



R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY
(An Autonomous Institution)

R.S.M. Nagar, PUDUVOYAL-601 206
Approved by AICTE, New Delhi /Affiliated to Anna University, Chennai
Accredited by NBA, New Delhi (All Eligible Courses)/ NAAC with 'A' Grade
An ISO 21001:2018 Certified Institution

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB MANUAL

22AI401 - MACHINE LEARNING (LAB INTEGRATED)

Academic Year : 2025-2026
Regulations : 2022
Batch : 2023-2027
Year / Semester : III / V

Prepared By,

Dr. Pranamita Nanda, ASP / CSE

Ms Sreelekha P, AP / CSE



R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY
(An Autonomous Institution)

R.S.M. Nagar, PUDUVOYAL-601 206
Approved by AICTE, New Delhi /Affiliated to Anna University, Chennai
Accredited by NBA, New Delhi (All Eligible Courses)/ NAAC with 'A' Grade
An ISO 21001:2018 Certified Institution

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB MANUAL

22AI401 - MACHINE LEARNING (LAB INTEGRATED)

Academic Year : 2025-2026
Regulations : 2022
Batch : 2023-2027
Year / Semester : III / V

Prepared By,

Dr. Pranamita Nanda, ASP / CSE

Ms Sreelekha P, AP / CSE



R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution)

R.S.M. Nagar, PUDUVOYAL-601 206

Approved by AICTE, New Delhi /Affiliated to Anna University, Chennai
Accredited by NBA, New Delhi (All Eligible Courses)/ NAAC with 'A' Grade

An ISO 21001:2018 Certified Institution

Institute Vision and Mission

Vision

To be knowledge hub of providing quality technical education and promoting research for building up of our nation and its contribution for the betterment of humanity

Mission

- To make the best use of state-of-the-art infrastructure to ensure quality technical education
- To develop industrial collaborations to promote innovation and research capabilities
- To inculcate values and ethics to serve humanity



R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution)

R.S.M. Nagar, PUDUVOYAL-601 206

Approved by AICTE, New Delhi /Affiliated to Anna University, Chennai
Accredited by NBA, New Delhi (All Eligible Courses)/ NAAC with 'A' Grade
An ISO 21001:2018 Certified Institution

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision

To be a source of knowledge in the field of Computer Science and Engineering to cater to the growing need of industry and society

Mission

- To avail state-of-the art infrastructure for adopting cutting edge technologies and encouraging research activities
- To promote industrial collaborations for professional competency
- To nurture social responsibility and ethics to become worthy citizens



R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution)

R.S.M. Nagar, PUDUVOYAL-601 206

Approved by AICTE, New Delhi /Affiliated to Anna University, Chennai
Accredited by NBA, New Delhi (All Eligible Courses)/ NAAC with 'A' Grade
An ISO 21001:2018 Certified Institution

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Program Educational Objectives (PEOs)

Graduates of Computer Science and Engineering Program will

1. Become a globally competent professional in all spheres and pursue higher education world over.
2. Successfully carry forward domain knowledge in computing and allied areas to solve complex real world engineering problems.
3. Continuously upgrade their technical knowledge and expertise to keep pace with the technological revolution.
4. Serve the humanity with social responsibility combined with ethics.

Program Specific Outcomes (PSOs)

Graduates of Computer Science and Engineering Program will be able to:

- Analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.
- Apply software engineering principles and practices for developing quality software for scientific and business applications.
- Implement cost-effective solutions for the betterment of both industry and society with the technological skills acquired through the Centres of Excellence.



R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution)

R.S.M. Nagar, PUDUVOYAL-601 206

Approved by AICTE, New Delhi /Affiliated to Anna University, Chennai
Accredited by NBA, New Delhi (All Eligible Courses)/ NAAC with 'A' Grade
An ISO 21001:2018 Certified Institution

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Program Outcome

1. **Engineering Knowledge:** Apply knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.
2. **Problem Analysis:** Identify, formulate, research literature and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and engineering sciences.
3. **Design/ Development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal and environmental considerations.
4. **Conduct investigations of complex problems** using research-based knowledge and research methods including design of experiments, analysis and interpretation of data and synthesis of information to provide valid conclusions.
5. **Modern Tool Usage:** Create, select and apply appropriate techniques, resources and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The Engineer and Society:** Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to professional engineering practice.
7. **Environment and Sustainability:** Understand the impact of professional engineering solutions in societal and environmental contexts and demonstrate knowledge of and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of engineering practice.
9. **Individual and Team Work:** Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations and give and receive clear instructions.
11. **Project Management and Finance:** Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long Learning:** Recognize the need for and have the preparation and ability to Engage in independent and life- long learning in the broadest context of technological Change.



R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution)

R.S.M. Nagar, PUDUVOYAL-601 206

Approved by AICTE, New Delhi /Affiliated to Anna University, Chennai
Accredited by NBA, New Delhi (All Eligible Courses)/ NAAC with 'A' Grade
An ISO 21001:2018 Certified Institution

22AI401	MACHINE LEARNING (Lab Integrated)	L	T	P	C
		3	0	2	4

COURSE OBJECTIVES:

- To discuss the basics of Machine Learning and model evaluation.
- To study dimensionality reduction techniques.
- To understand the various classification algorithms.
- To elaborate on unsupervised learning techniques.
- To discuss the basics of neural networks and various types of learning.

