



University of Kalyani
Department of Computer Science and Engineering
Kalyani – 741235; West Bengal

**Master of Science in Computer Science
(M.Sc. in CS)**

CURRICULUM
based on
National Education Policy, 2020 (NEP 2020)

With effect from the 2026-2027 academic session

Approved in the PGBOS meeting held on 07.07.2026

Graduate Attributes
(based on section 6.0 of UGC-C&CFPP)

1. **Advanced Disciplinary Knowledge** – Demonstrate comprehensive knowledge of core and emerging areas of computer science, including algorithms, operating systems, databases, computer networks, software engineering, artificial intelligence, and computing systems.
2. **Research Competency** – Identify research problems, formulate research questions, design experiments, and apply appropriate methodologies to conduct independent research with academic and ethical rigor.
3. **Digital and Technological Fluency** – Apply modern programming languages, software development tools, computational platforms, simulation environments, and emerging technologies to design, implement, and evaluate computing solutions.
4. **Critical Thinking and Problem-Solving** – Apply logical, analytical, and algorithmic thinking to effectively model, analyze, and solve complex, real-world computational problems.
5. **Computational and Analytical Skills** – Demonstrate proficiency in algorithm design, data structures, database management, system analysis, and data processing to evaluate and optimize computational solutions.
6. **Interdisciplinary Awareness** – Integrate computer science principles with mathematics, natural sciences, engineering, social sciences, and Indian Knowledge Systems (IKS) to develop innovative, context-aware technologies.
7. **Communication Skills** – Communicate technical solutions, research findings, and project outcomes effectively through written reports, technical documentation, presentations, visualizations, and oral communication.
8. **Teamwork and Leadership:** Collaborate effectively in multidisciplinary environments and demonstrate leadership, project management, and coordination skills in software and computing projects.
9. **Ethical and Social Responsibility:** Apply ethical, legal, professional, and social principles in the design and use of computing technologies, prioritizing privacy, security, intellectual property, and accessibility.
10. **Societal and Community Engagement:** Leverage computer science knowledge to address community challenges, promote digital inclusion, and contribute to sustainable, socially beneficial technological development.
11. **Employability and Professionalism:** Demonstrate industry-ready competencies through practical training, laboratory work, projects, internships, and exposure to contemporary computing methodologies and professional practices.
12. **Lifelong Learning and Innovation:** Exhibit adaptability, creativity, entrepreneurship, and a commitment to continuous learning to respond effectively to evolving computing paradigms and emerging technologies.

Programme Learning Outcomes (PLOs)
(aligned with NHEQF Level 6.5)

- PLO1.** Synthesize and apply core theoretical foundations of computer science—including algorithms, data structures, operating systems, compiler design, databases, computer networks, architecture, theory of computation, and software engineering—to analyze complex computing environments..
- PLO2.** Apply advanced algorithms and computational techniques to analyse and solve complex problems with considerations of correctness, efficiency, scalability, and computational complexity.
- PLO3.** Design, construct, test, and evaluate complex software systems using modern programming paradigms, software engineering methodologies, and automated development tools.
- PLO4.** Design, manage, and optimize database systems and data analytics pipelines using modern data models, statistical methods, query processing, and data processing frameworks.
- PLO5.** Analyze and architect secure computing infrastructure by applying principles of operating systems, distributed architecture, communication networks, and cybersecurity.
- PLO6.** Apply artificial intelligence, machine learning, and soft computing techniques to build intelligent, autonomous, and adaptive systems across diverse domains.
- PLO7.** Utilise emerging computing technologies, including cloud computing, distributed computing, Internet of Things, edge and fog computing, blockchain, cybersecurity, and other contemporary computing paradigms.
- PLO8.** Conduct independent research by formulating problems, reviewing literature, designing methodologies, executing experiments, analyzing results, and disseminating findings through formal technical reports and presentations.
- PLO9.** Demonstrate interdisciplinary competence by applying computer science concepts and computational technologies to problems in science, engineering, social sciences, humanities, and other application domains.
- PLO10.** Identify and address ethical, legal, security, and societal implications of computing, including data privacy, intellectual property rights, accessibility, and responsible technology deployment.
- PLO11.** Integrate perspectives from Indian Knowledge Systems (IKS) and global traditions to evaluate the historical, mathematical, and scientific context of modern computing.
- PLO12.** Demonstrate leadership, effective technical communication, teamwork, project management, and adaptability to support career readiness, entrepreneurship, and continuous learning.

M.Sc. in Computer Science (M.Sc. in CS)

Overview

Integrated Two Academic Sessions :: Four Semester Course :: NEP-2020 based Curriculum Framework *(With effect from the 2026-2027 academic session)*

Eligibility: M.Sc. in Computer Science (Home University – 80% and Other University – 20%) Eligibility for 80% category (Home): 4 years / 3 years B.Sc.Honours (other than NEP)/Major (under NEP 2020) in Computer Science. Eligibility for 20% category (Other): 3 years B.Sc.Honours (other than NEP)/Major (under NEP 2020) in Computer Science Or BCA.

Intake: Thirty (27) [May change from time to time based on University notification]

Academic Calendar: As per the University Academic Calendar.

Admission procedure: As per the University notification.

Course Structure:

- The M.Sc. in Computer Science (M.Sc. CS) programme is at Level 6.5 of NHEQF-2023 and follows the NCrF-2023 and UGC-C&CFPP-2024 frameworks.
- The M.Sc. in CS curriculum consists of 88 credits distributed across 24 courses over 2 academic sessions and 4 semesters.
- Each student is required to complete a total of 84 credits from this curriculum, including 4 credits from an Interdisciplinary Studies (IDS) course offered by various departments of the University of Kalyani, including the Department of Computer Science and Engineering.
- 50 credits are allocated to Major Courses, 16 credits to discipline-specific Minor/Skill-Enhanced courses, 4 credits to the Interdisciplinary Studies (IDS) course, 4 credits to Indian Knowledge Systems (IKS), 10 credits to guided research project/dissertation, and 4 credits to the Grand Viva. The programme also includes 2 non-credit mandatory Value-Added Courses (VACs).
- The PGBOS in Computer Science and Engineering will determine related/equivalent online courses offered through platforms such as SWAYAM, NPTEL, e-PG Pathshala, etc. A total of 20% of the programme credits, particularly from VAC, SEC, IKS, Minor, and Elective courses, may be earned by students through virtual learning mode.
- The department will determine the intake capacity for the IDS course (CS-IDS-451) in every academic session.

Core Objective: The core objective of the M.Sc. in Computer Science programme is to equip learners with strong theoretical foundations, computational thinking, programming expertise, and practical skills to design, analyse, develop, and deploy innovative computing solutions. The programme integrates fundamental and advanced concepts of computer science, including algorithms, artificial intelligence, machine learning, data analytics, software engineering, theoretical computer science, and emerging computing technologies, while fostering research aptitude, problem-solving ability, ethical awareness, and lifelong learning.

Equivalence:

The combined first and second semesters of the 2-year M.Sc. in Computer Science (M.Sc. CS) degree programmes are equivalent to a 1-year Postgraduate Diploma in Computer Science (PG Diploma in Computer Science), subject to fulfilment of the prescribed academic requirements and applicable UGC/NHEQF provisions.

Exit Option:

At the end of the 1st year of the M.Sc. in Computer Science programme, upon successful completion of the prescribed 48 credits, students may opt for an exit award of Postgraduate Diploma in Computer Science (PG Diploma in CS) at Level 6 of NHEQF, as per the applicable UGC/NHEQF-2023 provisions.

