# Dr. B. C. Roy Engineering College, Durgapur

## Department of CSE(AIML)

| Field | Details |
|---|---|
| Course Name | AI, Ethics, Society |
| Course Code | AM-401 |
| Semester | 4 |
| Course Category | Program Core Courses |
| Credits | 3 |
| Hours per Week | 3L:0T:0P |

## 1. Prerequisites

• Introductory programming and data structures (e.g., Python fundamentals)

• Basic probability and statistics for understanding data and model evaluation

• Foundations in philosophy/ethics or a social-science course covering ethical reasoning

## 2. Course Learning Objectives

• Cultivate a deep, interdisciplinary understanding of ethical theories, professional codes, and societal contexts that shape responsible AI development and deployment.

• Enable students to critically analyze the entire AI lifecycle--from data acquisition to model deployment--identifying points of ethical risk and proposing mitigation strategies grounded in technical and normative frameworks.

• Develop the ability to evaluate, document, and communicate AI system properties (bias, fairness, transparency, accountability) using industry-standard tools such as model-cards, data-sheets, and impact assessments.

• Prepare students to engage with emerging legal and governance regimes (e.g., EU AI Act, GDPR, ISO/IEC standards) and to formulate policy-informed recommendations for high-risk AI applications.

• Foster interdisciplinary collaboration skills that empower students to lead responsible AI initiatives, bridging computer science with law, sociology, and public policy to build trustworthy AI solutions.

## 3. Teaching Methodology

• Lectures and Presentations

• Interactive Discussions and Case Studies

• Lab Sessions

• Guest Lectures

## 4. Evaluation System

| Activities | Class Test Full Marks | Assignment Full Marks | Attendance Full Marks | Total Marks |
|---|---|---|---|---|
| CIA-1 | 25 | 10 | 05 | 40 |
| CIA-2 | 25 | 10 | 05 | 40 |
| End Semester Examination (ESE) | - | - | - | 60 |
| Total | | | | 100 |

## 5. Course Modules

| Module | Topics | Hours |
|---|---|---|
| 1 | **Foundations of Ethics and the Evolution of AI**<br><br>- History of ethics (Western, Indian, Buddhist, Islamic)<br>- Major ethical theories: Kantian deontology, Consequentialism, Utilitarianism, Virtue Ethics, Ethics of Care, Ubuntu, Distributive Justice<br>- Professional codes of conduct: ACM Code of Ethics, ALA Code of Ethics & Core Values, IEEE Ethically Aligned Design<br>- Introduction to AI: definition, brief historical | 8 |

| | | |
|---|---|---|
| | milestones, generative vs. predictive AI<br>- Societal dimensions and motivations for AI ethics<br>- Overview of the AI lifecycle (data -> model -> deployment) to frame ethical considerations | |
| 2 | **Technical Foundations of AI and Core Ethical Challenges**<br><br>- Supervised learning basics (classification, regression) and predictive analytics<br>- Unsupervised learning basics and high-level overview of generative AI concepts (GANs, VAEs, diffusion models)<br>- Ethical issues in generative AI: synthetic data, deep-fakes, authenticity<br>- Bias in predictive models and real-world consequences<br>- The AI alignment problem: embedding human values, moral machines (conceptual level)<br>- Overview of AI decision-making pipelines (data -> model -> deployment) and points of ethical risk | 7 |
| 3 | **Data Ethics, Privacy, Bias, and Fairness**<br><br>- Data ownership, informed consent, and ethical data handling practices<br>- Privacy-enhancing concepts: differential privacy, federated learning (principles, not mathematics)<br>- Key data-protection regulations: GDPR, CCPA, and emerging global frameworks<br>- Sources of bias in data collection, labeling, and model training<br>- Bias detection techniques (statistical tests, model-card documentation)<br>- Mitigation strategies (pre-processing, in-processing, post-processing)<br>- Fairness concepts: statistical, individual, and group fairness<br>- Fairness metrics and evaluation methods | 7 |
| 4 | **Accountability, Transparency, Explainability & Legal Landscape**<br><br>- The black-box problem and interpretability techniques (feature importance, LIME, SHAP)<br>- Explainable AI: technical methods and emerging legal requirements (e.g., EU AI Act)<br>- Accountability frameworks: algorithmic impact assessments, model-cards, documentation standards<br>- Legal landscape governing AI: intellectual property, liability, AI-specific regulations, GDPR implications<br>- Transparency, power, and agency in AI deployments | 6 |

| | | |
|---|---|---|
| | - Ethical guidelines & standards for explainable and accountable AI (IEEE 7010, ISO/IEC 42001) | |
| 5 | **Autonomous Systems, Societal Impacts & Controversial Applications**<br><br>- Ethical challenges in autonomous vehicles, drones, and robots (trolley problem, moral decision-making)<br>- AI and the future of work: automation, inequality, reskilling strategies<br>- Case studies of AI failures (biased policing, hiring algorithms, deep-fake misuse)<br>- Military and surveillance applications of AI (autonomous weapons, mass monitoring)<br>- AI in creative industries and biomedical research (authorship, data sharing, reproducibility)<br>- Responsibility and accountability for autonomous decision-making<br>- Emerging governance approaches for high-risk AI systems | 7 |
| 6 | **Governance, Policy, and Responsible AI Practice**<br><br>- International and national AI governance initiatives (OECD AI Principles, UNESCO Recommendation, EU AI Act)<br>- Human-rights-centered AI design and normative models<br>- Policy recommendations, regulatory challenges, and future directions<br>- Interdisciplinary collaboration for AI ethics (law, sociology, computer science)<br>- Responsible AI engineering practices and ethical AI lifecycle (model-cards, data-sheets, impact assessments)<br>- Development of AI ethics guidelines, standards, and certification (ISO/IEC 42001, IEEE 7010)<br>- Strategies for building public trust, societal engagement, and addressing public concerns | 7 |

## 6. References

**Textbooks:**


**1.** Müller, Vincent C. "Ethics of artificial intelligence and robotics." (2020).Müller, Vincent C. "Ethics of artificial intelligence and robotics." (2020).