UNIT I	INTRODUCTION	9+6
--------	--------------	-----

Machine Learning – Types – Applications – Preparing to Model – Activities – Data – Exploring structure of Data – Data Quality and Remediation – Data Pre-processing – Modelling and Evaluation: Selecting a Model -Training a Model – Model representation and Interpretability – Evaluating Performance of a Model – Improving Performance.

Lab Programs:

1. Implementation of Candidate Elimination algorithm
2. Implementation of ML model evaluation techniques (R-Squared/Adjusted R-Squared/Mean Absolute Error/Mean Squared Error)
3. Implementation of ML model evaluation techniques (Confusion Matrix/F1 Score/AUC-ROC Curve)

UNIT II	FEATURE ENGINEERING AND DIMENSIONALITY REDUCTION	9+6
---------	--	-----

Feature Engineering – Feature Transformation – Feature Subset Selection - Principle Component Analysis – Feature Embedding – Factor Analysis – Singular value decomposition and Matrix Factorization – Multidimensional scaling – Linear Discriminant Analysis – Canonical Correlation Analysis – Isomap – Locally linear Embedding – Laplacian Eigenmaps.

Lab Programs:

1. Write python code to identify feature co-relations (PCA)
2. Interpret Canonical Covariates with Heatmap
3. Feature Engineering is the way of extracting features from data and transforming them into formats that are suitable for Machine Learning algorithms. Implement python code for Feature Selection/ Feature Transformation/ Feature Extraction.
4. Mini Project – Feature Subset Selection

UNIT III	SUPERVISED LEARNING	9+6
----------	---------------------	-----

Linear Regression -Relation between two variables – Steps – Evaluation – Logistic Regression – Decision Tree – Algorithms – Construction – Classification using Decision Tree – Issues – Rule - based Classification – Pruning the Rule Set – Support Vector Machines – Linear SVM – Optimal Hyperplane – Radial Basis Functions – Naïve Bayes Classifier – Bayesian Belief Networks.

Lab Programs:

1. Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select the appropriate data set for your experiment and draw graphs.
2. Implement and demonstrate the working of the decision tree-based ID3 algorithm
3. Build a Simple Support Vector Machines using a data set

UNIT IV	UNSUPERVISED LEARNING	9+6
----------------	------------------------------	------------

Clustering – Types – Applications - Partitioning Methods – K-means Algorithm – K-Medoids – Hierarchical methods – Density based methods DBSCAN – Finding patterns using Association Rules – Hidden Markov Model.

Lab Programs:

1. Implement a k-Nearest Neighbour algorithm to classify the iris data set. Print both correct and wrong predictions
2. Implement market basket analysis using association rules

Mini Project using Clustering analysis.

UNIT V	NEURAL NETWORKS AND TYPES OF LEARNING	9+6
---------------	--	------------

Biological Neuron – Artificial Neuron – Types of Activation function – Implementations of ANN – Architectures of Neural Networks – Learning Process in ANN – Back propagation – Deep Learning – Representation Learning – Active Learning – Instance based Learning – Association Rule Learning – Ensemble Learning Algorithm – Regularization Algorithm- Reinforcement Learning – Elements- Model-based- Temporal Difference Learning.

Lab Programs:

1. Build an ANN by implementing the Single-layer Perceptron. Test it using appropriate data sets.
2. Implement Multi-layer Perceptron and test the same using appropriate data sets.
3. Build a RBF Network to calculate the fitness function with five neurons.

Mini Project – Face recognition,

TOTAL: 45 +30=75 PERIODS

OUTCOMES:

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

CO1: Explain the basics of Machine Learning and model evaluation.

CO2: Study dimensionality reduction techniques.

CO3: Understand and implement various classification algorithms.

CO4: Understand and implement various unsupervised learning techniques.

CO5: Build Neural Networks and understand the different types of learning.

CO6: Develop simple projects using machine learning concepts

TEXTBOOKS:

1. Saikat Dutt, Subramanian Chandramouli, Amit Kumar Das, “Machine Learning”, Pearson,2019. (Unit 1 – chap 1,2,3/ Unit 2 – Chap 4 / Unit 4 – 9 / Unit 5 – Chap 10, 11)
2. Ethem Alpaydin, “Introduction to Machine Learning, Adaptive Computation and Machine Learning

Series”, Third Edition, MIT Press, 2014. (Unit 2 – Chap 6 / Unit 4 – chap 8.2.3/ Unit 5 – Chap 18)

REFERENCES:

1. Anuradha Srinivasaraghavan, Vincy Joseph, “Machine Learning”, First Edition, Wiley, 2019. (Unit 3 – Chap 7,8,9,10,11 / Unit 4 – 13, 11.4, 11.5,12)
2. Peter Harrington, “Machine Learning in Action”, Manning Publications, 2012.
3. Stephen Marsland, “Machine Learning – An Algorithmic Perspective”, Second Edition,
4. Chapman and Hall/CRC Machine Learning and Pattern Recognition Series, 2014.
5. Tom M Mitchell, “Machine Learning”, First Edition, McGraw Hill Education, 2013.
6. Christoph Molnar, “Interpretable Machine Learning - A Guide for Making Black Box Models Explainable”, Creative Commons License, 2020.
7. NPTEL Courses:
 - a. Introduction to Machine Learning - https://onlinecourses.nptel.ac.in/noc23_cs18/preview

SOFTWARE REQUIREMENTS:

Systems with Anaconda, Jupyter Notebook, Python, Pytorch, scikit-learn, Tensorflow, Colab

Course Articulation Matrix

Course Code/ Course Name: 22AI401/ Machine Learning (Lab Integrated)

COURSE OUTCOMES	
CO1	Explain the basics of Machine Learning and model evaluation.
CO2	Study dimensionality reduction techniques
CO3	Understand and implement various classification algorithms.
CO4	Understand and implement various unsupervised learning techniques.
CO5	Build Neural Networks and understand the different types of learning.
CO6	Develop simple projects using machine learning concepts.