Course Credit Structure

Sem	Course Code	Course Title	Course Type	Credit Pattern			Credit Value	Marks Distribution		
				Lecture (L)	Tutorial (T)	Practice (P)		Summative assessment	Continuous assessment	Total Marks
S E M I	CS-MJR-T-401	Advanced Mathematics	Major (Th)	3	1	0	4	80	20	100
	CS-MJR-T-402	Software Engineering	Major (Th)	3	1	0	4	80	20	100
	CS-MJR-T-403	Machine Learning	Major (Th)	3	1	0	4	80	20	100
	CS-MNR-T-404	Statistics and Data Analytics	Minor (Th)	3	1	0	4	80	20	100
	CS-MJR-P-405	Machine Learning Laboratory	Major (Pr)	0	0	2	2	--	--	100
	CS-MNR-P-406	Statistics and Data Analytics Laboratory	Minor (Pr)	0	0	2	2	--	--	100
	CS-VAC-449	449 A: Communicative English and Business Presentation 449 B: Critical Thinking and Problem Solving	Value-added Course	0	0	0	0	--	--	100
	NHEQF Level 6.5: UGC Course Level 400-449				12	4	4	20		
S E M # II	CS-IDS-451	Computer Science in Practice*	Interdisciplinary	2	0	2	4	40 (Th) + 40 (Pr)	10 (Th) + 10 (Pr)	100
	CS-MJR-T-452	Theory of Computation	Major (Th)	3	1	0	4	80	20	100
	CS-MJR-T-453	Data Warehousing and Data Mining	Major (Th)	3	1	0	4	80	20	100
	CS-MJR-T-454	Artificial Intelligence	Major (Th)	3	1	0	4	80	20	100
	CS-MJR-T-455	Soft Computing	Major (Th)	3	1	0	4	80	20	100
	CS-MJR-P-456	Data Warehousing and Data Mining Laboratory	Major (Pr)	0	0	2	2	--	--	100
	CS-MJR-P-457	Artificial Intelligence Laboratory	Major (Pr)	0	0	2	2	--	--	100
	CS-MJR-P-458	Soft Computing Laboratory	Major (Pr)	0	0	2	2	--	--	100
	CS-DISS-459	Dissertation-I	Dissertation	--	--	2	2	--	--	100
	CS-VAC-499	499 A: Scientific Writing with LaTeX 499 B: Open Source Software and Collaborative Development	Value-added Course	0	0	0	0	--	--	100
	NHEQF Level 6.5: UGC Course Level 450-499				14	4	10	28		

* Meant for an interdisciplinary course for students from different programmes of KU, including M.Sc.(CS). M.Sc.(CS) students will opt for an interdisciplinary course from the M.Sc.(CS)/ other PG programmes of KU (mandatory).

After the 1st and 2nd semesters, students may opt for the exit option with a Postgraduate Diploma in CS (as per UGC nomenclature – NHEQF Level 6).

SEM III	CS-IKS-T-501	Indian Knowledge Systems in Computing*	IKS (Th)	3	1	0	4	80	20	100
	CS-MJR-T-502	Compiler Design	Major (Th)	3	1	0	4	80	20	100
	CS-MJR-T-503	Advanced Algorithms	Major (Th)	3	1	0	4	80	20	100
	CS-MNE-T-504	Elective-I	Minor (Elec Th)	3	1	0	4	80	20	100
	CS-MNE-T-505	Elective-II	Minor (Elec Th)	3	1	0	4	80	20	100
	CS-MJR-P-506	Advanced Algorithms Laboratory	Major (Pr)	0	0	2	2	--	--	100
	CS-SEC-P-507	Generative AI Applications Laboratory	SEC (Pr)	0	0	2	2	--	--	100
	CS-DISS-508	Dissertation-II	Dissertation	--	--	4	4	--	--	200
	NHEQF Level 6.5: UGC Course Level 500-549				14	4	8	28		
* Course on IKS (Indian Knowledge System)										
SEM IV	CS-DISS-551	Dissertation-III	Dissertation	--	--	8	8	--	--	400
	CS-GV-552	Grand Viva	Grand Viva	--	--	--	4	--	--	200
	CS-VAC-599	Research Methodology	Major (Th)	0	0	0	0	--	--	100
	NHEQF Level 6.5: UGC Course Level 550-599				0	0	8	12		
Total							88			3000

Credit Distribution – 88 Credits

Major (MJR) courses – 46 credits	Minor [MNR] + Minor (Elec) [MNE] + SEC courses – 6 + 8 + 2 = 16 credits	
Theory + Practical Visualisation Courses: 14 Lecture Credits: 27 Tutorial Credits: 09 Practical Credits: 10	MNR (Mandatory)	MNE [Electives] – 8 Credits & SEC Courses – 2 credits
	Theory + Practical	
	Courses: 2 Lecture Credits: 3 Tutorial Credits: 1 Practical Credits: 2	Courses: 3 Lecture Credits: 6 Tutorial Credits: 2 Practical Credits: 2
Total: 14 courses and 46 credits	Total: 1 course and 6 credits	Total: 3 courses and 10 credits

Interdisciplinary – 4 credits (IDS)	Indian Knowledge System – 4 credits (IKS)	Dissertation + Grand Viva – 12+4 = 16 credits
Course: 1 Lecture Credits: 2 Practical Credits: 2	Course: 1 Lecture Credits: 3 Tutorial Credits: 1	Course: Dissertation (3) + Grand Viva (1) Dissertation Credits: 10 Grand Viva Credits: 4
Total: 1 course and 4 credits	Total: 1 course and 4 credits	Total: 4 courses and 16 credits

Learning Outcomes Mapping: Programme Learning Outcomes (PLOs) to Course Learning Outcomes (CLOs)

Legend: ■ 3 = High ■ 2 = Moderate ■ 1 = Low ● - = No mapping

Semester I

Course	PLO 1	PLO 2	PLO 3	PLO 4	PLO 5	PLO 6	PLO 7	PLO 8	PLO 9	PLO 10	PLO 11	PLO 12
CS-MJR-T-401	■ 3	■ 2	● -	■ 3	■ 3	■ 2	■ 1	● -	● -	■ 1	■ 1	■ 3
CS-MJR-T-402	■ 3	■ 2	■ 3	■ 3	■ 3	■ 1	■ 2	■ 3	■ 2	■ 1	■ 3	■ 3
CS-MJR-T-403	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 2	■ 2	■ 2	■ 3	■ 3	■ 3
CS-MNR-T-404	■ 3	■ 2	■ 3	■ 3	■ 3	■ 2	■ 2	■ 2	■ 2	■ 3	■ 3	■ 3
CS-MJR-P-405	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 2	■ 2	■ 2	■ 3	■ 3	■ 3
CS-MNR-P-406	■ 2	■ 3	■ 3	■ 3	■ 3	■ 2	■ 3	■ 2	■ 2	■ 3	■ 3	■ 3

Semester II

Course	PLO 1	PLO 2	PLO 3	PLO 4	PLO 5	PLO 6	PLO 7	PLO 8	PLO 9	PLO 10	PLO 11	PLO 12
CS-IDS-451	■ 3	■ 2	■ 3	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 3	■ 3	■ 3
CS-MJR-T-452	■ 3	■ 2	■ 1	■ 3	■ 3	■ 1	■ 1	● -	● -	● -	■ 1	■ 3
CS-MJR-T-453	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 2	■ 2	■ 2	■ 3	■ 3	■ 3
CS-MJR-T-454	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 2	■ 2	■ 3	■ 3	■ 3	■ 3
CS-MJR-T-455	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 2	■ 2	■ 2	■ 2	■ 3	■ 3
CS-MJR-P-456	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 2	■ 2	■ 2	■ 3	■ 3	■ 3
CS-MJR-P-457	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 3	■ 3	■ 2	■ 3	■ 3	■ 3
CS-MJR-P-458	■ 3	■ 3	■ 3	■ 3	■ 3	■ 2	■ 2	■ 2	■ 2	■ 3	■ 3	■ 3
CS-DISS-459	■ 3	■ 3	■ 2	■ 3	■ 3	■ 2	■ 3	■ 3	■ 3	■ 2	■ 3	■ 3

Semester III

Course	PLO 1	PLO 2	PLO 3	PLO 4	PLO 5	PLO 6	PLO 7	PLO 8	PLO 9	PLO 10	PLO 11	PLO 12
CS-IKS-T-501	 3	 2	 1	 2	 2	 3	 3	 2	 3	 3	 2	 3
CS-MJR-T-502	 3	 2	 3	 3	 3	 1	 1	 1	 1	 1	 3	 3
CS-MJR-T-503	 3	 3	 3	 3	 3	 2	 1	 2	 1	 2	 3	 3
CS-MNE-T-504	 3	 2	 3	 3	 3	 2	 2	 2	 2	 2	 3	 3
CS-MNE-T-505	 3	 2	 3	 3	 3	 2	 2	 2	 2	 2	 3	 3
CS-MJR-P-506	 3	 3	 3	 3	 3	 2	 2	 2	 1	 2	 3	 3
CS-SEC-P-507	 3	 3	 3	 3	 3	 2	 3	 2	 2	 3	 3	 3
CS-DISS-508	 3	 3	 3	 3	 3	 3	 3	 3	 3	 3	 3	 3

Semester IV

Course	PLO 1	PLO 2	PLO 3	PLO 4	PLO 5	PLO 6	PLO 7	PLO 8	PLO 9	PLO 10	PLO 11	PLO 12
CS-DISS-551	 3	 3	 3	 3	 3	 3	 3	 3	 3	 3	 3	 3
CS-GV-552	 3	 3	 2	 3	 3	 2	 3	 3	 3	 2	 3	 3
CS-VAC-599	 3	 3	 2	 3	 3	 2	 3	 2	 3	 1	 3	 3

FIRST SEMESTER

Course Code: CS-MJR-T-401

Course Category: Major

Course Title: Advanced Mathematics

Prerequisites: None

Course Type: Theory

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks		
3	1	0	4	40	80	20	100	To provide students with a strong mathematical foundation in optimization, probability, algebraic structures, and matrix theory for analyzing and solving complex problems in computer science and related fields.	• Apply optimization techniques such as linear programming, integer programming, transportation, and assignment models to solve real-world decision-making problems. • Analyze combinatorial structures and probability models using counting techniques, mathematical induction, and Bayes' theorem. • Evaluate random variables and probability distributions for modeling uncertainty and stochastic processes. • Demonstrate an understanding of algebraic structures, including groups, rings, and fields, and their applications in computing and cryptography. • Solve systems of linear equations and perform matrix-based computations using determinants, eigenvalues, eigenvectors, and matrix decomposition techniques.

