | Finite Automata: Deterministic FA | 1 |
| 3 | Nondeterministic FA | 1 |
| 4 | Regular expressions | 1 |
| 5 | Finite Automaton with €- moves | 1 |
| 6 | Regular Expression, Regular Languages and Kleene’s theorem– Conversion of NFA to DFA | 1 |
| 7 | Equivalence of finite Automaton and regular expressions, Minimization of DFA | 2 |
| 8 | Arden’s Theorem. Myhill Nerode | 1 |
| 9 | Pumping Lemma for Regular sets, Problems based on Pumping Lemma | 2 |
| **CLASS TEST-I** | | | |
| 10 | Context Free Grammar: Grammar, Types of Grammar | 1 | **Unit-II**  **(6)** |
| 11 | Context Free Grammars and Languages, Derivations, Ambiguity | 1 |
| 12 | Relationship between derivation and derivation trees | 1 |
| 13 | Elimination of Useless symbols - Unit productions - Null productions | 1 |
| 14 | Chomsky normal form (CNF), | 1 |
| 15 | Greibach Normal form (GNF), Problems related to CNF and GNF | 1 |
| 16 | Pushdown Automata Moves, Instantaneous descriptions, Deterministic/non-deterministic pushdown automata | 2 | **Unit-III**  **(7)** |
| 17 | Conversion from CFG-to-PDA | 1 |
| 18 | Conversion from PDA-to-CFG | 1 |
| **MID SEMESTER EVALUATION** | | | |
| 19 | Equivalence of Pushdown automata and CFL | 1 |  |
| 20 | Pumping lemma for CFL, problems based on pumping Lemma | 2 |
| 21 | Definition and Types of Turing machines, Computable languages and functions | 2 | **Unit-IV**  **(9)**  **Unit-V**  **(7)** |
| 22 | Techniques for Turing machine construction | 2 |
| 23 | Multi head and Multi tape Turing Machines | 1 |
| 24 | The Halting problem, Partial Solvability | 1 |
| 25 | Problems about Turing machine | 2 |
| 26 | Chomsky hierarchy of languages | 1 |
| 27 | Unsolvable Problems and Computable Functions | 1 |
| 28 | Recursive and recursively enumerable languages | 1 |
| 29 | Universal Turing machine | 1 |
| 30 | Measuring and classifying complexity - Tractable and Intractable problems | 1 |
| 31 | Church-Turing Thesis & Universal Turing Machines | 1 |
| 32 | P, NP, and NP-hard problems | 1 |
| 33 | Decidable and Undecidable problems | 1 |

**E. SELF STUDY**

| **Sr. No.** | **Topic** | **Unit** |
| --- | --- | --- |
| 1 | (i)How Vedic or philosophical concepts might challenge, inspire, or provide insight into traditional computational theory?  (ii)How does the deterministic nature of finite automata (DFA) reflect the Vedic concept of following a clear, preordained path (Svadharma), where every decision is predetermined and fixed? | **Unit I** |
| 2 | (i)What are the foundational principles of "computation" from a Vedic perspective?  (ii)In Vedic philosophy, "Brahman" represents an underlying, universal truth. How might context-free grammars (CFG) represent the underlying structure of languages, transcending surface-level variations?  (iii)The Vedic concept of "Chitta" (mind) can store infinite possibilities. How does the stack in a pushdown automaton (PDA) symbolize the mind's capacity to process and store infinite possibilities within finite constraints? | **Unit II & III** |
| 3 | (i)Can Vedic mathematical sutras lead to optimal algorithms for complex computations?  (ii)How does the concept of undecidability in Turing machines challenge the Vedic understanding of "Maya" (illusion), where some truths may forever remain beyond our grasp? | **Unit IV** |
| 4 | Could Vedic concepts of infinity and cyclic time inform complexity theory? | **Unit V** |