Course Outcome – Program Outcome & Program Specific Outcome Mapping:

Course Outcomes (COs)	Program Outcomes (POs), Program Specific Outcomes (PSOs)														
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	3	3	2	-	2	-	-	-	-	-	-	2	3	2	-
CO2	3	2	2	-	2	-	-	1	-	-	-	2	3	3	-
CO3	3	2	2	-	2	-	-	1	-	-	-	2	3	2	-
CO4	3	2	2	-	2	-	-	1	-	-	-	2	3	2	-
CO5	3	3	3	-	2	-	-	1	-	-	-	2	3	2	-
CO6	3	3	2	-	2	-	-	1	-	-	-	2	3	2	-

EXPERIMENTS WITH MAPPED CO

S.NO	NAME OF THE EXERCISE	Mapped CO
1	Implementation of Candidate Elimination algorithm	CO1
2	Implementation of ML model evaluation techniques (R-Squared/Adjusted R-Squared/Mean Absolute Error/Mean Squared Error)	CO1
3	Implementation of ML model evaluation techniques (Confusion Matrix/F1 Score/AUC-ROC Curve)	CO1
4	Write python code to identify feature co-relations (PCA)	CO2
5	Interpret Canonical Covariates with Heatmap	CO2
6	Feature Engineering is the way of extracting features from data and transforming them into formats that are suitable for Machine Learning algorithms. Implement python code for Feature Selection/ Feature Transformation/ Feature Extraction	CO2
7	Mini Project – Feature Subset Selection	CO2
8	Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select the appropriate data set for your experiment and draw graphs.	CO3
9	Implement and demonstrate the working of the decision tree-based ID3 algorithm	CO3
10	Build a Simple Support Vector Machines using a data set	CO3
11	Implement a k-Nearest Neighbour algorithm to classify the iris data set. Print both correct and wrong predictions	CO3
12	Implement market basket analysis using association rules	CO4
13	Mini Project using Clustering analysis.	CO4
14	Build an ANN by implementing the Single-layer Perceptron. Test it using appropriate data sets.	CO5
15	Implement Multi-layer Perceptron and test the same using appropriate data sets.	CO5
16	Build a RBF Network to calculate the fitness function with five neurons.	CO5
17	Mini Project – Face recognition,	CO1, CO2, CO5,CO6

VIVA QUESTIONS

1. Candidate Elimination Algorithm

1. What is the Candidate Elimination algorithm?
2. Define Version Space.
3. What are the specific and general boundaries?
4. What is the hypothesis space?
5. Why is Candidate Elimination sensitive to noise?
6. What type of learning does it support?
7. What happens when a positive example is encountered?
8. What is the role of the general hypothesis?
9. How is consistency maintained in hypotheses?
10. What is the limitation of this algorithm?

2. Model Evaluation (R^2 , Adjusted R^2 , MAE, MSE)

1. What does R^2 value indicate?
2. What is the range of R^2 ?
3. How is Adjusted R^2 calculated?
4. When would Adjusted R^2 be more useful than R^2 ?
5. Define Mean Absolute Error.
6. Define Mean Squared Error.
7. Which is more sensitive to outliers, MAE or MSE?
8. What is a good R^2 value?
9. How do you interpret a negative R^2 ?
10. What does a low MAE signify?

3. Classification Evaluation (Confusion Matrix, F1, AUC-ROC)

1. What is a confusion matrix?
2. Define TP, TN, FP, FN.
3. What is the accuracy metric?
4. What is precision?
5. What is recall?

6. What is the F1 score?
7. Why is F1 score important?
8. What is an ROC curve?
9. What does AUC represent?
10. How can you improve the AUC score?

4. PCA for Feature Correlation

1. What is Principal Component Analysis?
2. What is the purpose of PCA?
3. What are eigenvectors and eigenvalues?
4. Why is standardization important in PCA?
5. What is dimensionality reduction?
6. How many components should be selected?
7. Can PCA be used on categorical data?
8. What is the explained variance ratio?
9. How does PCA handle multicollinearity?
10. What are the limitations of PCA?

5. Canonical Covariates with Heatmap

1. What are canonical covariates?
2. What is a heatmap?
3. What does a heatmap show?
4. How do you interpret color in a heatmap?
5. What does a correlation coefficient of +1 mean?
6. What does -1 correlation indicate?
7. How is a heatmap useful in ML?
8. How do you calculate correlation?
9. What is covariance?
10. Difference between correlation and covariance?

6. Feature Engineering (Selection, Transformation, Extraction)

1. What is feature engineering?
2. Why is feature engineering important?
3. What is feature selection?
4. What is feature transformation?
5. What is feature extraction?
6. Name any two feature selection methods.
7. What is normalization?
8. What is standardization?
9. What is one-hot encoding?
10. How does PCA help in feature extraction?

7. Mini Project – Feature Subset Selection

1. What is subset selection in ML?
2. Why is it necessary to reduce features?
3. What is Recursive Feature Elimination (RFE)?
4. What is a wrapper method?
5. What is a filter method?
6. What is a hybrid method?
7. How is model performance affected by irrelevant features?
8. What is overfitting and how does it relate to features?
9. How to evaluate selected features?
10. Difference between feature selection and dimensionality reduction?