Unit 1: Optimization Techniques: Linear programming, mathematical formulation of linear programming problems, graphical solution method, simplex method, integer programming, transportation problems, assignment problems, applications of optimization techniques. **[8H]**

Unit 2: Advanced Calculus and Multivariable Mathematics: Functions of several variables, limits and continuity, partial derivatives, total derivatives, chain rule, Jacobian and Hessian matrices, Taylor series and multivariable Taylor expansion, maxima and minima of functions of several variables, constrained optimization and Lagrange multipliers, multiple integrals, vector-valued functions, gradient, divergence and curl. **[12 H]**

Unit 3: Algebraic Structures: Groups, subgroups, semigroups, product and quotient of algebraic structures, isomorphism, homomorphism, automorphism, rings, integral domains, fields, applications of group theory in computer science **[10H]**

Unit 4: Linear Algebra and Matrix Theory: Vector spaces and subspaces, linear dependence and independence, basis and dimension, linear transformations, matrix representation of linear transformations, rank and nullity, systems of linear equations, eigenvalues and eigenvectors, characteristic polynomial, diagonalization, orthogonal vectors and orthogonal transformations, inner product spaces, Gram–Schmidt orthogonalization, singular value decomposition (SVD). **[10H]**

Suggested Reading List

1. Hamdy A. Taha. "Operations Research: An Introduction". Pearson.
2. B.S. Grewal. "Higher Engineering Mathematics". Khanna Publishers.
3. J.L. Mott, A. Kandel, and T.P. Baker: Discrete Mathematics for Computer Scientists, Reston, Virginia, 1983
4. R.A. Brualdi: Introductory Combinatorics, North-Holland, New York, 1977.
5. W. Feller: An Introduction to Probability Theory and its Applications (Volume I and II), 3rd ed. John Wiley, New York, 197

FIRST SEMESTER

Course Code: CS-MJR-T-402

Course Category: Major

Course Title: Software Engineering

Prerequisites: None

Course Type: Theory

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To develop an understanding of software engineering principles, requirements, project management, design, quality assurance, and software testing.	- Understand the evolving role, characteristics, and process models of software development. - Gain proficiency in requirement analysis, modeling, and the creation of Software Requirements Specifications (SRS). - Develop skills in project management, estimation, scheduling, and risk management within software projects. - Apply quality assurance practices and software testing strategies, including both black-box and white-box techniques. - Learn to design software architectures and integrate data flow within component-level designs.
3	1	0	4	40	80	20	100		

Unit I: Software Engineering Fundamentals and Process Models: The evolving role of software, software characteristics, changing nature of software, software engineering as a layered technology, software process framework, framework and umbrella activities, software process models, and Capability Maturity Model Integration (CMMI). **[8H]**

Unit II: Software Requirements Engineering: Software requirements analysis, initiating the requirements engineering process, requirements elicitation and analysis, requirements modelling techniques, flow-oriented modelling, Software Requirements Specification (SRS), characteristics of a good SRS, and components of SRS. **[8H]**

Unit III: Software Project and Risk Management: Software project management, estimation in project planning, project scheduling, software risks, risk identification, risk projection, risk refinement, risk mitigation, monitoring and management, and Risk Mitigation, Monitoring and Management (RMMM) plan. **[8H]**

Unit IV: Software Design and Quality Management: Software quality concepts, Software Quality Assurance (SQA), software reviews, process and project metrics, design concepts, architectural design elements, software architecture, data design at architectural and component levels, mapping data flow into software architecture, and component-level design modelling. **[8H]**

Unit V: Software Testing Strategies and Tactics: Software testing fundamentals, strategic approach to software testing, test strategies for conventional software, validation testing, system testing, black-box testing and its techniques, white-box testing and its techniques, basis path testing, and test-case design. **[8H]**

Suggested Reading List

1. R.S. Pressman, Software Engineering: A Practitioner's Approach (7th Edition), McGrawHill, 2009.
2. P. Jalote, An Integrated Approach to Software Engineering (2nd Edition), Narosa Publishing House, 2003.
3. K.K. Aggarwal and Y. Singh, Software Engineering (2nd Edition), New Age International Publishers, 2008.
4. I. Sommerville, Software Engineering (8th edition), Addison Wesley, 2006.
5. D. Bell, Software Engineering for Students (4th Edition), Addison-Wesley, 2005.
6. R. Mall, Fundamentals of Software Engineering (2nd Edition), Prentice-Hall of India, 2004.

FIRST SEMESTER

Course Code: CS-MJR-T-403

Course Category: Major

Course Title: Machine Learning

Prerequisites: Linear Algebra, Probability and Statistics, DSAs, and Programming (Python/C++)

Course Type: Theory

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To equip students with advanced machine learning techniques, deep learning models, reinforcement learning, and explainable AI methodologies for developing intelligent systems and solving complex real-world problems.	• Apply advanced machine learning algorithms for clustering, classification, regression, and predictive analytics. • Analyze the performance of machine learning models using appropriate evaluation metrics, validation strategies, and optimization techniques. • Design and implement deep learning models using neural networks, CNNs, RNNs, LSTMs, GANs, and transfer learning for diverse applications. • Evaluate ensemble learning, probabilistic graphical models, reinforcement learning, and explainable AI techniques for intelligent decision-making. • Develop machine learning solutions for real-world applications involving computer vision, natural language processing, streaming data, healthcare, finance, and cybersecurity.
3	1	0	4	40	80	20	100		

Unit I: Machine Learning Fundamentals and Clustering: Introduction to machine learning, supervised learning, unsupervised learning, reinforcement learning, advanced clustering methods, variants of K-means, hierarchical clustering, BIRCH, DBSCAN, Expectation-Maximization (EM), cluster evaluation techniques, internal and external validation measures, clustering ensemble methods. **[8 H]**

Unit II: Supervised Learning and Model Evaluation: Linear Discriminant Analysis (LDA), Support Vector Machines (SVM), Naïve Bayes classifier, gradient descent, stochastic gradient descent, class evaluation measures, confusion matrix, precision, recall, F1-score, ROC curve, AUC, overfitting, underfitting, bias-variance trade-off, cross-validation, validation strategies, hyperparameter tuning. **[8 H]**

Unit III: Deep Learning and Neural Networks: Artificial neural networks, Linear Threshold Unit (LTU), perceptron, multilayer perceptron (MLP), activation functions, loss functions, optimizers, momentum, Adadelta, RMSProp, Adam optimizer, early stopping, dropout, batch normalization, convolutional neural networks (CNN), recurrent neural networks (RNN), LSTM, GRU, encoder-decoder networks, transfer learning, autoencoders, generative adversarial networks (GANs). **[8 H]**

Unit IV: Advanced Machine Learning Models: Word embedding, Continuous Bag of Words (CBOW), Skip-gram, GloVe, ELMo, ensemble learning, bagging, committee machines, stacking, boosting, gradient boosting, undirected graphical models, Markov chains, Hidden Markov Models (HMM), random walk Monte Carlo, variable elimination, belief propagation. **[8 H]**

Unit V: Reinforcement Learning, Explainable AI and Applications: Introduction to reinforcement learning, sampling-based techniques, Q-learning, policy learning, Explainable AI (XAI), LIME, SHAP, learning from online streaming data, model deployment concepts, machine learning applications in healthcare, finance, cybersecurity, natural language processing, computer vision, recommender systems, recent trends in machine learning. **[8 H]**

Suggested Reading List

1. Tom Mitchell, Machine Learning, McGraw-Hill, 1997.
2. Duda, Richard O., Peter E. Hart, and David G. Stork. Pattern classification. JohnWiley & Sons, 2012.
3. Alpaydin, Ethem. Introduction to machine learning. MIT Press,2009.
4. Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. The elements of statistical learning. Vol.1. No. 10. New York: Springer series in statistics,2001.
5. Kevin P. Murphy. Machine Learning: A Probabilistic Perspective. MIT Press, 2012.
6. Ian Goodfellow, Yoshua Bengio and Aaron Courville. Deep Learning. MIT Press, 2016.

