Charotar University of Science and Technology [CHARUSAT]

Chandubhai S. Patel Institute of Technology [CSPIT]

Department of Computer Science & Engineering

Date:23/06/2025

Practical List

Academic Year	:	2025-26	Semester	:	5
Course Code	:	CSE303	Course Name	:	Machine Learning

Sr. No.	Aim	Hrs.	CO
1.	**Basics of Pandas**	**2**	**1**
	1.1 Import CSV File • Use pd.read_csv. • Set pandas display options (float format, display width). • Rename 'month_year' to 'measuredate'. • Remove rows with missing 'avgtemp' values. • Display DataFrame info, head, shape, missing values, and summary statistics. **1.2 Import Excel File** • Use pd.read_excel • Rename the 'Year' column to 'metro'. • Strip leading/trailing spaces from the 'metro' column. • Convert year columns (2000-2020) to numeric (float64) using pd.to_numeric with errors='coerce'. • Rename year columns to 'pcGDP' + year (e.g., 'pcGDP2000'). • Remove rows where all year columns (2000-2020) are NaN. • Set 'metro' as the index. • Display DataFrame info, head, shape, dtypes, and summary statistics. **1.3 Import SQL File** • Use pymssql and mysql.connector to connect to SQL Server and MySQL databases. • Execute the provided SQL query to fetch columns from the 'studentmath' table. • Use pd.read_sql to load query results into a DataFrame. • Close database connections. • Rearrange columns as specified in the document. • Set 'studentid' as the index. • Replace coded values in columns (e.g., 'famrel', 'freetime') with descriptive labels. • Convert specified columns to 'category' dtype.		

	• Calculate value counts or percentages for certain columns (e.g., 'famrel', 'freetime'). • Display DataFrame info, head, dtypes, column counts, and unique value counts.		
2.	**Taking Measure of Data through Pandas**	**2**	**1**
	• Load and inspect the nls97 and covidtotals datasets. • Set indices for nls97 (personid) and covidtotals (iso_code). • View column names and data types for both datasets. • Display the first few rows of both datasets. • Check the shape and unique index values of both datasets. • Select specific columns from nls97 using different methods ([], loc, iloc). • Select multiple columns from nls97 based on a list of variables. • Select rows from nls97 using slicing, loc, and iloc. • Filter nls97 rows based on single and multiple conditions (e.g., nightlyhrssleep <= 4 and childathome >= 3). • Convert object columns in nls97 to categorical data type. • Generate frequency distributions for categorical variables in nls97 (e.g., maritalstatus, government responsibility columns). • Save frequency distributions of categorical columns to a text file. • Calculate descriptive statistics for continuous variables in covidtotals (e.g., total_cases, total_deaths). • Compute quantiles for continuous variables in covidtotals. • Create a histogram of total_cases in covidtotals. • Set up PandasAI with covidtotals for natural language queries.		
3.	**Outliers Multivariate**	**2**	**1**
	• Load and inspect the covidtotals and nls97 datasets. • Set indices for covidtotals (iso_code) and nls97 (personid). • Generate descriptive statistics for covidtotals case data. • Calculate quantiles for covidtotals continuous variables. • Compute skewness and kurtosis for covidtotals variables. • Test covidtotals variables for normality using Shapiro-Wilk test. • Create Q-Q plots for total_cases and total_cases_pm in covidtotals. • Identify outliers in covidtotals using interquartile range (IQR) method. • Create a function to detect outliers across multiple covidtotals columns. • Examine outliers for total_deaths_pm with demographic variables. • Plot histograms of total_cases and log-transformed total_cases in covidtotals. • Generate a correlation matrix for covidtotals cumulative and demographic variables. • Categorize covidtotals cases and deaths into quantiles and create a crosstab. • Identify countries with unexpected case-death relationships in covidtotals. • Create scatterplots of total_cases vs. total_deaths and total_cases_pm vs. total_deaths_pm. • Filter nls97 for individuals with wage income but zero weeks worked in 2020. • Check nls97 for individuals ever enrolled in a four-year college.		