**2.** Markus D. Dubber, Frank Pasquale, Sunit Das. "The Oxford Handbook of Ethics of AI." Oxford University Press Edited book, 2020

**Reference Books:**

**1.** S. Matthew Liao. Ethics of Artificial Intelligence‖, Oxford University Press Edited Book, 2020

## 7. Course Outcomes

| ID | Statement | Action Verb | Knowledge Level |
|---|---|---|---|
| AM-401.1 | List at least five major ethical theories (e.g., Kantian deontology, Utilitarianism, Virtue Ethics, Ubuntu, Ethics of Care) and three professional codes of conduct (ACM, ALA, IEEE) that are relevant to AI development. | List | Remember |
| AM-401.2 | Explain the AI lifecycle (data -> model -> deployment) and identify three ethical considerations associated with each stage. | Explain | Understand |
| AM-401.3 | Apply bias detection techniques (e.g., statistical parity tests, model-card documentation) to a provided dataset and implement at least one mitigation strategy (pre-processing, in-processing, or post-processing) to improve fairness metrics. | Apply | Apply |
| AM-401.4 | Analyze the legal and accountability frameworks governing AI (e.g., GDPR, EU AI Act, IEEE 7010) and produce a concise report that maps at least four regulatory requirements to corresponding points of risk in an AI decision-making pipeline. | Analyze | Analyze |
| AM-401.5 | Evaluate the effectiveness of two explainability methods (e.g., LIME, SHAP) for a black-box predictive model by comparing their impact on transparency, stakeholder trust, and compliance with emerging legal standards. | Evaluate | Evaluate |
| AM-401.6 | Design a responsible AI solution for a high-risk application (e.g., autonomous vehicle, AI-enabled hiring system) that integrates ethical guidelines, governance policies, and an algorithmic impact assessment, and present a | Design | Create |

| | | | |
|---|---|---|---|
| | prototype with documented model-cards and data-sheets. | | |

## 8. CO-PO Mapping

| CO | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1 | 1 | 1 | 1 | 1 | - | 2 | 1 | 3 | - | 1 | - | 1 |
| CO2 | 1 | 2 | 1 | 1 | 2 | 3 | 1 | 3 | - | 2 | - | 2 |
| CO3 | 1 | 2 | 3 | 2 | 3 | 2 | 1 | 3 | 1 | 2 | 1 | 2 |
| CO4 | 1 | 2 | 1 | 2 | 1 | 3 | 1 | 3 | 1 | 3 | 1 | 2 |
| CO5 | 1 | 2 | 2 | 2 | 2 | 2 | 1 | 3 | 1 | 2 | 1 | 2 |
| CO6 | 1 | 2 | 3 | 2 | 2 | 3 | 1 | 3 | 2 | 3 | 2 | 2 |

## 9. CO-PSO Mapping

| CO | PSO1 | PSO2 | PSO3 |
|---|---|---|---|
| CO1 | 2 | 1 | 3 |
| CO2 | 3 | 1 | 2 |
| CO3 | 2 | 3 | 3 |
| CO4 | 2 | 1 | 3 |
| CO5 | 2 | 2 | 3 |
| CO6 | 3 | 3 | 3 |

# Dr. B. C. Roy Engineering College, Durgapur

## Department of CSE(AIML)

| Field | Details |
|---|---|
| Course Name | Design and Analysis of Algorithm |
| Course Code | AM-402 |
| Semester | 4 |
| Course Category | Program Core Courses |
| Credits | 3 |
| Hours per Week | 3L:0T:0P |

## 1. Prerequisites

• Proficiency in a high-level programming language (e.g., Python, Java, C++)

• Fundamentals of discrete mathematics (sets, functions, proofs, logarithms, basic combinatorics, limits)

• Introductory data structures and basic algorithm concepts (arrays, linked lists, stacks/queues, recursion, simple sorting)

## 2. Course Learning Objectives

• This course introduces students to fundamental concepts and applications of the subject

• Students will learn theoretical foundations and practical skills relevant to the field

## 3. Teaching Methodology

• Lectures and Presentations

• Interactive Discussions and Case Studies

• Lab Sessions

• Guest Lectures

## 4. Evaluation System

| Activities | Class Test Full Marks | Assignment Full Marks | Attendance Full Marks | Total Marks |
|---|---|---|---|---|
| CIA-1 | 25 | 10 | 05 | 40 |
| CIA-2 | 25 | 10 | 05 | 40 |
| End Semester Examination (ESE) | - | - | - | 60 |
| Total | | | | 100 |

## 5. Course Modules

| Module | Topics | Hours |
|---|---|---|
| 1 | **Foundations of Algorithms and Complexity**<br><br>- What is an algorithm? - algorithm as a technology, design & analysis<br>- Models of computation (RAM), termination and correctness<br>- Measuring performance: best-, worst- and average-case<br>- Asymptotic notation: O, Ω, Θ, little-o, little-ω and conditional forms<br>- Review of essential mathematics (sets, functions, logarithms, basic limits)<br>- Basic abstract data types (ADTs): arrays, linked lists, stacks, queues, records & pointers<br>- Introduction to hash tables and binary-search trees (fundamental for AI data handling)<br>- Overview of elementary data-structure concepts (usage, strengths, limitations) | 5 |
| 2 | **Recurrences and Divide-and-Conquer Techniques**<br><br>- Recurrence relations and why they arise in algorithm analysis<br>- Methods for solving recurrences: substitution, recursion-tree, Master theorem<br>- Divide-and-Conquer paradigm and its design steps<br>- Binary search and its recurrence solution<br>- Merge sort - algorithm, recurrence, and analysis | 6 |

| | | |
|---|---|---|
| | - Quick sort (deterministic) - partitioning, recurrence, and analysis<br>- QuickSelect (deterministic) for order statistics<br>- Heap data structure: heap property, building a heap, priority-queue operations<br>- Heap sort - algorithm and analysis<br>- Linear-time integer sorting: counting sort, radix sort, bucket sort<br>- Linear-time selection (median-of-medians) - deterministic linear-time algorithm<br>- Multiplication of large integers (Karatsuba) - divide-and-conquer integer multiplication<br>- *Note:* Strassen's matrix multiplication removed (requires advanced linear-algebra background) | |
| 3 | **Greedy Algorithms and Priority-Queue Structures**<br><br>- Greedy algorithm paradigm - characteristics and proof techniques<br>- Activity-selection (interval scheduling) problem<br>- Making change problem - greedy coin systems<br>- Huffman coding - construction and optimality proof<br>- Minimum-spanning-tree algorithms (Kruskal's and Prim's) - classic greedy applications<br>- Greedy task-scheduling and other standard greedy problems<br>- Priority queues implemented with binary heaps (reinforcement from Module 2)<br>- Dijkstra's shortest-path algorithm presented as a greedy technique<br>- Analysis of greedy algorithms (correctness, optimality, complexity) | 7 |
| 4 | **Dynamic Programming and Amortized Analysis**<br><br>- Dynamic programming fundamentals: optimal substructure, overlapping subproblems, memoization<br>- Classic DP problems:<br>* Rod-cutting<br>* Matrix-chain multiplication<br>* Longest common subsequence (LCS)<br>* 0/1 knapsack and subset-sum<br>* Independent set in trees<br>* Assembly-line scheduling<br>* Floyd-Warshall (all-pairs shortest paths) as a DP example<br>* Computing binomial coefficients<br>- Amortized analysis techniques: aggregate, accounting, potential methods<br>- Dynamic tables (array resizing) as a case study of | 8 |