FIRST SEMESTER

Course Code: CS-MNR-T-404

Course Category: Minor

Course Title: Statistics and Data Analytics

Prerequisites: Programming , and DBMS.

Course Type: Theory

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks		
3	1	0	4	40	80	20	100	To equip students with fundamental concepts and analytical techniques in data analytics, probability, and statistical modeling for data-driven decision making.	- Understand data analytics processes, including data collection, preprocessing, and ETL techniques. - Apply descriptive statistics and probability concepts to summarize and analyze data. - Analyze probability distributions and sampling methods for real-world data modeling. - Perform statistical inference using hypothesis testing, confidence intervals, and ANOVA techniques. - Develop and evaluate predictive models using regression, classification, and dimensionality reduction methods.

Unit 1: Introduction to Data Analytics & Data Preparation: Introduction to data analytics and data science, types of data (structured, unstructured, categorical, numerical), data collection methods and sources, data blending and preparation, data preprocessing techniques, ETL (Extract, Transform, Load) processes, data mining concepts, data science project life cycle. **[8H]**

Unit 2: Descriptive Statistics & Probability Fundamentals: Data summaries and descriptive statistics, measures of central tendency and dispersion, variance, covariance and correlation, basic probability concepts, probability calculus, conditional probability, Bayes' theorem, mathematical expectation. **[8H]**

Unit 3: Probability Distributions & Sampling: Uniform distribution, normal distribution, exponential distribution, binomial distribution, chi-square distribution, Student's t-distribution, F-distribution, central limit theorem, sampling techniques, random sample generation, measurement, errors in data. **[8H]**

Unit 4: Statistical Inference & Hypothesis Testing: Hypothesis testing, A/B testing, confidence intervals, p-values, t-test, Z-test, F-test, ANOVA, post hoc analysis, sign test, Wilcoxon signed-rank test, Spearman's rank-order correlation. **[8H]**

Unit 5: Advanced Analytics & Modeling Techniques: Regression, bias-variance tradeoff, linear regression, multivariate regression, subset selection, shrinkage methods, principal component regression, partial least squares, logistic regression, linear discriminant analysis, principal component analysis, multifactor dimensionality reduction, Gaussian mixture models, expectation maximization, time series analysis, ARMA models. **[8H]**

Suggested Reading List

- Walpole, Ronald E., Raymond H. Myers, Sharon L. Myers, and Keying Ye. *Probability and Statistics for Engineers and Scientists*. Pearson, 2012.
- Cady, Field. *The data science handbook*. John Wiley & Sons, 2017.
- Peter Bruce and Andrew Bruce, *Practical Statistics for Data Science*, O'Reilly, 2017.
- Moreira, João, Andre Carvalho, and Tomás Horvath. *A general introduction to data analytics*. John Wiley & Sons, 2018.
- J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann/Elsevier India, 2001.
- Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. No. 10. New York: Springer series in statistics, 2001.
- Alpaydin, Ethem. *Introduction to machine learning*. MIT press, 2009.
- D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*, MIT Press, 2001.

FIRST SEMESTER

Course Code: CS-MJR-P-405

Course Title: Machine Learning Laboratory

Prerequisites: Python, Statistics and Data Analytics.

Course Category: Major

Course Type: Practical

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To provide practical experience in implementing, evaluating, and deploying machine learning algorithms using modern software tools for solving real-world data analytics and predictive modeling problems.	- Implement data preprocessing, visualization, and feature engineering techniques for machine learning applications. - Develop supervised and unsupervised machine learning models using standard algorithms and frameworks. - Evaluate machine learning models using appropriate performance metrics and validation techniques. - Apply deep learning techniques for classification and prediction using modern libraries such as TensorFlow and Keras. - Design and implement an end-to-end machine learning solution for a real-world application using industry-standard tools.
0	0	2	2	20	80	20	100		

Laboratory Syllabus

1. Introduction to the Python programming environment for Machine Learning (NumPy, Pandas, Matplotlib, Seaborn, Scikit-learn).
2. Data preprocessing, data cleaning, handling missing values, feature scaling, and feature encoding.
3. Exploratory Data Analysis (EDA) and data visualization techniques.
4. Implementation of Linear Regression and Multiple Linear Regression.
5. Implementation of Logistic Regression for classification problems.
6. Implementation of K-Nearest Neighbors (KNN) classifier.
7. Implementation of Naïve Bayes classifier.
8. Implementation of Decision Tree and Random Forest algorithms.
9. Implementation of Support Vector Machine (SVM) for classification.
10. Implementation of K-Means and Hierarchical Clustering algorithms.
11. Dimensionality reduction using Principal Component Analysis (PCA).
12. Implementation of Artificial Neural Networks (ANN) using TensorFlow/Keras.
13. Implementation of Convolutional Neural Networks (CNN) for image classification.
14. Performance evaluation using Confusion Matrix, Accuracy, Precision, Recall, F1-Score, ROC Curve, and AUC.
15. Hyperparameter tuning using Cross-Validation and Grid Search.
16. Mini-project involving the design, implementation, and evaluation of a Machine Learning model on a real-world dataset.

Suggested Software/Tools : Python, Jupyter Notebook, Google Colab, NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow, Keras, OpenCV (optional), Kaggle Datasets / UCI Machine Learning Repository

FIRST SEMESTER

Course Code:CS-MNR-P-406

Minor

Course Category:

Course Title: Statistics and Data Analytics Laboratory

Prerequisites:Probability and Statistics, Python/R, DBMS , and ML concepts.

Course Type: Practical

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To provide practical experience in statistical analysis, data visualization, predictive modeling, and data analytics using modern statistical and machine learning tools.	- Perform statistical analysis and exploratory data analysis using Python/R and standard analytical libraries. - Apply data preprocessing, visualization, and feature engineering techniques to real-world datasets. - Develop statistical and predictive models using regression, classification, and clustering algorithms. - Evaluate analytical models using appropriate statistical measures and machine learning performance metrics. - Design and implement a complete data analytics solution for solving practical problems using statistical and computational techniques.
0	0	2	2	20	80	20	100		

Laboratory Syllabus

1. Introduction to Python/R for statistical computing and data analytics (NumPy, Pandas, Matplotlib, SciPy, Scikit-learn).
2. Data collection, data import, data cleaning, and preprocessing.
3. Data visualisation using bar charts, histograms, pie charts, box plots, scatter plots, and heatmaps.
4. Computation of descriptive statistics: mean, median, mode, variance, standard deviation, skewness, and kurtosis.
5. Probability distributions: Binomial, Poisson, and Normal distributions.
6. Correlation analysis and covariance.
7. Simple and multiple linear regression analysis.
8. Hypothesis testing using Z-test, t-test, Chi-square test, and ANOVA.
9. Exploratory Data Analysis (EDA) using real-world datasets.
10. Data transformation, normalisation, feature scaling, and handling missing values.
11. Principal Component Analysis (PCA) for dimensionality reduction.
12. Classification using Logistic Regression, Naïve Bayes, and Decision Trees.
13. Clustering using K-Means and Hierarchical Clustering.
14. Model evaluation using Confusion Matrix, Accuracy, Precision, Recall, F1-Score, ROC Curve, and AUC.
15. Time series visualisation and basic forecasting techniques.
16. Mini-project involving statistical analysis and predictive analytics on a real-world dataset.

Suggested Software/Tools: Python, Jupyter Notebook, Google Colab, NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow, Keras, OpenCV (optional), Kaggle Datasets / UCI Machine Learning Repository

FIRST SEMESTER

Course Code:CS-VAC-449A

Course Title: Communicative English and Business Presentation

Prerequisites:None.