8. Locally Weighted Regression

1. What is Locally Weighted Regression (LWR)?
2. How is LWR different from Linear Regression?
3. What does the bandwidth parameter (τ) control?
4. Is LWR parametric or non-parametric?
5. What kernel function is used in LWR?
6. What is the role of weights in LWR?
7. When would you prefer LWR over linear regression?

8. What is the time complexity of LWR?
9. Can LWR handle non-linear data?
10. What is the main drawback of LWR?

9. Decision Tree (ID3 Algorithm)

1. What is a decision tree?
2. What is the ID3 algorithm?
3. Define entropy.
4. Define information gain.
5. Why is ID3 suitable for categorical data?
6. How does ID3 build the tree?
7. What are the limitations of ID3?
8. How can overfitting be avoided in decision trees?
9. What is pruning?
10. What are leaf nodes in decision trees?

10. Support Vector Machines (SVM)

1. What is the basic idea behind SVM?
2. What is a hyperplane?
3. What are support vectors?
4. What is the kernel trick?
5. Name different types of kernels.
6. What does the C parameter control?
7. What is the margin in SVM?
8. How does SVM handle outliers?
9. Can SVM be used for regression?
10. What are advantages and disadvantages of SVM?

11. k-Nearest Neighbour (k-NN)

1. What is k-NN?

2. How is the value of k selected?
3. What is a lazy learner?
4. What distance metrics are used in k-NN?
5. Is k-NN sensitive to irrelevant features?
6. How does k-NN handle missing values?
7. What is the time complexity of k-NN?

8. Can k-NN be used for regression?
9. How does k-NN work for multi-class classification?
10. What is the effect of high-dimensional data on k-NN?

12. Market Basket Analysis (Association Rules)

1. What is Market Basket Analysis?
2. What is support?
3. What is confidence?
4. What is lift?
5. What does lift > 1 indicate?
6. Name algorithms for association rule mining.
7. What are frequent itemsets?
8. What is the Apriori principle?
9. What is FP-Growth algorithm?
10. What are real-life applications of association rules?

13. Mini Project – Clustering Analysis

1. What is clustering?
2. Is clustering supervised or unsupervised?
3. What is the KMeans algorithm?
4. What is the Elbow method?
5. What is hierarchical clustering?
6. What is DBSCAN?
7. What is the silhouette score?

8. What is centroid in KMeans?
9. How do you choose the number of clusters?
10. Applications of clustering in real life?

14. Single-layer Perceptron

1. What is a perceptron?
2. Who invented the perceptron?
3. What activation function is used?
4. What type of data can a perceptron classify?
5. Can a single-layer perceptron solve XOR?
6. What is the perceptron learning rule?
7. What is the bias term?
8. What are the weights in a perceptron?
9. What is a threshold function?
10. What is the limitation of single-layer perceptrons?

15. Multi-layer Perceptron (MLP)

1. What is an MLP?
2. What is the structure of MLP?
3. What is backpropagation?
4. What activation functions are used?
5. What is the vanishing gradient problem?
6. What is the difference between MLP and logistic regression?
7. How many hidden layers should be used?
8. What is the cost function in MLP?
9. What is epoch in training?
10. What libraries are used to implement MLP in Python?

16. Radial Basis Function (RBF) Network

1. What is an RBF network?
2. What activation function is used in RBF?

3. What is the difference between RBF and MLP?
4. What is the role of centers in RBF?
5. What is the Gaussian function?
6. What is the spread (sigma) in RBF?
7. How are weights calculated in RBF?
8. What are the advantages of RBF?
9. What are limitations of RBF networks?
10. Where are RBF networks used?

17. Mini Project – Face Recognition

1. What is face recognition?
2. What features are extracted from face images?
3. What is Eigenface?
4. How does PCA help in face recognition?
5. What is LBP (Local Binary Pattern)?
6. What challenges exist in face recognition?
7. What is the difference between face detection and face recognition?
8. What is a distance threshold in recognition?
9. What is OpenCV?
10. What real-time applications use face recognition?

Rubrics for Assessment

	Excellent	Good	Average	Poor
Problem Understanding & Logic Design	Clear understanding with efficient logic (3 marks)	Good logic with minor issues (2 marks)	Partial understanding (1 mark)	unclear understanding (0 mark)
Code Implementation	Error-free, well-structured, meets all requirements (3 marks)	Minor errors, mostly meets objectives (2 marks)	Major errors (1 mark)	incomplete implementation (0 mark)
Output & Viva Explanation	All test cases passed, confident explanation (4 marks)	Some test cases passed, fair explanation (3 marks)	Few test cases passed, weak explanation (2 marks)	Attempted, no test cases passed, no explanation (1 mark)



R.M.K. COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution)

R.S.M. Nagar, PUDUVOYAL-601 206

Approved by AICTE, New Delhi /Affiliated to Anna University, Chennai
Accredited by NBA, New Delhi (All Eligible Courses)/ NAAC with 'A' Grade
An ISO 21001:2018 Certified Institution