	• Identify nls97 individuals with graduate enrollment but no bachelor's enrollment. • Find nls97 individuals with bachelor's or higher degrees but no four-year college enrollment. • Detect nls97 individuals with high wage income (3 standard deviations above mean). • Identify nls97 individuals with significant changes in weeks worked in 2021. • Create a crosstab of nls97 highest grade completed vs. highest degree for grades.		
4.	**Visualization through Matplotlib and Seaborn**	**2**	**1**
	4.1 Histogram of Average Temperatures • Create a histogram of avgtemp from landtemps to visualize the distribution, including mean line, skewness (-0.3856), and kurtosis (2.794). **4.2 Q-Q Plot for Temperature Normality** • Generate a Q-Q plot for landtemps.avgtemp to assess deviation from a normal distribution. **4.3 Stacked Histogram of COVID-19 Cases** • Plot a stacked histogram of total_cases_pm from covidtotals for regions (Oceania/Aus, East Asia, Southern Africa, Western Europe) to compare distributions. **4.4 Boxplot of SAT Verbal Scores** • Create a boxplot of nls97.satverbal to identify outliers, annotating median (500), quartiles (430, 570), and outlier thresholds (220, 780). **4.5 Grouped Boxplots of Weeks Worked** • Use Seaborn to plot boxplots of nls97.weeksworked21 by highestdegree, ordered by degree level, highlighting no lower outliers for some groups. **4.6 Violin Plot of SAT Verbal Scores** • Generate a violin plot of nls97.satverbal to show distribution shape and outliers, annotating quartiles, median, and frequency. **4.7 Scatter Plot of Latitude vs. Temperature** • Plot landtemps.latabs vs. avgtemp, coloring points by elevation (low: blue, high: red) to explore bivariate relationships. **4.8 3D Scatter Plot of Temperature, Latitude, Elevation** • Create a 3D scatter plot of landtemps variables (elevation, latabs, avgtemp) to visualize multivariate relationships. **4.9 Line Plot of COVID-19 Cases and Deaths** • Plot daily new_cases and new_deaths from coviddaily (July 2023–March 2024) to examine trends over time. **4.10 Heat Map of Correlation Matrix** • Generate a heat map of covidtotals correlation matrix, focusing on total_cases, total_deaths, total_cases_pm, total_deaths_pm, using Seaborn's coolwarm cmap. **4.11 Series Operations** • Accessing Values with Bracket Slicing • Selecting Values with loc Accessor • Selecting Values with iloc Accessor		

	• Generating Summary Statistics with describe • Computing Quantiles with quantile • Computing Quantiles with quantile • Complex Conditional Assignment with np.select • Applying User-Defined Functions with apply • String Cleaning with str Methods • Date Parsing and Manipulation with to_datetime		
5.	**Linear Regression**	2	**2**
	Energy Efficiency Estimation for Smart Buildings You are part of a smart infrastructure team working on developing an energy efficiency model for residential buildings. The dataset includes architectural and environmental features like wall area, roof area, glazing area, orientation, relative compactness, and overall height. Your goal is to predict heating load in kilowatts based on these attributes. You must analyze the role of each feature, check for linearity assumptions, and determine whether simple linear approaches suffice or polynomial transformations are necessary. Investigate both underfitting and overfitting scenarios by comparing training and testing errors. You are expected to critically evaluate and justify your modeling decisions using residual plots, correlation matrices, and error metrics. **Key Questions / Analysis / Interpretation to be Evaluated** 1. Conduct exploratory data analysis to determine if linear or nonlinear patterns exist. 2. Select relevant features by analyzing correlation and multicollinearity. 3. Build a predictive model for heating load, interpret its coefficients, and explain each feature's influence. 4. Calculate MAE, MSE, RMSE, and R^2 for both training and test sets. 5. Report on any signs of overfitting. 6. Assess bias and variance using learning curves and validate assumptions via residual plots. 7. Recommend whether polynomial transformations or feature interaction terms may improve performance. **Supplementary Problems** Predict electricity consumption using weather and occupancy data. **Key Skills Addressed** Regression modeling, error interpretation, residual analysis, transformation handling. **Applications** Linear regression, when mastered through practical implementations such as energy load prediction, serves as a foundational technique for numerous real-world tasks involving continuous value estimation. **Learning Outcome** Upon completing this practical: Students will master regression diagnostics, identify modeling pitfalls, and improve estimation accuracy using real-world features. **Dataset/Test Data** Source: UCI Energy Efficiency dataset		