* 1. **Artificial Intelligence**

1. **OVERVIEW OF THE COURSE:**

| Name of Course | Artificial Intelligence |
| --- | --- |
| Offering Department | COMPUTER SCIENCE & ENGINEERING |
|  |  |

#### OVERVIEW**:**

The **Artificial Intelligence (AI)** course is designed for students to understand the foundational concepts, problem-solving techniques, and applications of AI. The course emphasizes intelligent agents, search strategies, knowledge representation, reasoning, and metaheuristic optimization methods. Through this course, students will develop skills in building intelligent systems, solving complex problems, and utilizing heuristic and optimization strategies. Additionally, they will gain insights into the practical and theoretical aspects of AI, including risks, benefits, and ethical considerations. The course also introduces key concepts of metaheuristics and equips students to understand and implement AI-based solutions in real-world applications.

This course ensures that students are well-prepared to engage in AI-driven innovation, develop AI applications, and gain a strong understanding of the methodologies that underpin modern artificial intelligence systems

#### SYLLABUS

| **COURSE NO** | **TITLE OF THE COURSE** | **COURSE STRUCTURE** | **PRE-REQUISITE** |
| --- | --- | --- | --- |
| CACSC402 | Artificial Intelligence | 3L - 0T - 2P | Design and Analysis of Algorithms |
| **COURSE OUTCOMES (COs)**  After completion of this course, the students are expected to be able to demonstrate the following knowledge, skills, and attitudes:   1. Understand the basic concepts, history, and ethical considerations of AI and intelligent agents, enabling the analysis of rational behaviour in different environments. 2. Apply search algorithms and optimization techniques to solve structured and complex problems, including those in unknown or partially observable environments. 3. Implement adversarial search algorithms and solve constraint satisfaction problems by applying appropriate inference and backtracking methods. 4. Design intelligent agents using propositional and first-order logic for reasoning and inferencing in structured environments. 5. Develop solutions to problems using advanced knowledge representation methods. | | | |
| **COURSE CONTENTS**  **Unit I**  **Introduction:** What is AI, Foundations of Artificial Intelligence, History of Artificial Intelligence, State of the Art, Risks and Benefits of AI  **Intelligent Agents:** Agents and Environments, Good Behavior - The Concept of Rationality, Nature of Environments, Structure of Agents  **Unit II**  **Problem-solving -I**  Solving Problems by Searching Problem-Solving Agents, Example Problems, Search Algorithms, Uninformed Search Strategies, Informed (Heuristic) Search Strategies, Heuristic Functions, Basics of Metaheuristics, Genetic Algorithm, Grey Wolf Optimizer and Applications.  **Search in Complex Environments:** Local Search and Optimization Problems, Local Search in Continuous Spaces, Search with Nondeterministic Actions, Search in Partially Observable Environments, Online Search Agents and Unknown Environments  **Unit III**  **Problem-solving -II**  Adversarial Search and Games Game Theory, Optimal Decisions in Games, Heuristic Alpha--Beta Tree Search, Limitations of Game Search Algorithms  **Constraint Satisfaction Problems**: Defining Constraint Satisfaction Problems, Constraint Propagation: Inference in CSPs, Backtracking Search for CSPs, Local Search for CSPs, The Structure of Problems  **Unit IV**  **Knowledge, reasoning and representation**  Logical Agents Knowledge-Based Agents, The Wumpus World, Logic, Propositional Logic: A Very Simple Logic, Propositional Theorem Proving, Effective Propositional Model Checking, Agents Based on Propositional Logic First-Order Logic Syntax and Semantics of First-Order Logic, Knowledge Engineering in First-Order Logic Inference in First-Order Logic Propositional vs. First-Order Inference, Unification and First-Order Inference, Forward Chaining, Backward Chaining, Resolution  **Unit V**  **Ontologies and Knowledge Representation**  Ontological Engineering, Categories and Objects, Events, Mental Objects and Modal Logic, Reasoning Systems for Categories, Reasoning with Default Information, knowledge graphs | | | |
| **SUGGESTED READINGS**   1. Stuart Russell & Peter Norvig, [Artificial Intelligence: A Modern Approach](http://aima.cs.berkeley.edu/), Prentice-Hall, Fourth Edition (2021). 2. Ian GoodFellow, Yoshua Bengio & Aaron Courville, [Deep Learning](http://www.deeplearningbook.org/), MIT Press (2016). 3. Elaine Rich, Kevin Knight & Shivashankar B Nair, Artificial Intelligence, Tata McGraw-Hill (2009) | | | |
|  | | | |