| | | |
|---|---|---|
| | amortized analysis | |
| 5 | **Advanced Graph Algorithms and Network Flows**<br><br>- Graph representations: adjacency lists, adjacency matrices<br>- Elementary graph traversals: BFS, DFS<br>- Topological sorting of DAGs<br>- Strongly connected components (Kosaraju/Tarjan)<br>- Minimum-spanning-tree recap (Kruskal, Prim) - integration with graph framework<br>- Single-source shortest-path algorithms:<br>* Dijkstra's algorithm (non-negative weights)<br>* Bellman-Ford (negative edges, detection of negative cycles)<br>* Shortest paths in DAGs<br>- All-pairs shortest paths: Floyd-Warshall algorithm (reinforced from DP module)<br>- Maximum-flow fundamentals: flow networks, residual graph, augmenting paths<br>- Ford-Fulkerson method and Edmonds-Karp implementation<br>- Maximum bipartite matching as a flow problem | 8 |
| 6 | **NP-Completeness, Approximation, Randomization & Backtracking (AI-Focused)**<br><br>- Complexity classes: P, NP, PSPACE; definitions of NP-hard and NP-complete<br>- Reducibility and Cook's theorem (conceptual overview)<br>- Classic NP-complete decision problems: SAT, Clique, Vertex-cover, k-Coloring, Hamiltonian circuit, Traveling Salesperson, Subset-sum, Decision-knapsack, Assignment problem<br>- Approximation algorithms for selected NP-hard problems (greedy set-cover, MST-based TSP heuristic)<br>- Randomized algorithms:<br>* Randomized quicksort and quickselect analysis<br>* Rabin-Karp string matching (Monte-Carlo)<br>* Distinction between Las-Vegas and Monte-Carlo approaches<br>- String-matching algorithms: naïve, Knuth-Morris-Pratt, finite-automata method<br>- Backtracking techniques and examples: N-Queens (4-queen, 8-queen), Hamiltonian circuit<br>- Branch-and-bound framework with case studies: 15-puzzle, Traveling Salesperson<br>- AI-relevant heuristic search: A* algorithm and admissible heuristics (brief introduction) | 8 |

## 6. References

**Textbooks:**

**1.** Aho, Hopcroft, Ullman "Design & Analysis of Computer Algorithms", Pearson Education

**Reference Books:**

**1.** S. Sridhar "Design and Analysis of Algorithms", OUP

## 7. Course Outcomes

| ID | Statement | Action Verb | Knowledge Level |
|---|---|---|---|
| AM-402.1 | Recall the definitions of an algorithm, the RAM computational model, and the asymptotic notations O, Ω, Θ, o, and ω, as well as the properties and operations of basic abstract data types such as arrays, linked lists, stacks, queues, hash tables, and binary-search trees. | Recall | Remember |
| AM-402.2 | Explain the divide-and-conquer and greedy paradigms, including how to formulate recurrence relations, solve them with substitution, recursion-tree, and the Master theorem, and describe the step-by-step design process for greedy algorithms. | Explain | Understand |
| AM-402.3 | Implement and correctly execute at least three classic algorithms (e.g., merge sort, heap sort, Dijkstra's shortest-path) on given input instances and compute their worst-case time and space complexities. | Implement | Apply |
| AM-402.4 | Analyze and compare the amortized cost of dynamic table resizing using the aggregate, accounting, and potential methods, and evaluate the efficiency of alternative algorithmic solutions (e.g., counting sort versus radix sort) for specified problem sizes. | Analyze | Analyze |
| AM-402.5 | Evaluate the suitability of graph- | Evaluate | Evaluate |

| | | | |
|---|---|---|---|
| | based and flow-based algorithms, as well as approximation or heuristic methods (e.g., MST-based TSP, A* search), for solving AI/ML data-handling problems such as nearest-neighbor search or model selection, providing justification based on computational complexity and solution quality. | | |
| AM-402.6 | Design a customized algorithmic solution that integrates a heuristic search (e.g., A* with an admissible heuristic) and appropriate data structures to address a defined AI/ML task (e.g., feature selection or clustering), and produce a correctness proof together with a formal complexity analysis of the solution. | Design | Create |

## 8. CO-PO Mapping

| CO | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1 | 3 | 2 | 1 | 1 | 2 | - | - | - | - | - | - | 2 |
| CO2 | 3 | 3 | 2 | 2 | 2 | - | - | - | 1 | 2 | 1 | 2 |
| CO3 | 3 | 3 | 3 | 2 | 3 | - | - | - | 2 | 2 | 2 | 2 |
| CO4 | 3 | 3 | 3 | 3 | 3 | - | - | - | 2 | 2 | 2 | 2 |
| CO5 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 3 | 2 | 2 |
| CO6 | 3 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 3 | 3 | 3 | 3 |

## 9. CO-PSO Mapping

| CO | PSO1 | PSO2 | PSO3 |
|---|---|---|---|
| CO1 | 3 | 1 | 1 |
| CO2 | 3 | 2 | 1 |
| CO3 | 3 | 3 | 1 |
| CO4 | 3 | 2 | 1 |
| CO5 | 3 | 3 | 2 |
| CO6 | 3 | 3 | 2 |

# Dr. B. C. Roy Engineering College, Durgapur

## Department of CSE(AIML)

| Field | Details |
|---|---|
| Course Name | Operating System |
| Course Code | AM-403 |
| Semester | 4 |
| Course Category | Program Core Courses |
| Credits | 3 |
| Hours per Week | 3L:0T:0P |

## 1. Prerequisites

• Fundamental computer architecture concepts (CPU organization, memory hierarchy, I/O basics)

• Proficiency in a systems programming language such as C/C++ and familiarity with Unix/Linux command-line environment

• Basic understanding of data structures, algorithms, and discrete mathematics (e.g., linked lists, trees, complexity analysis)

## 2. Course Learning Objectives

• Enable students to articulate the fundamental architecture and components of modern operating systems, linking historical evolution to current OS families and their roles in supporting diverse computing environments.

• Develop students' ability to analyze and compare process and thread abstractions, including creation, lifecycle, scheduling, and isolation mechanisms, with emphasis on their impact on performance and scalability of AI and parallel workloads.

• Equip students with a deep conceptual understanding of synchronization, deadlock, and memory-management techniques, allowing them to evaluate trade-offs among primitives, algorithms, and hardware support in real-world system design.

• Prepare students to assess storage, I/O, and security subsystems--including file-system structures, device scheduling, protection models, and virtualization/containerization--and to recommend appropriate OS-level strategies for reliability, performance, and isolation of modern applications.