Course Category: VAC

Course Type: Theory+Practice

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To develop effective English communication, vocabulary, writing, comprehension, and professional presentation skills for academic, workplace, and business communication.	• Apply vocabulary-building techniques using roots, prefixes, suffixes, synonyms, antonyms, and abbreviations. • Construct clear, coherent, grammatically correct, and well-organized written communication. • Identify and correct common grammatical and stylistic errors in writing. • Develop concise, logical, and effective descriptive and professional writing. • Deliver effective business presentations using appropriate verbal, visual, and non-verbal communication techniques.
0	0	0	0	20			100		

Unit 1: Vocabulary Building: Concept of word formation, root words and their origins from foreign languages and their use in English, prefixes and suffixes from foreign languages and their use in forming derivatives, synonyms, antonyms, standard abbreviations, contextual vocabulary, and strategies for vocabulary enrichment. **[4H]**

Unit 2: Basic Writing Skills: Sentence structures, simple, compound, and complex sentences, use of phrases and clauses in sentences, importance and application of proper punctuation, creating coherence and cohesion, principles of paragraph organization, logical sequencing of ideas, techniques for precise and concise writing. **[4H]**

Unit 3: Identifying and Correcting Common Errors in Writing: Subject–verb agreement, noun–pronoun agreement, misplaced and dangling modifiers, appropriate use of articles, prepositions, redundant expressions, clichés, common grammatical errors, sentence correction, and editing practices. **[4H]**

Unit 4: Nature and Style of Sensible Writing: Principles of effective writing, describing, defining, classifying, providing examples and evidence, developing logical arguments, writing effective introductions and conclusions, organizing ideas, maintaining clarity and coherence, and techniques for professional and formal writing. **[4H]**

Unit 5: Writing Practices and Business Presentation: Reading comprehension, précis writing, essay writing, principles of effective oral communication, structure and organization of business presentations, preparation of presentation content, use of visual aids and presentation slides, verbal and non-verbal communication, delivery techniques, audience engagement, handling questions and feedback, and professional presentation etiquette. **[4H]**

Suggested Reading List

1. Wren, P. C. and H. Martin. *High School English Grammar and Composition*. S. Chand Publishing.
2. Michael Swan. *Practical English Usage*. Oxford University Press.
3. Norman Lewis. *Word Power Made Easy*. Penguin Books.
4. John Seely. *The Oxford Guide to Effective Writing and Speaking*. Oxford University Press.
5. Raymond Murphy. *English Grammar in Use*. Cambridge University Press.
6. David Gruber. *Business English: A Complete Guide to Professional Communication*. Pearson.
7. John Hughes and Jon Naunton. *Business Result: Intermediate*. Oxford University Press.

FIRST SEMESTER

Course Code:CS-VAC-449B

Course Title: Critical Thinking and Problem Solving

Prerequisites:None.

Course Category: VAC
Course Type: Theory+Practice

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To develop effective English communication, vocabulary, writing, comprehension, and professional presentation skills for academic, workplace, and business communication.	• Apply critical thinking techniques to analyze information, assumptions, arguments, and evidence. • Use logical reasoning and structured approaches to identify and solve problems. • Evaluate alternatives and make informed decisions in complex situations. • Apply creative thinking techniques to generate innovative solutions to practical problems. • Communicate problem-solving approaches and justify decisions using clear and logical reasoning.
0	0	0	0	20			100		

Unit 1: Fundamentals of Critical Thinking: Meaning and importance of critical thinking, characteristics of critical thinkers, observation and interpretation, facts and opinions, assumptions, biases and stereotypes, questioning techniques, evidence and reasoning. **[4H]**

Unit 2: Logical Reasoning and Argument Analysis: Basic principles of logical reasoning, deductive and inductive reasoning, arguments and conclusions, identifying premises, validity and soundness, logical fallacies, cause-and-effect reasoning, analogy and inference. **[4H]**

Unit 3: Problem Identification and Analysis: Nature and types of problems, problem identification, problem definition, problem decomposition, root-cause analysis, 5W1H technique, brainstorming, mind mapping, problem trees, and systematic problem analysis. **[4H]**

Unit 4: Creative Thinking and Decision Making: Creative thinking, divergent and convergent thinking, lateral thinking, brainstorming techniques, SCAMPER technique, evaluating alternatives, decision-making processes, decision matrices, risk and uncertainty, and avoiding common decision-making biases. **[4H]**

Unit 5: Practical Problem Solving: Structured problem-solving approaches, case-based problem solving, real-world decision-making scenarios, ethical and professional problem solving, teamwork and collaborative problem solving, presenting solutions, evaluating outcomes, and lessons learned. **[4H]**

Suggested Reading List

1. Edward de Bono. Six Thinking Hats. Penguin Books.
2. Diane F. Halpern. Thought and Knowledge: An Introduction to Critical Thinking. Psychology Press.
3. Richard Paul and Linda Elder. Critical Thinking: Tools for Taking Charge of Your Learning and Your Life. Pearson.
4. Michael Kallet. Think Smarter: Critical Thinking to Improve Problem-Solving and Decision-Making Skills. Wiley.
5. Tony Proctor. Creative Problem Solving for Managers. Routledge.
6. Scott G. Isaksen, K. Brian Dorval, and Donald J. Treffinger. Creative Approaches to Problem Solving. SAGE Publications.

SECOND SEMESTER

Course Code: CS-IDS-451

Course Category: Major

Course Title: Computer Science in Practice*

Prerequisites: None

Course Type: Theory + Practice

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks		
3	1	0	4	50	40+40	10+10	100	To provide students with a comprehensive understanding of computer organisation, processor design, memory systems, pipelining, and parallel processing techniques for analysing and improving computer system performance.	- Explain the fundamental concepts of computer organisation, instruction set architecture, and arithmetic operations used in modern computer systems. - Analyse CPU design components, arithmetic algorithms, and control unit implementations, including hardwired and microprogrammed control. - Evaluate pipelined processor architectures, identify pipeline hazards, and apply optimisation techniques to improve performance. - Assess memory hierarchy, cache, and virtual memory management techniques, and CPU-memory interfacing mechanisms. - Examine advanced computer architectures and parallel processing techniques for enhancing computational efficiency and system performance.

Unit 1: Computer Organisation and Architecture: Introduction to computer organisation and architecture, basic components of computer architecture, stored program concept, instruction formats, addressing modes, instruction set architecture, commonly used number systems, fixed-point arithmetic, floating-point arithmetic **[10H]**.

Unit 2: CPU Design and Control Unit: CPU organisation, ripple carry adder design, carry look-ahead adder design, multiplication algorithms, division algorithms, hardwired control unit, microprogrammed control unit, introduction to RISC architecture, introduction to CISC architecture. **[10H]**

Unit 3: Pipelining and Processor Performance: Introduction to pipelining, instruction pipelining, arithmetic pipelining, pipeline hazards, structural hazards, data hazards, control hazards, hazard handling techniques, pipeline optimisation techniques, vector processors, array processors, quantitative techniques in computer design, performance measurement, and reporting. **[12H]**

Unit 4: Memory Organisation and Input/Output Systems: Memory hierarchy, memory organisation, CPU-memory interfacing, cache memory organisation, cache mapping techniques, virtual memory organisation, memory management techniques, memory replacement policies, input/output organisation and interfacing. **[10H]**

Unit 5: Advanced Architectures and Parallel Processing: Advanced processor architectures, memory management issues, performance enhancement techniques, introduction to parallel processing, parallel architectures, shared memory systems, distributed memory systems, applications of parallel computing. **[8H]**

Suggested Reading List

- William Stallings, "Computer Organization and Architecture", 7th Edition, Prentice Hall, 2013.
- Hennessy J and Patterson D, "Computer Architecture - A Quantitative Approach", 5th Edition, Morgan Kaufmann, 2011.
- M. Morris Mano, Michael D. Ciletti, "Digital Design", 4th Edition, Pearson Education, 2011.
- "Computer Architecture and Parallel Processing" Kai Hwang and A. Briggs, International Edition, McGraw-Hill.
- John P Hayes, "Computer Architecture and Organization", 2nd Edition, McGraw-Hill, 2002.
- Advanced Computer Architectures, Dezso Sima, Terence Fountain, Peter Kacsuk, Pearson.

SECOND SEMESTER

Course Code: CS-MJR-T-452

Course Category: Major

Course Title: Theory of Computation

Prerequisites: Discrete Mathematics.

Course Type: Theory

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks		
3	1	0	4	40	80	20	100	To develop a formal understanding of automata, formal languages, computational models, computability, and the fundamental limitations of computation.	- Construct and analyze finite automata and regular expressions for recognizing regular languages. - Apply context-free grammars and pushdown automata to model and analyze context-free languages. - Analyze properties of regular and context-free languages using closure properties and pumping lemmas. - Design and analyze Turing machines and other computational models for solving formal problems. - Determine the decidability, recognizability, and computational limitations of problems using formal techniques.