#### C. CO-PO & CO-PSO MAPPING TABLE

| **CO\PO** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PSO1** | **PSO2** | **PSO3** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CO1 | 3 | 0 | 1 | 1 | 1 | 3 | 2 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| CO2 | 1 | 3 | 3 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 3 | 3 | 2 |
| CO3 | 1 | 3 | 3 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 3 | 3 | 2 |
| CO4 | 1 | 3 | 3 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 3 | 3 | 1 |
| CO5 | 2 | 3 | 3 | 3 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 3 | 1 |

#### D. THEORY LECTURE PLAN

| **S.No.** | **CONTENT** | **NUMBER**  **OF**  **LECTURES** | **Unit** |
| --- | --- | --- | --- |
| 1 | What is AI, Foundations of Artificial Intelligence | 1 | **Unit-I**  **(5)** |
| 2 | History of Artificial Intelligence, State of the Art, Risks and Benefits of AI | 1 |
| 3 | Agents and Environments, Good Behavior - The Concept of Rationality | 2 |
| 4 | Nature of Environments, Structure of Agents | 1 |
| 5 | Solving Problems by Searching Problem-Solving Agents, Example Problems, Search Algorithms, | 2 | **Unit-II**  **(12)** |
| 6 | Uninformed Search Strategies | 2 |
| 7 | Informed (Heuristic) Search Strategies | 2 |
| 8 | Heuristic Functions, Basics of Metaheuristics | 1 |
| 9 | Genetic Algorithm | 1 |
| 10 | Grey Wolf Optimizer and Applications. | 1 |
| 11 | Search in Complex Environments: Local Search and Optimization Problems, Local Search in Continuous Spaces | 2 |
| 12 | Search with Nondeterministic Actions, Search in Partially Observable Environments | 1 |
| 13 | Problem-solving -II  Adversarial Search and Games Game Theory | 1 | **Unit III**  **(9)** |
| 14 | Optimal Decisions in Games | 1 |
| 15 | Heuristic Alpha--Beta Tree Search, Limitations of Game Search Algorithms | 2 |
| 16 | Constraint Satisfaction Problems: Defining Constraint Satisfaction Problems, | 2 |
| 17 | Constraint Propagation: Inference in CSPs | 2 |
| 18 | Local Search for CSPs, The Structure of Problems | 1 |
| 19 | Knowledge, reasoning and representation, Logical Agents Knowledge-Based Agents | 1 | **Unit-IV**  **(8)** |
| 20 | Logic, Propositional Logic: A Very Simple Logic | 1 |
| 21 | Propositional Theorem Proving, Effective Propositional Model Checking | 1 |
| 22 | Agents Based on Propositional Logic First-Order Logic Syntax and Semantics of First-Order Logic | 1 |
| 23 | Knowledge Engineering in First-Order Logic Inference in First-Order Logic Propositional vs. First-Order Inference | 2 |
| 24 | Unification and First-Order Inference, Forward Chaining, Backward Chaining, Resolution | 2 |
| 25 | Knowledge Representation: Ontological Engineering | 2 | **Unit-V**  **(6)** |
| 26 | Categories and Objects, Events, Mental Objects and Modal Logic | 2 |
| 27 | Reasoning Systems for Categories, Reasoning with Default Information, knowledge graphs | 2 |