• Foster the capacity to synthesize OS concepts across modules to design, troubleshoot, and optimize end-to-end system solutions, demonstrating how scheduling, concurrency control, memory management, and I/O interact in complex, high-performance computing scenarios.

## 3. Teaching Methodology

• Lectures and Presentations

• Interactive Discussions and Case Studies

• Lab Sessions

• Guest Lectures

## 4. Evaluation System

| Activities | Class Test Full Marks | Assignment Full Marks | Attendance Full Marks | Total Marks |
|---|---|---|---|---|
| CIA-1 | 25 | 10 | 05 | 40 |
| CIA-2 | 25 | 10 | 05 | 40 |
| End Semester Examination (ESE) | - | - | - | 60 |
| Total | | | | 100 Marks |

## 5. Course Modules

| Module | Topics | Hours |
|---|---|---|
| 1 | **Foundations of Operating Systems**<br><br>- Introduction & history of operating systems<br>- Basic organization of a computing system (CPU, memory, I/O)<br>- OS components & services: process, memory, I/O, file, security<br>- OS structures & models: monolithic, micro-kernel, hybrid, hypervisor (overview only) | 5 |

| | | |
|---|---|---|
| | - System-call interface, user-OS interaction and standard libraries<br>- Boot process & system initialization (BIOS/UEFI, bootloader, init)<br>- Interrupt handling<br>- Overview of common OS families (UNIX/Linux, Windows, real-time, embedded) | |
| 2 | **Processes and Threads**<br><br>- Process concept, states, and PCB<br>- Process creation & termination (fork/exec, Windows CreateProcess, exit)<br>- Process control operations: wait, signal, context-switch<br>- Inter-process communication: shared memory, pipes, sockets, UNIX signals<br>- Thread fundamentals: definition, life-cycle, POSIX threads and Windows threads<br>- Multicore programming issues and thread-scheduling models (1:1, M:N)<br>- Process isolation, namespaces and lightweight containers (conceptual overview)<br>- Comparison of process vs. thread abstractions and typical AI workload patterns | 6 |
| 3 | **Process Scheduling and Concurrency Control**<br><br>- Scheduling goals: throughput, latency, fairness, response time<br>- Classic algorithms (FCFS, SJF, RR, priority, multilevel feedback)<br>- Multiprocessor scheduling basics and load-balancing strategies<br>- Context-switch mechanics and overhead considerations<br>- Fairness, priority inversion and simple mitigation (priority inheritance)<br>- Introduction to concurrency sources (multiprocessors, interrupts, I/O)<br>- Critical-section problem and basic atomic operations<br>- Simple real-time scheduling concept (Earliest-Deadline-First) | 7 |
| 4 | **Synchronization Primitives and Deadlock**<br><br>- Critical-section problem and correctness criteria<br>- Classic software solutions (Peterson's, Dekker, Lamport's bakery)<br>- Hardware primitives: test-and-set, compare-and-swap, memory barriers<br>- Synchronization objects: mutexes, spinlocks, semaphores, condition variables, monitors, barriers | 8 |

| | | |
|---|---|---|
| | - Common patterns: producer-consumer, readers-writers, dining philosophers<br>- Deadlock model: resource-allocation graph, four necessary conditions<br>- Deadlock handling: prevention, avoidance (banker's algorithm overview), detection & recovery<br>- Intro to lock-free / wait-free ideas and atomic primitives used in AI frameworks | |
| 5 | **Memory Management and Virtual Memory**<br><br>- Memory-management background and hardware support (MMU, protection bits)<br>- Allocation strategies: swapping, contiguous allocation, fragmentation<br>- Segmentation and paging fundamentals; hierarchical & inverted page tables<br>- TLB role and basic performance impact<br>- Virtual memory: demand paging, copy-on-write, page-fault handling<br>- Page-replacement policies (FIFO, LRU, Clock) - conceptual only<br>- Working-set model, thrashing, frame allocation policies<br>- Memory protection techniques (ASLR, guard pages) and NUMA awareness for AI workloads<br>- Kernel memory allocation, memory-mapped files and brief intro to garbage collection | 8 |
| 6 | **Storage, I/O, Protection, Security & Advanced OS Topics**<br><br>- Storage hierarchy: magnetic disks, SSDs, RAID levels, NFS & cloud object storage<br>- Disk structure, basic scheduling (elevator, SSTF, C-LOOK) and DMA vs. programmed I/O<br>- File-system concepts: files, directories, links, namespace, permission bits<br>- File-system implementation: allocation methods, free-space management.<br>- Crash recovery basics, journaling and consistency checking<br>- Protection & security models: UNIX permissions, ACLs, capabilities, basic encryption<br>- Common OS-level attacks and mitigation (buffer overflow, privilege escalation)<br>- OS support for accelerators (GPU/TPU scheduling, cgroups, resource quotas) and brief look at cloud-native OS services | 8 |

## 6. References

**Textbooks:**

**1.** Silberschatz, A, Galvin, P.B., Gagne G., "Operating System Concepts" 9th edition, Wiley Publishers, 2016.

**2.** The Design of the Unix Operating System - Maurice Bach, Prentice Hall, 1988.

**3.** William Stallings, Operating Systems: Internals and Design Principles. Prentice-Hall, 6th Ed., 2008

**Reference Books:**

**1.** AS Tanenbaum, Modern Operating Systems, 3rd Ed., Pearson, 2009

**2.** Operating Systems: Principles and Practice, Thomas Anderson and Michael Dahlin, Recursive Books, 2014

## 7. Course Outcomes

| ID | Statement | Action Verb | Knowledge Level |
|---|---|---|---|
| AM-403.1 | Identify and list the fundamental components, services, and historical milestones of modern operating systems, including CPU, memory, I/O subsystems, system-call interface, boot process, and common OS families. | Identify | Remember |
| AM-403.2 | Explain the concepts of processes and threads, their life-cycle operations, and how they are used to implement typical AI/ML workloads such as data preprocessing pipelines and model inference. | Explain | Understand |
| AM-403.3 | Apply at least two classic scheduling algorithms (e.g., Round-Robin, Multilevel Feedback Queue) and one synchronization primitive (e.g., mutex or semaphore) to construct a functional prototype that coordinates producer-consumer tasks for a small-scale neural-network inference service. | Apply | Apply |
| AM-403.4 | Analyze the performance impact | Analyze | Analyze |

| | | | |
|---|---|---|---|
| | of different synchronization mechanisms (spinlocks, mutexes, lock-free structures) and memory-management policies (paging, NUMA placement, working-set sizing) on the training time of a medium-size deep-learning model. | | |
| AM-403.5 | Evaluate the effectiveness of OS-level isolation and resource-control techniques (containers, cgroups, virtualization, ACLs, ASLR) for securely deploying AI models in a multi-tenant environment, citing measurable security and throughput metrics. | Evaluate | Evaluate |
| AM-403.6 | Design and implement an optimized OS configuration--including scheduling policy selection, memory-allocation tuning, I/O scheduler choice, and security hardening--that maximizes throughput and minimizes latency for a specified AI workload, and justify each design decision with quantitative performance data. | Design | Create |