Unit I: Finite Automata and Regular Languages: Alphabets, strings, languages, deterministic finite automata (DFA), nondeterministic finite automata (NFA), ϵ -NFA, DFA-NFA equivalence, DFA minimization, regular expressions, regular grammars, equivalence of finite automata and regular expressions, closure properties of regular languages, applications of finite automata. **[6 H]**

Unit II: Properties of Regular Languages: Pumping lemma for regular languages, applications of the pumping lemma, proving languages to be non-regular, closure properties, decision properties of regular languages, Myhill–Nerode theorem, distinguishability and equivalence relations, minimization using Myhill–Nerode theorem. **[6 H]**

Unit III: Context-Free Languages and Pushdown Automata: Context-free grammars (CFG), derivations and parse trees, ambiguity in CFGs, normal forms, Chomsky Normal Form, Greibach Normal Form, pushdown automata (PDA), deterministic and nondeterministic PDA, equivalence of CFGs and PDA, closure properties of context-free languages, pumping lemma for CFLs. **[8 H]**

Unit IV: Turing Machines and Computational Models: Basic structure and operation of Turing machines, formal definition, transition functions, configurations and instantaneous descriptions, language recognition and generation, designing Turing machines, variants of Turing machines, multitape Turing machines, nondeterministic Turing machines, universal Turing machine, Church–Turing thesis, computational equivalence of models. **[8 H]**

Unit V: Computability and Decidability: Decidable and recognizable languages, decidable problems, closure properties of decidable and recognizable languages, reductions, undecidable problems, halting problem, acceptance problem, Post Correspondence Problem, Rice’s theorem, applications of undecidability, Computational complexity and the classes P and NP. **[8 H]**.

Suggested Reading List

1. Michael Sipser. Introduction to the Theory of Computation. Cengage Learning.
2. John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. Introduction to Automata Theory, Languages, and Computation. Pearson.
3. Peter Linz. An Introduction to Formal Languages and Automata. Jones & Bartlett Learning.
4. Dexter C. Kozen. Automata and Computability. Springer.
5. John C. Martin. Introduction to Languages and the Theory of Computation. McGraw-Hill.
6. Harry R. Lewis and Christos H. Papadimitriou. Elements of the Theory of Computation. Pearson.

SECOND SEMESTER

Course Code: CS-MJR-T-453

Course Category: Major

Course Title: Data Warehousing and Data Mining

Prerequisites: DBMS and ML.

Course Type: Theory

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To develop an understanding of data warehousing, business intelligence, and data mining techniques for analyzing large-scale datasets and extracting meaningful insights.	- Design and implement data warehouses using appropriate schemas (Star, Snowflake) and perform ETL processes effectively. - Apply OLAP operations and multidimensional analysis for business intelligence and decision support. - Utilize data mining techniques (classification, clustering, association rules) to discover hidden patterns in large datasets. - Preprocess and prepare data using cleaning, transformation, and reduction techniques for analytical tasks. - Analyze and mine complex data such as spatial, text, and web data for real-world business applications.
3	1	0	4	40	80	20	100		

Unit 1: Data Warehousing and OLAP Data warehousing components, building a data warehouse, Data Warehouse Architecture, DBMS schemas for decision support (star, snowflake, fact constellation), Data Extraction, Cleanup, and Transformation Tools, Metadata, reporting, query tools and applications, Online Analytical Processing (OLAP), OLAP operations (roll-up, drill-down, slice, dice, pivot), multidimensional data analysis. **[8H]**

Unit 2: Data Preprocessing and Mining Overview Data mining functionalities, Data preprocessing — data cleaning, data integration and transformation, data reduction, data discretization and concept hierarchy generation; Architecture of a typical data mining system; Classification of data mining systems; KDD process. **[7H]**

Unit 3: Association Rule Mining Efficient and scalable frequent itemset mining methods (Apriori, FP-Growth); Mining various kinds of association rules; Association mining to correlation analysis; Constraint-based association mining; Applications of association mining. **[7H]**

Unit 4: Cluster Analysis Types of data in cluster analysis; Categorization of major clustering methods; Partitioning methods (K-means, K-medoids); Hierarchical methods ; Density-based methods; Grid-based methods; Model-based clustering; Clustering high-dimensional data; Outlier analysis. **[9H]**

Unit 5: Classification, Prediction and Advanced Mining Issues regarding classification and prediction; Nearest Neighbor classification; Decision Tree induction (ID3, C4.5); Bayesian classification; Rule-based classification; Support Vector Machines; Associative classification; Accuracy and error measures. **[9H]**

Suggested Reading List

1. Reema Thareja, “Data Warehousing”, Oxford University Press.
2. Jiawei Han and Micheline Kamber, “Data Mining Concepts & Techniques”, Elsevier Pub.
3. Margret H. Dunham “Data Mining: Introductory and Advanced topics” Pearson Education.
4. Paulraj Ponniah, “Data Warehousing Fundamentals”, John Wiley & Sons, Inc.
5. Vikram Pudi, P. Radha Krishana “Data Mining”, Oxford University press.

SECOND SEMESTER

Course Code: CS-MJR-T-454

Course Category: Major

Course Title: Artificial Intelligence

Prerequisites: Data Structures and Algorithms

Course Type: Theory

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks		
3	1	0	4	40	80	20	100	To introduce fundamental AI concepts, including intelligent agents, search strategies, knowledge representation, planning, and reasoning under uncertainty.	• Explain the foundations and history of Artificial Intelligence, as well as the science of agent design. • Illustrate the use of problem-solving techniques, such as the various search methods, games, and constraint satisfaction problems. • Demonstrate AI's use of knowledge representation, through logic agents and first-order logic to address AI problems. • Design simple software to experiment with various AI learning concepts and analyze results. • Build self-learning and research skills to be able to tackle a topic of interest on his/her own or as part of a team.

Unit I: Introduction to Artificial Intelligence: Intelligent agents, agents and environments, rational behavior, nature of environments, structure of agents, problem-solving agents, problem formulation, example problems, searching for solutions, uninformed search strategies, and avoiding repeated states. **[10H]**

Unit II: Searching Techniques: Informed search strategies, heuristic functions, local search algorithms, optimization problems, local search in continuous spaces, online search agents and unknown environments, constraint satisfaction problems (CSP), backtracking search, local search for CSP, problem structures, adversarial search, game playing, optimal decisions, and alpha-beta pruning. **[10H]**

Unit III: Knowledge Representation: First-order logic, syntax and semantics, knowledge representation using first-order logic, knowledge engineering, inference in first-order logic, propositional versus first-order logic, unification and lifting, forward chaining, backward chaining, and resolution. **[10H]**

Unit IV: Planning and Reasoning with Uncertainty: Classical planning, STRIPS representation, partial-order planning, hierarchical task network planning, state-space and plan-space search, rule-based expert systems, expert system architecture, probability review, Bayesian reasoning, Bayesian networks, Dempster-Shafer theory, certainty factors, and applications of uncertainty-based reasoning in expert systems. **[10H]**

Suggested Reading List

1. Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson.
2. Elaine Rich, Kevin Knight, and Shivashankar B. Nair. *Artificial Intelligence*. McGraw-Hill.
3. Nils J. Nilsson. *Artificial Intelligence: A New Synthesis*. Morgan Kaufmann.
4. Patrick H. Winston. *Artificial Intelligence*. Addison-Wesley.
5. George F. Luger. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Pearson.

SECOND SEMESTER

Course Code: CS-MJR-T-455
Course Title: Soft Computing

Prerequisites: Algorithms basics

Course Category: Major
Course Type: Theory

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks		
3	1	0	4	40	80	20	100	To provide students with a comprehensive understanding of soft computing techniques, including neural networks, fuzzy logic, genetic algorithms, and neuro-fuzzy systems, for solving complex real-world problems involving uncertainty, optimization, and intelligent decision-making. .	● Explain the principles of soft computing and compare neural networks, fuzzy logic, and evolutionary computing techniques. ● Apply genetic algorithms and optimization techniques to solve complex computational and engineering problems. ● Design and implement artificial neural network models for classification, prediction, and pattern recognition tasks. ● Develop fuzzy logic and neuro-fuzzy systems for reasoning, control, and decision-making under uncertainty. ● Evaluate the applicability of soft computing techniques in image processing, natural language processing, robotics, bioinformatics, financial forecasting, and other intelligent systems..