LIST OF EXPERIMENTS

SL.NO	NAME OF THE EXPERIMENT
1	Implementation of Candidate Elimination algorithm
2	Implementation of ML model evaluation techniques (R-Squared/Adjusted R-Squared/Mean Absolute Error/Mean Squared Error)
3	Implementation of ML model evaluation techniques (Confusion Matrix/F1 Score/AUC-ROC Curve)
4	Write python code to identify feature co-relations (PCA)
5	Interpret Canonical Covariates with Heatmap
6	Feature Engineering is the way of extracting features from data and transforming them into formats that are suitable for Machine Learning algorithms. Implement python code for Feature Selection/ Feature
7	Mini Project – Feature Subset Selection
8	Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select the appropriate data set for your experiment and draw graphs.
9	Implement and demonstrate the working of the decision tree-based ID3 algorithm
10	Build a Simple Support Vector Machines using a data set
11	Implement a k-Nearest Neighbour algorithm to classify the iris data set. Print both correct and wrong predictions
12	Implement market basket analysis using association rules
13	Mini Project using Clustering analysis.
14	Build an ANN by implementing the Single-layer Perceptron. Test it using appropriate data sets.
15	Implement Multi-layer Perceptron and test the same using appropriate datasets.
16	Build a RBF Network to calculate the fitness function with five neurons.
17	Mini Project – Face recognition,

EX NO:01

CANDIDATE ELIMINATION ALGORITHM

DATE:

Aim:

The Candidate Elimination Algorithm learns from examples to identify concepts. It does this by refining a set of possible descriptions (hypotheses) based on whether examples are positive matches or not.

Algorithm :

Step1: Load Data set

Step2: Initialize General Hypothesis and Specific Hypothesis.

Step3: For each training example

Step4: If example is positive example

 if attribute_value == hypothesis_value:

 Do nothing

 else:

 replace attribute value with '?' (Basically generalizing it)

Step5: If example is Negative example

 Make generalize hypothesis more specific.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

The Candidate Elimination algorithm was successfully implemented and executed. The version space was updated correctly based on positive and negative training examples, resulting in a specific and general hypothesis that is consistent with the training data.

EXP NO:2**2A. R-Squared Error****DATE:****Aim:**

To write a program to implement model evaluation technique R-Squared Error.

Algorithm:

1. First, calculate the mean of the target/dependent variable y and we denote it by \bar{y}
2. Calculate the Total Sum of Squares (TSS) by subtracting each observation y_i from \bar{y} , then squaring it and summing these square differences across all the values. It is denoted by

$$TSS = \sum_{i=1}^n (y_i - \bar{y})^2$$

3. We estimate the model parameter using a suitable regression model such as Linear Regression or SVM Regressor
4. We calculate the Sum of squares due to regression which is denoted by RSS. This is calculated by subtracting each predicted value of y denoted by $y_{\text{pred}i}$ from y_i squaring these differences and then summing all the n terms.

Calculate the regression sum of squares (RSS).

$$RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

3. Calculate R-squared using the formula: $R^2 = 1 - (RSS / TSS)$.

2B. Adjusted RSquared

Aim:

To write a program to implement Adjusted R-Squared error model evaluation technique.

Algorithm:

1. First, calculate the mean of the target/dependent variable y and we denote it by \bar{y}

2. Calculate the Total Sum of Squares (TSS) by subtracting each observation y_i from \bar{y} , then squaring it and summing these square values. It is denoted by

$$TSS = \sum_{i=1}^n (y_i - \bar{y})^2$$

2. We calculate the Sum of which is denoted by RSS. subtracting each predicted y_{predi} from y_i squaring summing all the n terms.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

squares due to regression
This is calculated by value of y denoted by these differences and then

Calculate the regression sum of squares (RSS).

$$RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

3. Calculate the number of predictors (p) in the model.

4. Determine the number of observations(n).

5. Calculate the adjusted R-squared using the formula:

$$\text{Adjusted } R^2 = 1 - ((1 - R^2) * ((n - 1) / (n - p - 1))).$$

2C. Mean Absolute Error

Aim:

To write a program to implement a model evaluation technique-mean absolute error.

Algorithm:

Input y and yCap. Where y is the Array of actual target values and yCap is an array of predicted target values.

1. Initialize a variable to zero.
2. For each prize of actual (y_i)and predicted variable(y_i Cap).

$var += |y_i - y_iCap|$

$var /= n$, where n is the number of data points.

4. Calculate Mean Absolute Error

$$MAE = \frac{\sum_{i=1}^n |y_i - \hat{y}_i|}{n}$$

2D.Mean Squared Error

Aim:

To write a program to implement model evaluation technique-Mean Squared Error.

Algorithm:

1. Find the equation for the regression line.
2. Insert X values in the equation found in step 1 in order to get the respective Y values i.e. \hat{Y}_i
3. Now subtract the new Y values (i.e. \hat{Y}_i) from the original Y values. Thus, found values are the error terms. It is also known as the vertical distance of the given point from the regression line. $Y_i - \hat{Y}_i$
4. Square the errors found in step 3. $(Y_i - \hat{Y}_i)^2$.
5. Sum up all the squares. $\sum_{i=0}^n (Y_i - \hat{Y}_i)^2$
6. Divide the value found in step 5 by the total number of observations.

$$MSE = \frac{1}{n} \sum_{i=0}^n (Y_i - \hat{Y}_i)^2$$

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

Various regression model evaluation metrics such as R^2 , Adjusted R^2 , Mean Absolute Error, and Mean Squared Error were computed to evaluate the model performance. The results helped in identifying the goodness of fit and accuracy of the regression model.

EX NO:3

IMPLEMENTATION OF ML MODEL EVALUATION

DATE:

TECHNIQUES

A. CONFUSION MATRIX

AIM:

To write a program to implementation of machine learning model evaluation Technique using confusion matrix.