## 8. CO-PO Mapping

| CO | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1 | 2 | 1 | 1 | 1 | 1 | - | - | - | - | - | - | 1 |
| CO2 | 2 | 2 | 1 | 1 | 2 | - | - | - | - | - | - | 1 |
| CO3 | 2 | 2 | 3 | 2 | 3 | - | - | - | 1 | - | 2 | - |
| CO4 | 2 | 3 | 2 | 3 | 3 | - | - | - | 1 | - | 2 | 1 |
| CO5 | 2 | 3 | 2 | 3 | 3 | 3 | 1 | 3 | 1 | 2 | 2 | 1 |
| CO6 | 2 | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 3 | 2 |

## 9. CO-PSO Mapping

| CO | PSO1 | PSO2 | PSO3 |
|---|---|---|---|
| CO1 | 2 | 1 | 1 |
| CO2 | 3 | 2 | 1 |
| CO3 | 3 | 3 | 1 |
| CO4 | 3 | 2 | 1 |
| CO5 | 2 | 3 | 3 |
| CO6 | 3 | 3 | 2 |

# Dr. B. C. Roy Engineering College, Durgapur

## Department of CSE(AIML)

| Field | Details |
|---|---|
| Course Name | Machine Learning |
| Course Code | AM 404 |
| Semester | 4 |
| Course Category | Program Core Courses |
| Credits | 3 |
| Hours per Week | 3L:0T:4P |

## 1. Prerequisites

• Linear Algebra

• Probability and Statistics

• Programming Proficiency (Python recommended)

## 2. Course Learning Objectives

• To provide students with a foundational understanding of machine learning principles, encompassing supervised, unsupervised, and reinforcement learning paradigms.

• To equip students with the practical skills to implement and evaluate various supervised learning algorithms, including linear models, tree-based methods, and neural networks.

• To foster students' ability to critically analyze and select appropriate machine learning models based on dataset characteristics, problem context, and performance metrics.

• To introduce students to the core concepts of deep learning architectures (CNNs and RNNs) and their applications in diverse domains, while emphasizing practical understanding over rigorous mathematical derivations.

• To cultivate awareness of the ethical implications and societal impact of artificial intelligence, promoting responsible development and deployment of machine learning systems.

## 3. Teaching Methodology

• Lectures and Presentations

• Interactive Discussions and Case Studies

• Lab Sessions

• Guest Lectures

## 4. Evaluation System

| **Activities** | **Class Test Full Marks** | **Assignment Full Marks** | **Attendance Full Marks** | **Total Marks** |
|---|---|---|---|---|
| CIA-1 | 25 | 10 | 05 | 40 |
| CIA-2 | 25 | 10 | 05 | 40 |
| End Semester Examination (ESE) | - | - | - | 60 |
| Total | | | | 100 Marks |

## 5. Course Modules

| **Module** | **Topics** | **Hours** |
|---|---|---|
| 1 | **Introduction to Machine Learning and Supervised Learning**<br><br>- Introduction to Machine Learning: Types of ML (Supervised, Unsupervised, Reinforcement), Applications in AI<br>- Supervised Learning Fundamentals: Regression vs. Classification<br>- Linear Models: Linear Regression and Least-Squares Fitting, Model Evaluation Metrics (MSE, | 10 |

| | | |
|---|---|---|
| | RMSE)<br>- Model Evaluation and Selection: Train-Test Split, Cross-Validation (k-fold), Bias-Variance Trade-off, Overfitting vs. Underfitting, Confusion Matrix, Precision, Recall, F1-Score, AUC-ROC | |
| 2 | **Supervised Learning Algorithms I: Classification**<br><br>- Logistic Regression for Binary and Multi-class Classification<br>- Regularization (L1/L2): Impact on model complexity and generalization<br>- Optimization Methods: Gradient Descent (Batch, Stochastic), Learning Rate Schedules<br>- Naive Bayes Classifier: Application and assumptions<br>- k-Nearest Neighbors: Distance metrics, parameter selection | 6 |
| 3 | **Supervised Learning Algorithms II: Trees and Ensembles**<br><br>- Decision Trees: Information Gain, Gini Impurity, Pruning<br>- Random Forests: Bagging, Random Subspace<br>- Gradient Boosting Machines (GBM): Boosting concept, example algorithms (e.g., XGBoost, LightGBM) - high-level overview<br>- Model Selection and Hyperparameter Tuning: Grid Search, Random Search, Cross-validation | 7 |
| 4 | **Introduction to Neural Networks**<br><br>- Perceptrons and Multilayer Perceptrons (MLPs): Forward propagation<br>- Activation Functions (Sigmoid, ReLU, Softmax)<br>- Backpropagation Algorithm: Intuition and process<br>- Loss Functions (MSE, Cross-entropy)<br>- Optimization Algorithms: Stochastic Gradient Descent (SGD), Adam (high-level understanding)<br>- Practical Considerations: Data preprocessing, feature scaling | 7 |
| 5 | **Unsupervised Learning and Deep Learning Architectures (Introduction)**<br><br>- Unsupervised Learning: Clustering (k-Means Algorithm), Dimensionality Reduction (PCA - conceptual understanding)<br>- Convolutional Neural Networks (CNNs): High-level overview of architecture and applications in image processing (no detailed mathematical | 6 |

| | | |
|---|---|---|
| | derivations)<br>- Recurrent Neural Networks (RNNs): High-level introduction and applications in sequential data (no detailed mathematical derivations) | |
| 6 | **AI Applications, Ethical Considerations & Reinforcement Learning (Introduction)**<br><br>- Applications of ML/DL in AI: Image recognition, natural language processing, robotics (brief overview)<br>- Ethical Considerations in AI: Bias in data, fairness, accountability, transparency<br>- Reinforcement Learning: Markov Decision Processes (MDPs) - conceptual overview, Q-learning (high-level understanding), no detailed mathematical derivations | 6 |