1. **LESSON PLAN for LAB**

| **Lab Class No.** | **Name of the Experiment** |
| --- | --- |
| 1. | Implement and compare the performance of Breadth-First Search (BFS) and Depth-First Search (DFS) on a maze or graph traversal problem. |
| 2. | Write a program to implement uninformed search algorithms; uniform cost search and bidirectional search, depth limited search |
| 3. | Write a program to implement heuristic-based search algorithms; A\* and Greedy Best-First Search. |
| 4. | Implement a Genetic Algorithm to solve an optimization-based, path-planning problem. Observe how mutation and crossover parameters affect performance. |
| 5. | Implement the Grey Wolf Optimization algorithm for an optimization problem, comparing its performance with the Genetic Algorithm and analyzing its convergence behavior. |
| 6. | Implement local search algorithm Hill Climbing on a problem with multiple local optima. |
| 7. | Develop a simple two-player game (e.g., Tic-Tac-Toe) using Minimax and Alpha-Beta pruning algorithms. Show how optimal strategies are derived in adversarial environments. |
| 8. | Implement a backtracking algorithm for solving a constraint satisfaction problem, the N-Queens problem, Sudoku. |
| 9. | Write a program to solve the N-Queens problem using local search strategies (e.g., min-conflicts heuristic) and analyze the effectiveness compared to backtracking. |
| 10. | Simulate a basic Wumpus World environment and implement a simple agent that navigates through it using propositional logic to avoid dangers (e.g., pits, Wumpus). Track the agent’s decisions and learning process. |
| 11. | Implementation of Travelling Salesman Problem1 |
| 12. | Implementation of Missionaries-Cannibals Problem |

### F. SELF STUDY

| **Sr. No.** | **Topic** |
| --- | --- |
| 1. | Online Search Agents and Unknown Environments |
| 2. | Ethics in AI |

* 1. **Machine Learning**

1. ***OVERVIEW OF THE COURSE:***

| Name of Course | Machine Learning |
| --- | --- |
| Offering Department | COMPUTER SCIENCE & ENGINEERING |
|  |  |

**OVERVIEW:**

The course on Machine Learning provides students with foundational knowledge and hands-on experience in developing intelligent algorithms that can learn from data. It covers key concepts and techniques, including supervised and unsupervised learning, neural networks, and model evaluation. This course emphasizes practical applications, aiming to prepare students to build and apply machine learning models to solve real-world problems.