ALGORITHM:

1. Initialize matrix: Create a square matrix with dimensions (N x N), where N is the number of classes in your classification problem. Initialize all values in the matrix to zero.
2. Iterate over predictions and ground truth: For each prediction-ground truth pair in your dataset:
 - If the predicted class matches the actual class:
 - Increment the corresponding entry in the confusion matrix (true positive).
 - If the predicted class does not match the actual class:
 - Increment the entry in the confusion matrix corresponding to the actual class and the predicted class (false positive for the predicted class, false negative for the actual class).
3. Calculate metrics: Optionally, you can calculate various evaluation metrics using the values in the confusion matrix, such as accuracy, precision, recall, and F1-score.

B. F1 Score

AIM:

To write a program to implementation of machine learning model evaluation Technique using F1 score

ALGORITHM:

1. Calculate Precision and Recall for Each Class:

For each class in your classification problem:

Calculate precision: $\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positive}}$

Calculate recall: $\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negative}}$

1. Calculate F1 Score for Each Class:

For each class in your classification problem:

Calculate F1 score: $\text{F1} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$

2. Aggregate F1 Scores:

Depending on your needs, you may want to aggregate the F1 scores in different ways. Common approaches include:

- Micro-average: Compute F1 score globally by considering total true positives, false positives, and false negatives across all classes.
- Macro-average: Compute the average F1 score across all classes without considering class imbalance.
- Weighted-average: Compute the average F1 score across all classes with each class weighted by its support (the number of true instances for each class).

C. AUCROC Curve

AIM:

To write a program to implementation of machine learning model evaluation Technique using AUCROC Curve

ALGORITHM:

- 1.Sort Predictions: Sort the predictions generated by your binary classification algorithm in descending order of confidence scores.
2. Initialize Variables: Set variables for true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) to zero.
- 3.Iterate Through Sorted Predictions:
 - Start iterating through the sorted predictions.
 - For each prediction:
 - If the prediction is a true positive (actual positive and predicted positive), increment TP.
 - If the prediction is a false positive (actual negative but predicted positive), increment FP.
 - If the prediction is a true negative (actual negative and predicted negative), increment TN.
 - If the prediction is a false negative (actual positive but predicted negative), increment FN.
3. Calculate True Positive Rate (TPR) and False Positive Rate (FPR):
 - True Positive Rate (TPR) is calculated as $TPR = TP / (FN + TP)$
 - False Positive Rate (FPR) is calculated as $FPR = FP / (TN + FP)$
4. Plot ROC Curve: Plot the FPR on the x-axis and the TPR on the y-axis.
5. Calculate AUC-ROC:
 - Calculate the area under the ROC curve using various methods such as trapezoidal rule or other numerical integration methods.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

Classification performance was evaluated using confusion matrix, F1 score, and AUC-ROC curve. The results provided insights into the precision, recall, and overall classification capability of the model.

EX NO:04

IDENTIFYING FEATURE CO-RELATION USING PCA

DATE:

AIM:

To write a python program to identify feature co-relation using PCA.

ALGORITHM:

1. Generated a sample dataset.
2. Identified the mean of each feature.
3. Computed the covariance matrix.
4. Computed eigenvalues and eigenvectors of the covariance matrix.
5. Sort the eigenvalues and eigenvectors in descending order.
6. Compute explained variance ratio.
7. Calculate cumulative explained variance and plot it.
8. Select the number of components based on the explained variance.
9. Project data onto the selected number of components.
10. Visualize the feature weight in the first principal component.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

Principal Component Analysis was applied to reduce the dimensionality and analyze feature correlations. The results were visualized, and key components contributing to data variance were identified.

EXP.NO: 5

Implementation of Interpret Canonical Covariates with Heatmap

Date:

Aim :

Implement interpret Canonical Covariates with Heatmap

Algorithm :

1.Perform Canonical Correlation Analysis (CCA):

- Perform CCA on your dataset, which involves two sets of variables (X and Y). CCA finds linear combinations of variables (canonical variates) from each set that maximize the correlation between them.

2.Obtain Canonical Loadings:

- After performing CCA, obtain the canonical loadings for each canonical variate. Canonical loadings represent the correlations between the original variables and the canonical variates.

3.Normalize Canonical Loadings:

- Normalize the canonical loadings to ensure comparability across variables. You can use methods like z-score normalization or min-max normalization.

4. Create a Heatmap:

- Create a heatmap to visualize the canonical loadings.
- Arrange the canonical loadings in a matrix where rows represent the variables from one set (e.g., X) and columns represent the canonical variates.
- Use a color gradient to represent the strength and direction of the canonical loadings. Positive loadings may be represented with one color (e.g., blue), while negative loadings may be represented with another color (e.g., red).

5.Interpret the Heatmap:

- Analyze the heatmap to interpret the relationships between variables from the two sets.
- Look for patterns of high positive or negative loadings, which indicate strong correlations between specific variables and canonical variates.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

Canonical covariates were interpreted and a heatmap was generated to visualize the correlation between features. Highly correlated variables were identified for further analysis.

EX NO:6

DATE:

**IMPLEMENTATION OF PYTHON CODE FOR FEATURE
SELECTION, TRANSFORMATION AND EXTRACTION**

AIM:

To implement the python code for Feature Selection, Feature Transformation and Feature Extraction.

ALGORITHM:

1. Load the Dataset: Load your dataset into a pandas DataFrame or any other suitable data structure.

2. Feature Selection:

- a. Decide on a feature selection method based on your problem and data characteristics. Common methods include:
- Filter methods: Use statistical measures like correlation, chi-square test, or mutual information to select features.
 - Wrapper methods: Train a machine learning model and select features based on their impact on model performance (e.g., recursive feature elimination).
 - Embedded methods: Select features as part of the model training process (e.g., L1 regularization).
- b. Implement the selected feature selection method using libraries like scikit-learn or feature selection algorithms directly.