Unit I: Introduction to Soft Computing and Neural Networks: Introduction to soft computing, characteristics of soft computing, biological neural networks, artificial neural networks, neuron models, network architectures, learning paradigms, introduction to fuzzy sets, fuzzy logic systems, applications of soft computing. **[8 H]**

Unit II: Genetic Algorithms and Evolutionary Computing: Introduction to genetic algorithms, chromosome representation, fitness function, genetic operators, selection methods, crossover, mutation, genetic algorithm parameters, theoretical foundations of genetic algorithms, optimization using genetic algorithms, implementation issues, applications of genetic algorithms. **[8 H]**

Unit III: Artificial Neural Networks: Perceptron, multilayer perceptron, feedforward neural networks, backpropagation learning algorithm, recurrent neural networks, self-organizing maps, radial basis function networks, neural network training, neural networks for classification and prediction, applications of neural networks in control systems and pattern recognition. **[8 H]**

Unit IV: Fuzzy Logic and Neuro-Fuzzy Systems: Fuzzy sets, membership functions, fuzzy operations, fuzzy relations, fuzzy reasoning, fuzzy inference systems, Mamdani and Sugeno models, fuzzy control systems, fuzzy clustering, neuro-fuzzy modeling, adaptive neuro-fuzzy inference systems (ANFIS), neuro-fuzzy control, applications of fuzzy systems. **[8 H]**

Unit V: Applications of Soft Computing: Pattern recognition, image processing, natural language processing, information retrieval systems, robotics and intelligent sensors, biological sequence alignment, drug discovery and design, financial and share market analysis, optimization problems, intelligent decision support systems, recent trends in soft computing. **[8 H]**

Suggested Reading List

1. M. Mitchell: An Introduction to Genetic Algorithms, Prentice-Hall.
2. J.S.R.Jang, C.T.Sun and E.Mizutani: Neuro-Fuzzy and Soft Computing, PHI, Pearson Education.
3. Timothy J.Ross: Fuzzy Logic with Engineering Applications, McGraw-Hill.
4. Davis E.Goldberg: Genetic Algorithms: Search, Optimization and Machine Learning, AddisonWesley.
5. S. Rajasekaran and G.A.V.Pai: Neural Networks, Fuzzy Logic and Genetic Algorithms, PHI.
6. D. E. Goldberg: Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley.

SECOND SEMESTER

Course Code:CS-MJR-P-456

Course Title: Data Warehousing and Data Mining Laboratory

Prerequisites: ML concepts.

Course Category: Major

Course Type: Practical

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To develop theoretical and practical knowledge of data warehousing, data mining, and business analytics for intelligent decision-making and real-world applications.	• Explain the architecture and design principles of data warehouses and data mining systems. • Apply data preprocessing, ETL, and OLAP techniques for decision-support applications. • Apply association rule mining, clustering, and classification techniques to discover patterns. • Evaluate data mining models using appropriate performance metrics. • Design data warehouse and business analytics solutions for real-world applications.
0	0	2	2	30	80	20	100		

Laboratory Syllabus

1. **Design and Create a Data Warehouse :** Implement star schema and snowflake schema using a sample dataset.
2. **ETL (Extract, Transform, Load) Implementation:** Use ETL tools (e.g., Talend, Pentaho, SSIS) to load cleaned and transformed data into the warehouse.
3. **Metadata Creation and Management:** Store and query metadata in the data warehouse environment.
4. **OLAP Cube Design and Querying:** Create and query OLAP cubes for multidimensional data analysis.
5. **Data Preprocessing:** Perform data cleaning, integration, transformation, and reduction using Python or R.
6. **Association Rule Mining:** Apply Apriori and FP-Growth algorithms on transactional datasets to discover frequent itemsets.
7. **Correlation and Constraint-Based Association Mining:** Analyze strong and weak correlations in datasets using association mining techniques.
8. **Clustering Analysis:** Implement k-Means, hierarchical clustering, and DBSCAN using sample datasets.
9. **Classification Models:** Implement Decision Tree, Naïve Bayes, and k-Nearest Neighbor classifiers.
10. **Prediction and Model Evaluation:** Build regression models and evaluate accuracy using confusion matrix, precision, recall, and F1-score.
11. **Ensemble Methods in Classification:** Apply Bagging, Boosting, and Random Forest methods to improve prediction accuracy.
12. **Text Mining:** Extract keywords, perform sentiment analysis, and generate word clouds from text datasets.
13. **Web Mining:** Scrape and analyze web data for trend detection.
14. **Spatial and Multimedia Data Mining:** Use GIS datasets for spatial analysis; extract features from images or audio datasets.
15. **Mini Project:** End-to-end data warehousing and mining project integrating ETL, OLAP, and advanced analytics.

Suggested Software/Tools: Python, Jupyter Notebook, Google Colab, NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow, Keras, OpenCV (optional), Kaggle Datasets / UCI Machine Learning Repository

SECOND SEMESTER

Course Code:CS-MJR-P-457

Course Title: Artificial Intelligence Laboratory

Prerequisites: ML concepts.

Course Category: Major

Course Type: Practical

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To develop practical skills in implementing intelligent agents, search, reasoning, constraint satisfaction, and optimization techniques using Python.	- Implement intelligent agents and search algorithms using Python. - Solve constraint satisfaction and optimization problems using suitable AI techniques. - Apply first-order logic and inference methods to solve reasoning problems. - Demonstrate AI concepts through simulations, puzzles, and game environments.
0	0	2	2	30	80	20	100		

Laboratory Syllabus

- Introduction to Intelligent Agents:** Simulate different types of agents (simple reflex, model-based, goal-based, utility-based) using Python.
Tools: Python, Jupyter Notebook, matplotlib for visualization.
- Uninformed Search Techniques:** Implement **Breadth-First Search (BFS)**, **Depth-First Search (DFS)**, and **Uniform Cost Search** for maze/pathfinding problems.
Tools: Python, networkx or custom graph classes.
- Informed Search and Heuristics:** Implement **A*** and **Greedy Best-First Search** for solving puzzles (e.g., 8-puzzle, pathfinding).
Tools: Python, heapq, numpy.
- Local Search Algorithms:** Solve optimization problems using **Hill Climbing**, **Simulated Annealing**, and **Genetic Algorithms**.
Tools: Python, random, matplotlib.
- Online Search and Unknown Environments:** Build a simple grid world where an agent navigates without prior map knowledge.
Tools: Python, pygame (optional for UI).
- Constraint Satisfaction Problems (CSP):** Solve classic CSPs like **Sudoku**, **Map Coloring**, or **N-Queens** using **Backtracking** and **Forward Checking**.
Tools: Python, Jupyter Notebook.
- Adversarial Search (Game Playing):** Create a **Tic-Tac-Toe** or **Connect Four** agent using **Minimax** and **Alpha-Beta Pruning**.
Tools: Python, tkinter (optional GUI).
- First-Order Logic Representation:** Encode facts and rules in **First Order Logic (FOL)** and perform **Forward and Backward Chaining**.
Tools: Prolog or Python-based logic engine (e.g., pyDatalog).
- Resolution and Unification:** Demonstrate **Unification** of FOL statements and apply **Resolution** for theorem proving.
Tools: Python, custom logic rule engine.

Lab Requirements

Software: Python (Anaconda), Jupyter Notebook, scikit-learn, numpy, matplotlib, pygame (optional)

Datasets: UCI repository (Iris, Wine, Titanic), custom puzzles and game environments.

Hardware: Standard desktop/laptop with 8GB RAM

SECOND SEMESTER

Course Code: CS-MJR-P-458

Course Title: Soft Computing Laboratory

Prerequisites: Python, Data Structures, Probability and Statistics, Soft Computing.