1. ***SYLLABUS***

| **COURSE NO** | **TITLE OF THE COURSE** | **COURSE STRUCTURE** | **PRE-REQUISITE** |
| --- | --- | --- | --- |
| COCSC403 | Machine Learning | 3L - 0T - 2P | Probability and Statistics, Linear Algebra |
| **COURSE OUTCOMES (COs)**  After completion of this course, students are expected to:   1. Understand foundational concepts in machine learning, focusing on supervised learning methods. 2. Implement and evaluate supervised machine learning models using relevant metrics. 3. Apply neural network models to practical tasks and understand model training techniques. 4. Project-based learning to Develop machine learning solutions to real-world challenges, interpreting and presenting results effectively. | | | |
| **COURSE CONTENTS**  **Unit I**  **Introduction to Machine Learning (ML)**: Introduction to Artificial Intelligence, History and Applications, Definition and Types of ML: Supervised, Unsupervised, and Reinforcement Learning, ML Workflow and Model Development Cycle, Data Preprocessing: Data Cleaning, Handling Missing Values, Encoding Categorical Data, Data Normalisation, Feature Scaling. Introduction to Python ML Libraries: Scikit-Learn, Pandas, Numpy, Dimensionality Reduction (PCA)  **Unit II**  **Regression Techniques in Supervised Learning**: Linear Regression: Simple and Multiple Linear Regression, Polynomial Regression: Basics and Use Cases, Regularization Techniques: Ridge and Lasso Regression, Feature Engineering, Model Evaluation Metrics for Regression: MSE, MAE, R-Squared, RMSE, Bias-Variance Tradeoff.  **Unit III**  **Classification Techniques in Supervised Learning**: Classification Basics: Definitions and Applications, Logistic Regression and Decision Boundaries, k-Nearest Neighbors (k-NN), Naïve Bayes, Support Vector Machines (SVM), Decision Trees and Random Forests for Classification, Ensemble Learning with Boosting Techniques, Model Evaluation Metrics: Accuracy, Precision, Recall, F1 Score, Confusion Matrix, ROC-AUC.  **Unit IV**  **Neural Networks and Training Techniques**: Introduction to Neural Networks: Perceptron, Activation Functions, Training Neural Networks: Gradient Descent and Ascent, Backpropagation, and Cost Functions, Optimizers, Overfitting and Regularization in Neural Networks, Basics of Deep Learning Architectures: Overview of CNN and RNN  **Unit V**  **Real-World Applications, Optimization, and Introduction to Unsupervised Learning**: Case Studies in Image Classification, Predictive Analytics, and Recommendation Systems, Model Optimization: Hyperparameter Tuning and Cross-Validation, Introduction to Unsupervised Learning: Clustering, Insights from Industry Applications and Emerging Trends. | | | |
| **SUGGESTED READINGS**   1. **Tom Mitchell**, *Machine Learning*, McGraw-Hill, 1997. 2. Aurelien Geron, **Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow**, O'Reilly Media, 2nd Edition, 2019. 3. Ian Goodfellow, Yoshua Bengio, Aaron Courville, **Deep Learning**, MIT Press, 1st Edition, 2016. 4. Peter Flach, **Machine Learning: The Art and Science of Algorithms that Make Sense of Data**, Cambridge University Press, 1st Edition, 2012. 5. Andreas C. Müller, Sarah Guido, **Introduction to Machine Learning with Python: A Guide for Data Scientists**, O'Reilly Media, 1st Edition, 2016. | | | |

***C. CO-PO & CO-PSO MAPPING TABLE***

| **CO\PO** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO 10** | **PO11** | **PO12** | **PSO1** | **PSO2** | **PSO3** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CO1 | 1 | 3 | 0 | 1 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 3 | 2 | 2 |
| CO2 | 3 | 3 | 2 | 2 | 3 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 3 | 2 | 2 |
| CO3 | 3 | 3 | 1 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 3 | 2 |
| CO4 | 3 | 3 | 1 | 3 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 2 | 2 | 3 |