3. Feature Transformation:

- a. Choose a feature transformation technique suitable for your data. Common techniques include:
- Standardization (scaling): Scale features to have a mean of 0 and a standard deviation of 1.
 - Normalization: Scale features to a specified range (e.g., [0, 1]).
 - PCA (Principal Component Analysis): Transform features into a lower-dimensional space while preserving the most important information.
 - LDA (Linear Discriminant Analysis): Find linear combinations of features that best separate different classes.
- b. Implement the chosen feature transformation technique using libraries like scikit-learn.

4. Feature Extraction:

- a. Decide on a feature extraction method suitable for your problem. Common techniques include:
- PCA: Extract principal components that capture the maximum variance in the data.
 - LDA: Extract linear discriminants that maximize class separability.
 - t-SNE (t-distributed Stochastic Neighbor Embedding): Non-linear dimensionality reduction technique for visualizing high-dimensional data.
 - Autoencoders: Train neural networks to learn compact representations of data.
- b. Implement the chosen feature extraction method using libraries like scikit-learn or deep learning frameworks like TensorFlow or PyTorch.

5. Apply Feature Selection, Transformation, and Extraction:

- Apply the selected feature selection, transformation, and extraction methods to your dataset.

6. Evaluate the Performance:

- Evaluate the performance of your model using the selected features, transformed features, or extracted features.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

Feature Selection, Transformation, and Extraction techniques were successfully applied to the dataset. These techniques improved the dataset quality and model performance.

Ex No 8

Locally Weighted Regression algorithm

DATE:

Aim :

Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select the appropriate data set for your experiment and draw graphs.

Algorithm:

1. **Calculate distances:**
 - Compute the Euclidean distance between new_point and each data point in X.
 - Store the distances in a vector distances.
2. **Calculate weights:**
 - Use the kernel_function to calculate weights for each data point based on its distance.
 - The weight of a data point is typically higher if it's closer to new_point.
 - Store the weights in a vector weights.
3. **Create weighted data:**
 - Multiply each row of X by its corresponding weight from weights.
 - Multiply y by weights.
 - Store the weighted data in weighted_X and weighted_y, respectively.
4. **Fit a linear model:**
 - Use ordinary least squares (OLS) to fit a linear model to the weighted data weighted_X and weighted_y.
5. **Predict:**
 - Use the fitted linear model to predict the value for new_point.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

The Locally Weighted Regression algorithm was implemented and successfully fit to the dataset. The results were visualized and demonstrated improved fitting for localized regions in the data.

Ex 9

DECISION TREE USING ID3 ALGORITHM

AIM:

To Implement and demonstrate the working of the decision tree-based ID3 algorithm

ALGORITHM:

Algorithm:

1. **Check for base case:**
 - If all samples in data have the same label, create a leaf node with that label and return.
2. **If no attributes remain:**
 - Create a leaf node with the most frequent label in data and return.
3. **Calculate information gain for each attribute:**
 - For each attribute, calculate the information gain using the entropy formula.
4. **Choose the attribute with highest information gain:**
 - Select the attribute with the highest information gain as the splitting attribute.
5. **Create a decision node:**
 - Create a decision node with the chosen attribute as the test condition.
6. **Create child nodes:**
 - For each possible value of the chosen attribute:
 - Create a subset of data containing samples with that value.
 - Recursively call the ID3 algorithm on the subset to create a child node.
7. **Attach child nodes:**
 - Attach the created child nodes to the decision node.
8. **Return the tree:**
 - Return the constructed decision tree.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

The ID3 algorithm was implemented to build a decision tree using information gain. The resulting tree structure accurately classified the dataset based on attribute values.

Ex 10
DATE:

SUPPORT VECTOR MACHINES

AIM:

To Implement and demonstrate the working of a simple Support Vector Machines(SVM) using a dataset

ALGORITHM:

1. **Load the Iris dataset:**
 - Load the dataset into a suitable data structure (e.g., pandas DataFrame).
2. **Preprocess the data:**
 - If necessary, handle missing values, normalize or standardize features, and split the data into training and testing sets.
3. **Train a SVM classifier for each class:**
 - For each class:
 - Create a binary classification problem by considering samples of that class as positive and others as negative.
 - Initialize an SVM classifier with the chosen kernel (linear in this case).
 - Train the classifier on the training set.
4. **Make predictions:**
 - For each test sample:
 - Predict the class probability for each class using the trained SVM classifiers.
 - Assign the sample to the class with the highest probability.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

A Support Vector Machine was trained on the dataset. The classifier was able to separate the classes with an optimal margin, demonstrating effective performance.

Ex 11
DATE:

K-NEAREST NEIGHBOUR ALGORITHM

AIM:

Implement a k-Nearest Neighbour algorithm to classify the iris data set. Print both correct and wrong predictions

ALGORITHM:

1. **Load the dataset:**
 - Load the Iris dataset into a suitable data structure (e.g., pandas DataFrame).
2. **Preprocess the data:**
 - If necessary, handle missing values, normalize or standardize features, and split the data into training and testing sets.
3. **For each test instance:**
 - Calculate Euclidean distance between the test instance and all training instances.
 - Find the k nearest neighbors to the test instance.
 - Determine the most frequent label among the k neighbors.
 - Assign this label as the predicted label for the test instance.
4. **Evaluate predictions:**
 - Compare the predicted labels with the actual labels in the test set.
 - Print correct and wrong predictions.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

The k-Nearest Neighbour algorithm was implemented and tested on the Iris dataset. The model correctly classified most instances and misclassified a few, which were printed for evaluation.