	Link: https://archive.ics.uci.edu/dataset/242/energy+efficiency **Dataset Information** We perform energy analysis using 12 different building shapes simulated in Ecotect. The buildings differ with respect to the glazing area, the glazing area distribution, and the orientation, amongst other parameters. We simulate various settings as functions of the afore-mentioned characteristics to obtain 768 building shapes. The dataset valued responses. It can also be used as a multi-class classification problem if the response is rounded to the nearest integer. **Tools/Technology** Python, pandas, scikit-learn, matplotlib, seaborn.		
6.	**Logistic regression**	2	**2**
	Diagnosing Disease from Symptoms You are working with a health analytics company to develop a predictive system that flags high-risk patients for chronic illness based on demographic, biometric, and lifestyle attributes. Your dataset contains fields like age, BMI, blood pressure, glucose levels, and behavioral flags (e.g., smoking, alcohol). You are tasked with designing a binary classifier that not only predicts the outcome but also justifies its sensitivity to false positives and false negatives under different decision thresholds. Your solution should deal with class imbalance and emphasize the use of probability-based predictions instead of hard labels. **Key Questions / Analysis / Interpretation to be Evaluated** 1. Which features have the highest impact? How is this validated? 2. What does the confusion matrix reveal about the model's classification priorities? 3. How does adjusting the decision threshold influence precision and recall? 4. Why is ROC-AUC a more appropriate evaluation metric than accuracy in this scenario? 5. Is the model biased toward a particular class? How can this bias be detected and addressed? 6. How well does the model generalize across multiple validation folds? **Supplementary Problems** Fraud detection in financial transactions. **Key Skills Addressed** Binary classification, threshold tuning, evaluation metrics (ROC-AUC, F1), cost-sensitive modelling. **Applications** Logistic regression is a fundamental algorithm for binary and multi-class classification tasks, widely adopted in various sectors where decisions are made based on probability-driven outcomes. Its mathematical simplicity, interpretability, and real-time efficiency make it a core skill in the data science and applied machine learning toolkit. **Learning Outcome** Upon completing this practical: Students will build interpretable classifiers, handle imbalanced datasets, and make cost-sensitive decisions **Dataset/Test Data** SOURCE: PIMA Diabetes dataset / Synthetic healthcare dataset		

	Link: https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database **About Dataset** Context This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage. Content The datasets consists of several medical predictor variables and one target variable, Outcome. Predictor variables includes the number of pregnancies the patient has had, their BMI, insulin level, age, and so on. **Tools/Technology** Python, sklearn, imbalanced-learn, seaborn.		
7.	**K-Nearest Neighbors (KNN)**	2	**2**
	Recommending Products Based on Browsing Behaviour A personalized recommendation system is to be developed to predict which product category a user is likely to interact with next, based on their browsing behavior. The dataset contains features such as time spent per category, last purchased product, time since last login, search keywords, and average rating of viewed products. The task involves building a similarity-based model that identifies the top-k closest users or items and classifies the next likely interaction. The model must evaluate multiple distance metrics (e.g., Euclidean, Manhattan, Cosine) and analyze their influence on classification performance. Additionally, KNN should be extended to a regression context to predict user engagement time or next session duration. Scalability concerns, the effect of dimensionality, and the role of normalization in distance-based models must be critically analyzed. **Key Questions / Analysis / Interpretation to be Evaluated** 1. Which distance metric gave the best performance, and what justifies this choice? 2. How does the value of K influence model accuracy and variance? 3. At what K-values does the model exhibit overfitting or underfitting? 4. What insights do misclassified categories provide about potential class overlap? 5. How were the features normalized, and why is normalization critical in this context? 6. What are the scalability limitations of this approach when applied to large datasets? 7. How does the regression model perform (e.g., in predicting session time)? Compare MAE/MSE across K values. **Supplementary Problems** News recommendation, personalized course suggestion. **Key Skills Addressed** Distance-based classification, multi-class confusion matrix, hyperparameter tuning. **Applications** K-Nearest Neighbors (KNN) is a non-parametric, instance-based learning algorithm widely used in both classification and regression tasks. Despite its simplicity, KNN is powerful in applications where pattern similarity, local structure, or case-based reasoning is important.		