Course Category: Major

Course Type: Practical

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To provide hands-on experience in implementing neural networks, fuzzy logic, genetic algorithms, and neuro-fuzzy systems for solving real-world optimization, classification, and intelligent decision-making problems.	- Implement artificial neural networks, fuzzy logic systems, and genetic algorithms using modern software tools. - Develop intelligent systems for classification, prediction, optimization, and control applications. - Analyze the performance of different soft computing techniques using appropriate evaluation metrics. - Apply soft computing methods to solve practical problems in image processing, pattern recognition, and optimization. - Design and demonstrate a complete soft computing solution for a real-world application through a laboratory mini-project.
0	0	2	2	30	80	20	100		

Laboratory Syllabus

1. Introduction to Python libraries for soft computing (NumPy, Pandas, Scikit-learn, TensorFlow/Keras).
2. Implementation of basic Artificial Neural Network (ANN) models.
3. Perceptron learning algorithm for binary classification.
4. Multilayer Perceptron (MLP) using backpropagation algorithm.
5. Implementation of activation functions and optimization algorithms.
6. Design and implementation of Convolutional Neural Networks (CNN) for image classification.
7. Implementation of Recurrent Neural Networks (RNN) and LSTM for sequential data analysis.
8. Design and implementation of Genetic Algorithms for optimization problems.
9. Implementation of selection, crossover, and mutation operators in Genetic Algorithms.
10. Solving optimization problems using Genetic Algorithms.
11. Design and implementation of fuzzy sets and membership functions.
12. Development of fuzzy inference systems using Mamdani and Sugeno models.
13. Design of fuzzy logic-based control systems.
14. Development of Neuro-Fuzzy (ANFIS) models for prediction and control.
15. Pattern recognition and classification using neural networks.
16. Image processing applications using soft computing techniques.
17. Comparative performance analysis of ANN, Fuzzy Logic, and Genetic Algorithms.
18. Mini-project involving the application of soft computing techniques to a real-world problem.

Suggested Software/Tools : Python, Jupyter Notebook, Google Colab, TensorFlow, Keras, Scikit-learn, NumPy, Pandas, Matplotlib, MATLAB (optional)

SECOND SEMESTER

Course Code: CS-DISS-459

Course Title: Dissertation-I
(Dissertation)

Prerequisites:

Course Category: Dissertation

Course Type: Practical

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/Internal Assessment	Total Marks	To develop research skills through problem identification, literature review, methodology design, experimentation, and preliminary analysis in science domain.	- Identify and formulate a relevant research problem in AI and data science. - Conduct a systematic literature review and identify research gaps. - Design an appropriate methodology and experimental framework. - Implement preliminary experiments and analyze research results. - Prepare and present a research proposal and progress report.
0	0	2	2		-	-	100		

Dissertation-I Activities (Research Proposal and Preliminary Study)

- Research Domain Selection:** Select a research area relevant to Computer Science/Engineering/AI etc.
- Problem Identification and Formulation:** Identify a research problem, define its scope, objectives, research questions, and expected contributions.
- Literature Survey:** Conduct a systematic review of relevant books, research papers, datasets, and existing methodologies.
- Research Gap Identification:** Compare existing approaches and identify limitations, gaps, and opportunities for improvement.
- Research Methodology Design:** Develop the proposed research framework, methodology, algorithms, models, and experimental plan.
- Dataset Identification and Preparation:** Identify appropriate datasets and perform data collection, cleaning, preprocessing, and exploratory analysis.
- Baseline Implementation:** Implement suitable existing methods or baseline models for subsequent comparison.
- Preliminary Experimentation:** Conduct initial experiments and evaluate baseline performance using appropriate metrics.
- Research Proposal Preparation:** Prepare a structured proposal containing the problem statement, literature review, research gap, objectives, methodology, and expected outcomes.
- Progress Presentation:** Present the proposed research and preliminary findings before the evaluation committee.

SECOND SEMESTER

Course Code:CS-VAC-499A
Course Title: Scientific Writing with LaTeX

Prerequisites:None.

Course Category: VAC
Course Type: Theory+Practice

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To develop practical skills in scientific writing, LaTeX typesetting, academic formatting, and preparation of professional research documents.	- Explain the fundamentals of scientific writing and LaTeX. - Create structured scientific documents using LaTeX. - Format equations, tables, figures, citations, and bibliographies. - Prepare research papers, reports, and theses using standard templates. - Develop complete scientific documents using LaTeX and collaborative tools.
0	0	0	0	20			100		

Unit 1: Introduction to Scientific Writing and LaTeX: Principles of scientific and technical writing, characteristics of scientific documents, structure of research papers, introduction to LaTeX, installation and online platforms, document classes, document structure, packages, commands, environments, compilation, comments, and basic text formatting. **[4H]**

Unit 2: Document Structure and Mathematical Typesetting: Sections and subsections, title pages, abstracts, paragraphs, lists, footnotes, cross-referencing, mathematical expressions, symbols, equations, aligned equations, matrices, cases, theorem-like environments, and mathematical notation. **[4H]**

Unit 3: Tables, Figures and Scientific Presentation: Creating and formatting tables, multi-column and multi-row tables, captions and labels, inserting and positioning figures, subfigures, diagrams, graphs, floating environments, references to figures and tables, and presentation of scientific results. **[4H]**

Unit 4: Citations, Bibliography and Academic Documents: Citation management, BibTeX and BibLaTeX, bibliography styles, reference databases, DOI and URL handling, citation commands, indexes and glossaries, appendices, preparing manuscripts using journal and conference templates, and avoiding common formatting errors. **[4H]**

Unit 5: Research Paper and Thesis Preparation: Structuring a complete research paper, thesis and project report preparation, page layout and formatting, custom commands, advanced tables and figures, source-code listings, algorithms and pseudocode, presentation of experimental results, collaborative writing using Overleaf, version control concepts, and preparation of a complete scientific document. **[4H]**

Suggested Practical Emphasis

- Create a basic LaTeX document using appropriate document classes and packages.
- Format sections, subsections, lists, quotations, footnotes, and references.
- Typeset mathematical equations, matrices, symbols, and mathematical expressions.
- Create professional tables using **tabular**, **multirow**, and **multicolumn**.
- Insert, position, caption, label, and cross-reference figures and subfigures.
- Manage citations and bibliographies using BibTeX/BibLaTeX.
- Prepare algorithms, pseudocode, and source-code listings.
- Format a research paper using an IEEE/ACM/journal template.
- Prepare a thesis/project report with chapters, appendices, references, figures, and tables.
- Mini Project:** Prepare a complete scientific research document in LaTeX, including title, abstract, sections, equations, tables, figures, citations, bibliography, and appendices.

Suggested Software/Tools: Overleaf, LaTeX, TeX Live/MiKTeX, BibTeX/BibLaTeX, JabRef, and Git/GitHub (optional).

SECOND SEMESTER

Course Code:CS-VAC-499B

Course Title: Open Source Software and Collaborative Development

Prerequisites: Basic Programming

Course Category: VAC

Course Type: Theory+Practice

Credit Structure					Marks Distribution			Objective	Course Learning Outcomes
Lecture	Tutorial	Practice	Total Credit	Hours	Summative Assessment	Continuous/ Internal Assessment	Total Marks	To develop practical skills in open-source software development, version control, collaborative workflows, licensing, and community-based project development.	• Explain open-source principles, software licenses, and development practices. • Apply Git and GitHub for version control and collaborative development. • Manage branches, issues, pull requests, code reviews, and project workflows. • Contribute effectively to open-source projects following community guidelines. • Develop and manage a collaborative open-source software project.
0	0		0	20			100		

Unit I: Introduction to Open Source Software: Concept of open source software, history and evolution of the open-source movement, open-source principles, free software and open source, open-source licenses, permissive and copyleft licenses, major open-source ecosystems, benefits and challenges of open-source development, and examples of open-source projects. **[4H]**

Unit II: Version Control and Git: Introduction to version control, centralized and distributed version control systems, Git fundamentals, repositories, commits, branches, merging, tagging, remote repositories, GitHub/GitLab, .gitignore, branching strategies, resolving merge conflicts, and basic Git workflows. **[4H]**

Unit III: Collaborative Software Development: Collaborative development workflows, GitHub-based collaboration, forks and clones, pull requests, issues, milestones, project boards, code review, contribution guidelines, documentation, release management, semantic versioning, continuous integration concepts, and managing collaborative software projects. **[4H]**

Unit IV: Open Source Contribution and Project Management: Finding suitable open-source projects, understanding project structure and documentation, identifying issues, creating contributions, submitting patches and pull requests, code review and feedback, issue tracking, project communication, community participation, contributor roles, governance models, and best practices for sustainable open-source projects. **[4H]**

Unit V: Open Source Licensing, Ethics and Practical Applications: Software licensing and copyright, license compatibility, attribution and compliance, ethical considerations in open-source development, security and responsible contribution, open-source software in research and industry, open-source communities, case studies of major open-source projects, and development of a collaborative open-source mini project. **[4H]**

Suggested practical emphasis

1. Creating and managing Git repositories
2. Branching and merging
3. Collaborative GitHub development
4. Issue and milestone management
5. Pull-request creation and review
6. Resolving merge conflicts
7. Writing README and contribution documentation
8. Applying an open-source license
9. Participating in an existing open-source project
10. **Mini Project:** Collaborative development and publication of an open-source software project