Ex 12
DATE:

MARKET BASKET ANALYSIS USING ASSOCIATION RULES

AIM:

To implement market basket analysis using association rules

ALGORITHM:

1. Load a transaction dataset with minimum support threshold, minimum confidence threshold.
2. Find Frequent itemsets.
 - Start with individual items as frequent itemsets.
 - Generate larger itemsets by joining frequent itemsets.
 - Prune itemsets that do not meet the minimum support threshold.
3. Generate Association Rules.
 - Generate all non-empty subsets.
 - Calculate the confidence of each rule.
 - Keep rules that meet the minimum confidence threshold.
4. Print frequent itemsets and association rules.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

Association rule mining was applied using the Apriori algorithm. Rules with high support, confidence, and lift were generated, revealing strong associations among items.

Ex 14
DATE:

ANN BY SINGLE-LAYER PERCEPTRON.

AIM:

To build an ANN by implementing the Single-layer Perceptron. Test it using appropriate data sets.

ALGORITHM:

1. Initialize parameters:

- weights: Initialize weights randomly (e.g., using a normal distribution).
- bias: Initialize bias randomly.
- learning_rate: Set the learning rate (a hyperparameter).

2. Define activation function:

- Choose an activation function like sigmoid or step function.

3. Training loop:

- For each epoch:
 - For each training example:
 - Calculate the weighted sum of inputs and bias.
 - Apply the activation function to get the predicted output.
 - Calculate the error (difference between predicted and actual output).
 - Update weights and bias using the gradient descent rule:
 - $\text{weights} = \text{weights} + \text{learning_rate} * \text{error} * \text{input}$
 - $\text{bias} = \text{bias} + \text{learning_rate} * \text{error}$

4. Testing:

- For each test example:
 - Calculate the weighted sum of inputs and bias.
 - Apply the activation function to get the predicted output.
 - Compare the predicted output with the actual output to evaluate the model's performance.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

A Single-layer Perceptron was implemented and trained on a linearly separable dataset. The perceptron was able to learn the decision boundary and correctly classify the data.

Ex 15

MULTIPLE-LAYER PERCEPTRON.

DATE:

AIM:

To implement Multi-layer Perceptron and test the same using appropriate data sets

ALGORITHM:

1. Initialization

- Randomly initialize weights w_{ij} and biases b_i for each neuron.

2. Forward Propagation

- Pass input through each layer:

3. Loss Calculation

- Compute the error between predicted and actual outputs using a **loss function**, such as:
 - **Mean Squared Error (MSE)** for regression
 - **Cross-entropy** for classification

4. Backward Propagation

- Use the **chain rule** to compute gradients of the loss with respect to weights and biases.
- Calculate error at the output and propagate it backward through the network.

5. Weight Update

- Update weights and biases using **gradient descent**:

6. Repeat

- Repeat forward and backward propagation for multiple **epochs** until convergence.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

The Multi-layer Perceptron model was successfully built and trained. The model achieved good accuracy on test data using backpropagation for weight updates.

Ex 16
DATE:

RBF NETWORK

AIM:

To Build a RBF Network to calculate the fitness function with five neurons.

ALGORITHM:

Determine the number of input features:

- This depends on the dimensionality of your fitness function's input space.

Choose the RBF function:

- Common RBF functions include Gaussian, Laplacian, and Multiquadric.

Initialize RBF centers:

- Randomly initialize 5 centers in the input space.

Determine RBF widths:

- Use a heuristic or optimization technique to determine the widths of the RBF functions.

Train the output layer:

- Use linear regression or other methods to train the weights between the hidden layer and the output layer.

Problem understanding and Design (3 marks)	
Code Implementation (3 marks)	
Output & Viva Explanation (4 marks)	
Total (10 marks)	

Result:

The Radial Basis Function Network was implemented with Gaussian activation. The model was trained to compute the fitness function, and results showed effective approximation.

Date:

CONTENT BEYOND SYLLABUS DEEP LEARNING WITH TENSORFLOW

AIM:

To implement deep learning using Tensorflow

ALGORITHM:

- **Import libraries:** TensorFlow is imported to implement the model. `tf.keras` is used for high-level deep learning tasks.
- **Load the dataset:** The dataset, such as MNIST or CIFAR-10, is loaded. Both training and testing sets include images and their respective labels.
- **Preprocess the dataset:** The image data is normalized to have pixel values between 0 and 1 to improve the convergence of the model.
- **Define the model:**
 - **Conv2D layer:** Performs convolution with 32 filters and a 3x3 kernel to detect features in the images.
 - **MaxPooling2D layer:** Reduces the spatial size of the feature maps, reducing computational load and focusing on the most prominent features.
 - **Flatten layer:** Converts the 2D matrix to a 1D vector for the fully connected layers.
 - **Dense layers:** Fully connected layers. The final Dense layer has a number of neurons equal to the number of output classes, and it uses the softmax activation function for multi-class classification.
- **Compile the model:** The Adam optimizer is used for optimization, categorical cross-entropy for the loss function, and accuracy is tracked as a metric.
- **Train the model:** The model is trained for 10 epochs with a batch size of 32, using the training images and labels, while validating against the test set.
- **Evaluate the model:** The model's performance is evaluated using the test dataset to get the accuracy.
- **Make predictions:** After training, predictions can be made on unseen images.