***D.THEORY LECTURE PLAN***

| **Sno** | **CONTENT** | **NUMBER**  **OF**  **LECTURES** | **Unit** |
| --- | --- | --- | --- |
| 1 | Introduction to Artificial Intelligence, History and Applications | 1 | **Unit-I**  **(6)** |
| 2 | Definition and Types of ML: Supervised, Unsupervised, and Reinforcement Learning | 1 |
| 3 | ML Workflow and Model Development Cycle | 1 |
| 4 | Data Preprocessing: Data Cleaning, Handling Missing Values, Encoding Categorical Data, Data Normalisation | 2 |
| 5 | Introduction to Python ML Libraries: Scikit-Learn, Pandas, Numpy. | 1 |
| 6 | Linear Regression: Simple and Multiple Linear Regression | 1 |
| 7 | Polynomial Regression: Basics and Use Cases | 1 |
| 8 | Regularization Techniques: Ridge and Lasso Regression | 1 |
| 9 | Feature Engineering, Dimensionality reduction (PCA) | 3 |
| **CLASS TEST-I** | | | |
| 10 | Model Evaluation Metrics for Regression: MSE, MAE, R-Squared, RMSE, Bias-Variance Tradeoff. | 2 | **Unit-II**  **(6)** |
| 11 | Classification Basics: Definitions and Applications | 1 |
| 12 | Logistic Regression and Decision Boundaries | 1 |
| 13 | k-Nearest Neighbors (k-NN) | 1 |
| 14 | Naïve Bayes | 1 |
| 15 | Support Vector Machines (SVM) | 1 |
| 16 | Decision Trees and Random Forests for Classification | 1 |
| 17 | Ensemble Learning with Boosting Techniques | 2 | **Unit-III**  **(10)** |
| 18 | Model Evaluation Metrics: Accuracy, Precision, Recall, F1 Score, Confusion Matrix, ROC-AUC. | 2 |
| **MID SEMESTER EVALUATION** | | | |
| 24 | Introduction to Neural Networks | 1 | **Unit-IV**  **(9)** |
| 25 | Perceptron, Activation Functions | 1 |
| 26 | Training Neural Networks: Gradient Descent and Ascent | 1 |
| 27 | Back-propagation, and Cost Functions | 1 |
| 28 | Optimizers, Overfitting and Regularization in Neural Networks | 2 |
| 29 | Basics of Deep Learning Architectures: Overview of CNN and RNN | 1 |
| 30 | Expert Topic from Industry | 1 |
| 31 | Flipped Learning | 1 |
| 32 | Project Discussion | 1 |
| 33 | Case Studies in Image Classification | 1 | **Unit-V**  **(7)** |
| 34 | Predictive Analytics, and Recommendation Systems | 1 |
| 35 | Model Optimization: Hyperparameter Tuning and Cross-Validation | 1 |
| 36 | Unsupervised Learning: Clustering | 2 |
| 37 | Insights from Industry Applications and Emerging Trends. | 2 |

1. ***LESSON PLAN for LAB***

| **Lab Class No.** | **Name of the Experiment** | |
| --- | --- | --- |
| 1 | Introduction to Python Lab, Platform, Basic Concepts, List, Tuple, Dictionary, Set. | |
| 2 | Python Advance Libraries, Numpy, Pandas, Matplotlib, scikit-learn. | |
| 3 | Linear Regression | |
| 4 | Logistic Regression | |
| 5 | Naïve Bayes | |
| 6 | SVM | |
| **MID SEMESTER EVALUATION** | | |
| 7 | Decision Tree & Random Forest | |
| 8 | Ensemble learning with boosting technique | |
| 9 | Neural Networks | |
| **CLASS TEST-2 (LAB EVALUATION)** | | |
| 10 | Project Based Learning | * 1. Problem definition for mini project, selecting an ML problem, building an appropriate model |
| 11 | * 1. Evaluate model accuracy, effectiveness, and interpretation of results. |
| 12 | * 1. Mini Project Discussion and Evaluation |
| **END SEMESTER EVALUATION** | | |

1. ***SELF STUDY***

| **Sr. No.** | **Topic** |  |
| --- | --- | --- |
| 1 | **Visualizing Gradient Descent**: Create a Python script that shows the gradient descent process for a simple linear regression model. Use it to see how the model minimizes errors with each step. | Unit II |
| 2 | **Reducing Dimensions with PCA**: Use Principal Component Analysis (PCA) on a dataset with many features to reduce it to two dimensions. Visualize the results and interpret how PCA helps simplify data while keeping important information. | Unit II & III & V |
| 3 | **Vedic Mathematics for Predictive Text Generation**: Using the "Ekadhikena Purvena" rule in Vedic mathematics, generate a sequence based on this pattern and develop a simple predictive model to continue the sequence. Analyze how well the model captures the pattern. | Unit V |
| 4 | Any topic as suggested by Course instructor |  |

2.4 The course **CA ECC 4 04 Microprocessors and Microcontrollers** will be offered by ECE deptt and therefore the syllabus for the same will be designed and approved by their respective BOS.

2.5 The course **CA ICC 4 05 Optimization Techniques** will be offered by ICE deptt and therefore, the syllabus for the same will be designed and approved by their respective BOS.