## 6. References

**Textbooks:**

**1.** Géron, A. (2022). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3rd ed.). O'Reilly Media.

**2.** Burkov, A. (2019). The Hundred-Page Machine Learning Book. Andriy Burkov.

**Reference Books:**

**1.** Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). Foundations of Machine Learning (2nd ed.). MIT Press.

**2.** Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

## 7. Course Outcomes

| ID | Statement | Action Verb | Knowledge Level |
|---|---|---|---|
| AM 404.1 | Students will be able to explain the fundamental concepts of machine learning, differentiating between supervised, unsupervised, and reinforcement learning, and identify various applications of machine learning in AI. | Explain | Understand |
| AM 404.2 | Students will be able to apply appropriate supervised learning | Apply | Apply |

| | | | |
|---|---|---|---|
| | algorithms (linear regression, logistic regression, Naive Bayes, k-NN, decision trees, random forests, and gradient boosting machines) to solve classification and regression problems, selecting and evaluating models using metrics such as MSE, RMSE, precision, recall, F1-score, and AUC-ROC. | | |
| AM 404.3 | Students will be able to implement and interpret the results of model evaluation techniques such as train-test split, cross-validation, and hyperparameter tuning (grid search, random search) to mitigate overfitting and underfitting in supervised learning models. | Implement | Apply |
| AM 404.4 | Students will be able to analyze the architecture and functionality of neural networks, including perceptrons, MLPs, CNNs, and RNNs, describing the roles of activation functions, backpropagation, and optimization algorithms (SGD, Adam). | Analyze | Analyze |
| AM 404.5 | Students will be able to evaluate and compare different unsupervised learning techniques, such as k-means clustering and PCA, and discuss their applications in data analysis and dimensionality reduction. | Evaluate | Analyze |
| AM 404.6 | Students will be able to synthesize knowledge from various modules to critically analyze ethical considerations in AI development and deployment, and provide a high-level overview of reinforcement learning concepts, including MDPs and Q-learning. | Synthesize | Evaluate |

## 8. CO-PO Mapping

| CO | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1 | 3 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | - | 1 | - | 1 |
| CO2 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 2 | 2 | 1 | 2 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 2 | 2 | 1 | 2 |
| CO4 | 3 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 2 | 1 | 2 |
| CO5 | 3 | 3 | 2 | 3 | 3 | 1 | 1 | 1 | 2 | 2 | 1 | 2 |
| CO6 | 2 | 2 | 2 | 2 | 1 | 3 | 2 | 3 | 2 | 2 | 1 | 1 |

## 9. CO-PSO Mapping

| CO | PSO1 | PSO2 | PSO3 |
|---|---|---|---|
| CO1 | 3 | 1 | 1 |
| CO2 | 3 | 3 | 1 |
| CO3 | 3 | 3 | 1 |
| CO4 | 3 | 2 | 1 |
| CO5 | 3 | 2 | 1 |
| CO6 | 2 | 1 | 3 |

# Dr. B. C. Roy Engineering College, Durgapur

## Department of CSE(AIML)

| Field | Details |
|---|---|
| Course Name | Mathematics for Machine Learning |
| Course Code | AM-405 |
| Semester | 4 |
| Course Category | Basic Science Courses |
| Credits | 3 |
| Hours per Week | 3L:0T:0P |

## 1. Prerequisites

• Proficiency in Python programming (including NumPy and pandas)

• Fundamental mathematics: single-variable calculus, linear algebra, and basic probability/statistics

• Introductory exposure to machine learning concepts (e.g., supervised vs. unsupervised learning)

## 2. Course Learning Objectives

• Equip students with a solid mathematical foundation--including linear algebra, calculus, and probability--that underpins modern AI and machine-learning techniques.

• Enable learners to design, implement, and critically evaluate a wide spectrum of supervised and unsupervised models using Python's scientific stack (NumPy, pandas, scikit-learn, and deep-learning frameworks).

• Develop the ability to apply rigorous optimization and regularization strategies to train reliable models, diagnose convergence issues, and prevent over-fitting.

• Foster analytical skills for interpreting model behavior, assessing performance with appropriate metrics, and communicating results responsibly, with attention to ethical and fairness considerations.

• Prepare students to transition from textbook algorithms to contemporary AI research and industry practice by exposing them to emerging concepts such as attention mechanisms, transformer architectures, and uncertainty quantification.

## 3. Teaching Methodology

• Lectures and Presentations

• Interactive Discussions and Case Studies

• Lab Sessions

• Guest Lectures

## 4. Evaluation System

| **Activities** | **Class Test Full Marks** | **Assignment Full Marks** | **Attendance Full Marks** | **Total Marks** |
|---|---|---|---|---|
| CIA-1 | 25 | 10 | 05 | 40 |
| CIA-2 | 25 | 10 | 05 | 40 |
| End Semester Examination (ESE) | - | - | - | 60 |
| Total | | | | 100 Marks |

## 5. Course Modules

| Module | Topics | Hours |
|---|---|---|
| 1 | **Linear Algebra Foundations for AI**<br><br>- Vectors, matrices and basic operations (addition, multiplication, transpose)<br>- Systems of linear equations and matrix inversion<br>- Rank, singularity, linear independence, and basis concepts<br>- Vector spaces, subspaces, dimension, and change of basis<br>- Linear transformations and their matrix | 6 |

| | | |
|---|---|---|
| | representations<br>- Norms, inner product, orthogonality, orthonormal bases<br>- Gram-Schmidt process and orthogonal projections<br>- Eigenvalues, eigenvectors, diagonalization<br>- Singular Value Decomposition (SVD) and low-rank approximation<br>- Practical computation with NumPy (matrix factorizations, condition numbers) | |
| 2 | **Calculus and Optimization Techniques**<br><br>- Differentiation of univariate functions and basic limits<br>- Partial derivatives, gradient, directional derivative<br>- Jacobian and Hessian matrices (interpretation)<br>- Multivariate Taylor series, linearization, and chain rule<br>- Unconstrained optimization: gradient descent, Newton's method, momentum<br>- Simple learning-rate schedules and early stopping concepts<br>- Constrained optimization basics: Lagrange multipliers<br>- Convex sets, convex functions, and fundamental convex optimization ideas<br>- Introductory view of regularization as an optimization constraint | 7 |
| 3 | **Probability & Statistics for AI**<br><br>- Probability space, axioms, sum and product rules<br>- Conditional probability, Bayes' theorem, independence<br>- Key discrete and continuous distributions (Bernoulli, Gaussian, exponential family)<br>- Expectation, variance, covariance, and correlation<br>- Sampling methods, descriptive statistics, and data summarization<br>- Hypothesis testing, confidence intervals, p-values, A/B testing<br>- Maximum Likelihood Estimation (MLE) and MAP estimation<br>- Basic Bayesian inference and conjugate priors (conceptual overview)<br>- Bias-variance trade-off, overfitting vs. underfitting, and model selection criteria | 6 |
| 4 | **Linear Models & Dimensionality Reduction**<br><br>- Data representation, feature engineering, and exploratory data analysis with NumPy & pandas<br>- Train/validation/test split, feature scaling and preprocessing | 8 |

| | | |
|---|---|---|
| | - Linear regression and multivariate least-squares solution<br>- Regularization techniques: ridge (L2) and lasso (L1)<br>- Logistic regression for binary classification and cross-entropy loss<br>- Principal Component Analysis (PCA): variance perspective, eigen computation, projection, low-rank approximation<br>- Linear Discriminant Analysis (LDA): theory and eigen-solution<br>- Model evaluation: k-fold cross-validation, bootstrapping, ROC/AUC, precision-recall, F1 score<br>- Introduction to model interpretability (coefficients, feature importance) | |
| 5 | **Classical Supervised Learning Algorithms**<br><br>- Support Vector Machines: hard-margin, soft-margin, kernel trick, dual formulation<br>- Decision trees and regression trees: splitting criteria, pruning, handling missing values<br>- Ensemble methods: bagging, random forests, boosting (AdaBoost, Gradient Boosting), basic hyper-parameter tuning<br>- Perceptron and other linear classifiers<br>- Classification evaluation measures: confusion matrix, precision/recall, ROC/AUC, F1 score<br>- Hyper-parameter tuning basics: grid search, validation sets, simple cross-validation<br>- Model interpretability tools: feature importance, SHAP/partial dependence plots | 7 |
| 6 | **Deep Learning, Unsupervised Learning & Uncertainty**<br><br>- Neural network fundamentals: activation functions (ReLU, sigmoid, softmax), loss functions (cross-entropy, MSE)<br>- Forward pass, backpropagation, and automatic differentiation<br>- Optimization for deep nets: SGD, momentum, Adam (overview)<br>- Regularization in deep learning: dropout, batch normalization, weight decay<br>- Convolutional neural network basics (convolution, pooling, simple architectures)<br>- Expectation-Maximization (EM) algorithm and Gaussian Mixture Models<br>- Clustering algorithms: k-means, hierarchical clustering, density-based (DBSCAN/BIRCH)<br>- Uncertainty quantification basics (Monte-Carlo dropout, predictive intervals) | 8 |

| | | |
|---|---|---|
| | - Snapshot of current trends: transformer encoder-decoder concept, attention mechanism<br>- Ethical considerations and responsible AI deployment (bias, fairness, reproducibility) | |

## 6. References

**Textbooks:**

**1.** Mathematics for Machine Learning by Marc Peter, Aldo Faisal, Cheng Soon, Cambridge University Press

**2.** Gilbert Strang, Linear Algebra and Learning from Data, Wellesley-Cambridge Press (2019)

**3.** Linear Algebra and Optimization for Machine Learning by Charu C Aggarwal, Springer

**Reference Books:**

**1.** The elements of statistical learning by Trevor Hastie, Robert Tibshirani, Jerome Friedman, Springer.

**2.** W. Cheney, Analysis for Applied Mathematics. New York: Springer Science+Business Medias, 2001.

## 7. Course Outcomes

| ID | Statement | Action Verb | Knowledge Level |
|---|---|---|---|
| AM-405.1 | Recall and accurately state the definitions, properties, and computational formulas for vectors, matrices, eigenvalues, gradients, Jacobians, and basic probability concepts required for AI/ML algorithms. | Recall | Remember |
| AM-405.2 | Explain the role of linear algebra, calculus, and statistical inference in formulating machine-learning models, including how matrix operations, gradient descent, and maximum-likelihood estimation are derived and interpreted. | Explain | Understand |
| AM-405.3 | Implement from scratch (using NumPy/pandas) at least three classical supervised-learning | Implement | Apply |

| | | | |
|---|---|---|---|
| | algorithms--linear regression, logistic regression, and a support-vector machine--and achieve a minimum of 75% classification accuracy on a provided benchmark dataset. | | |
| AM-405.4 | Analyze model performance by applying k-fold cross-validation, ROC/AUC, precision-recall, and bootstrapping techniques, and compare the suitability of linear models, tree-based ensembles, and kernel methods for different data characteristics. | Analyze | Analyze |
| AM-405.5 | Evaluate deep-learning architectures (e.g., CNNs) and unsupervised methods (e.g., k-means, Gaussian mixture models) by tuning hyper-parameters, measuring convergence with Adam/SGD, and quantifying predictive uncertainty using Monte-Carlo dropout or confidence intervals. | Evaluate | Evaluate |
| AM-405.6 | Create a reproducible end-to-end AI solution that integrates data preprocessing, model selection, hyper-parameter optimization, and ethical safeguards (bias detection, fairness metrics), and defend the design choices in a technical report meeting specified performance and responsible-AI criteria. | Create | Create |

## 8. CO-PO Mapping

| CO | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1 | 3 | 2 | 1 | 2 | 1 | 1 | - | - | - | - | - | 2 |
| CO2 | 3 | 3 | 2 | 2 | 2 | 1 | - | - | - | 2 | - | 2 |
| CO3 | 3 | 2 | 3 | 3 | 3 | 1 | - | 1 | 2 | 2 | 1 | 2 |
| CO4 | 3 | 3 | 2 | 3 | 3 | 1 | - | 1 | 2 | 3 | 2 | 2 |
| CO5 | 3 | 3 | 3 | 3 | 3 | 1 | - | 1 | 2 | 2 | 2 | 3 |
| CO6 | 3 | 3 | 3 | 3 | 3 | 3 | 2 | 3 | 3 | 3 | 3 | 3 |

## 9. CO-PSO Mapping

| CO | PSO1 | PSO2 | PSO3 |
|---|---|---|---|
| CO1 | 3 | 2 | 1 |
| CO2 | 3 | 2 | 1 |
| CO3 | 2 | 3 | 1 |
| CO4 | 3 | 2 | 1 |
| CO5 | 2 | 3 | 1 |
| CO6 | 3 | 3 | 3 |

# Dr. B. C. Roy Engineering College, Durgapur

## Department of CSE(AIML)

| Field | Details |
|---|---|
| Course Name | Data Handling and Data Visualization |
| Course Code | AM-497 |
| Semester | 4 |
| Course Category | Program Core Courses |
| Credits | 2 |
| Hours per Week | 0L:0T:4P |

## 1. Prerequisites

• Basic Python Programming

• Introductory Statistics

• Fundamentals of Data Analysis

## 2. Course Learning Objectives

• To equip students with a comprehensive understanding of the principles and practices of data visualization within the context of artificial intelligence, encompassing both theoretical foundations and practical application.

• To develop students' ability to effectively select, create, and interpret various data visualizations for different stages of the AI workflow, from data exploration and model building to result communication and explainability.

• To foster students' critical thinking skills in evaluating the quality, ethical implications, and communicative effectiveness of data visualizations used in AI, promoting responsible and unbiased AI development.

• To provide students with hands-on experience using a range of Python-based data visualization tools and libraries, enabling them to create reproducible, shareable, and insightful visualizations for AI applications.

• To enhance students' understanding of Explainable AI (XAI) and its relationship to data visualization, enabling them to leverage visualization techniques for interpreting and communicating complex AI models and their decisions.

## 3. Teaching Methodology

• Lectures and Presentations

• Interactive Discussions and Case Studies

• Lab Sessions

• Guest Lectures

## 4. Evaluation System

| **Activities** | **Class Test Full Marks** | **Assignment Full Marks** | **Attendance Full Marks** | **Total Marks** |
|---|---|---|---|---|
| CIA-1 | 25 | 10 | 05 | 40 |
| CIA-2 | 25 | 10 | 05 | 40 |
| End Semester<br><br>Examination (ESE) | - | - | - | 60 |
| Total | | | | 100 Marks |

## 5. Course Modules

| **Module** | **Topics** | **Hours** |
|---|---|---|
| 1 | **Introduction to Data Visualization for AI**<br><br>- Defining data visualization in the context of AI<br>- The visualization workflow for AI applications | 10 |

| | | |
|---|---|---|
| | (data exploration, model understanding, result communication)<br>- Principles of human perception and cognition relevant to AI insights<br>- Assessing visualization quality for AI model explainability<br>- Ethical considerations in data visualization for AI (avoiding bias, ensuring fairness)<br>- Effective visual communication of AI model performance and limitations | |
| 2 | **Fundamental Chart Types and Data Representation for AI**<br><br>- Data types and their visual representations in AI (categorical, numerical, temporal, spatial)<br>- Core chart types for AI: Histograms, scatter plots, box plots, line charts (focus on applications in regression, classification, clustering)<br>- Visualizing data distributions for model training and evaluation<br>- Effective use of color, labels, and legends for clear communication of AI results<br>- Choosing appropriate chart types based on data characteristics and AI task | 9 |
| 3 | **Data Preprocessing and Exploratory Data Analysis (EDA) for AI**<br><br>- Data cleaning and preprocessing techniques for AI (handling missing values, outliers, noise)<br>- Feature scaling and transformation methods relevant to AI algorithms<br>- Exploratory Data Analysis (EDA) techniques for AI: identifying patterns, relationships, and anomalies<br>- Data visualization techniques for EDA in AI (histograms, scatter plots, box plots, pair plots)<br>- Dimensionality reduction techniques and their visualization (PCA, t-SNE) | 6 |
| 4 | **Visualizing AI Models and Relationships**<br><br>- Visualizing model performance metrics (ROC curves, precision-recall curves, confusion matrices)<br>- Visualizing feature importance for model explainability (feature importance plots, partial dependence plots)<br>- Visualizing decision boundaries for classification models<br>- Visualizing relationships between features and target variables<br>- Techniques for visualizing high-dimensional data in the context of AI (e.g., parallel coordinates, | 7 |

| | | |
|---|---|---|
| | dimensionality reduction visualizations) | |
| 5 | **Data Visualization Tools for AI**<br><br>- Python libraries for data visualization (Matplotlib, Seaborn, Plotly): hands-on practice<br>- Introduction to interactive visualization libraries (Bokeh, Plotly Dash)<br>- Version control with Git for collaborative data visualization projects<br>- Best practices for creating reproducible and shareable visualizations<br>- Creating effective visualizations for reports and presentations | 5 |
| 6 | **Explainable AI (XAI) and Case Studies**<br><br>- Introduction to Explainable AI (XAI) and its importance in AI systems<br>- Techniques for visualizing model explanations (LIME, SHAP values)<br>- Case studies demonstrating the application of data visualization in XAI<br>- Ethical considerations in XAI and responsible AI development<br>- The future of data visualization in AI and related research areas | 5 |

## 6. References

**Textbooks:**


**1.** Claus O. Wilke, Fundamentals of Data Visualization, O'Reilly, 2019

**2.** Andy Kirk, Data Visualization A Handbook for Data Driven Design, Sage Publications, 2016


**Reference Books:**


**1.** Philipp K. Janert, Gnuplot in Action, Understanding Data with Graphs, Manning Publications, 2010

**2.** Alex Campbell, Data Visualization: Ultimate Guide to Data Mining and Visualization, 2020


## 7. Course Outcomes

| ID | Statement | Action Verb | Knowledge Level |
|---|---|---|---|
| AIML 496.1 | Identify and select appropriate | Identify | Remember/Understand |

| | | | |
|---|---|---|---|
| | chart types (histograms, scatter plots, box plots, line charts, etc.) for visualizing different data types (categorical, numerical, temporal, spatial) commonly encountered in AI applications. | | |
| AIML 496.2 | Apply data preprocessing techniques (handling missing values, outliers, noise; feature scaling) to prepare datasets for visualization and AI model training, and interpret the impact of these techniques on visualizations. | Apply | Apply |
| AIML 496.3 | Construct and interpret visualizations of AI model performance metrics (ROC curves, precision-recall curves, confusion matrices) and feature importance (feature importance plots, partial dependence plots) to evaluate model effectiveness and explainability. | Construct | Apply/Analyze |
| AIML 496.4 | Analyze visualizations of high-dimensional data using dimensionality reduction techniques (PCA, t-SNE) and interpret the results in the context of AI model understanding and feature relationships. | Analyze | Analyze |
| AIML 496.5 | Evaluate the ethical implications of data visualization in AI, including bias detection and mitigation strategies, and create visualizations that promote fairness and transparency in AI model outputs. | Evaluate | Evaluate |
| AIML 496.6 | Develop and present interactive data visualizations using Python libraries (Matplotlib, Seaborn, Plotly, Bokeh, Plotly Dash) to effectively communicate insights from AI models and data analysis to both technical and non-technical audiences. | Develop | Create/Evaluate |

## 8. CO-PO Mapping

| CO | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1 | 3 | 2 | 1 | 1 | 1 | - | - | - | - | 1 | - | 1 |
| CO2 | 3 | 3 | 2 | 3 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| CO3 | 3 | 3 | 2 | 3 | 2 | 1 | 1 | 1 | 1 | 2 | 1 | 1 |
| CO4 | 3 | 3 | 2 | 3 | 2 | 1 | 1 | 1 | 1 | 2 | 1 | 1 |
| CO5 | 1 | 1 | 2 | 1 | 1 | 3 | 2 | 3 | 1 | 2 | 1 | 1 |
| CO6 | 2 | 1 | 3 | 1 | 3 | 1 | 1 | 1 | 2 | 3 | 2 | 1 |

## 9. CO-PSO Mapping

| CO | PSO1 | PSO2 | PSO3 |
|---|---|---|---|
| CO1 | 3 | 1 | 1 |
| CO2 | 3 | 2 | 1 |
| CO3 | 3 | 2 | 1 |
| CO4 | 3 | 2 | 1 |
| CO5 | 2 | 1 | 3 |
| CO6 | 2 | 3 | 1 |