	Learning Outcome Upon completing this practical: Students will develop and analyze similarity-based classification systems under practical constraints. **Dataset/Test Data** Source: Retail user activity logs / Synthetic behavior data Link: https://archive.ics.uci.edu/dataset/352/online+retail **Dataset Information** This is a transactional data set which contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail.The company mainly sells unique all-occasion gifts. Many customers of the company are wholesalers. **Tools/Technology** Python, scikit-learn, seaborn, NumPy.		
8.	**Decision Tree & Random Forest**	2	**3**
	Customer Churn Prediction using Decision Tree & Random Forest In today's competitive telecom industry, retaining customers is crucial. Customer churn refers to the loss of clients who stop using a company's services. These practical aims to develop a predictive model using Decision Tree and Random Forest algorithms to identify customers likely to churn based on historical data such as service usage, billing information, and customer support interactions. By analyzing these patterns, the model will help the company proactively address issues and improve retention strategies. The key focus is on building accurate and interpretable models, identifying important features, and minimizing overfitting while ensuring better decision-making support for the business. **Key Questions / Analysis / Interpretation to be Evaluated** 1. What are the most important features influencing customer churn? 2. Can a decision tree model accurately classify churners vs non-churners? 3. Does the Random Forest model reduce overfitting compared to a single decision tree? 4. How does feature importance differ between the models? 5. How does model accuracy change with varying depths or number of trees? **Supplementary Problems** Handling missing or imbalanced data. **Key Skills Addressed** Students will gain hands-on experience in handling data imbalance, building and tuning Decision Tree and Random Forest models, analyzing feature importance, and evaluating models to minimize overfitting **Applications** Students will understand how machine learning can predict customer churn, aiding industries like telecom, banking, and e-commerce in reducing customer attrition through data-driven strategies. **Learning Outcome** Upon completing this practical:		

	Students will learn to preprocess data, build interpretable predictive models, identify key factors driving churn, and validate their models effectively for real-world applications. **Dataset/Test Data** Source: Telco Customer Churn Dataset (Kaggle) Link: https://www.kaggle.com/datasets/blastchar/telco-customer-churn Features include: tenure, MonthlyCharges, Contract, CustomerServiceCalls, PaymentMethod, etc. Target: Churn (Yes/No). **Tools/Technology** Python, pandas, numpy for data handling sklearn for ML models matplotlib, seaborn for visualization Jupyter Notebook or Google Colab.		
9.	**Support Vector Machine (SVM)**	2	**3**
	Email Spam Detection using Support Vector Machine (SVM) With the rapid increase in email usage, spam messages have become a major nuisance, often carrying malicious links or irrelevant promotions. The objective of this practical is to build a Support Vector Machine (SVM) classifier to distinguish between spam and non-spam (ham) emails based on the textual content of emails. The model will analyze word frequencies, patterns, and metadata to identify whether an email is likely spam. By leveraging key SVM concepts such as margin maximization and kernel tricks, the solution aims to deliver high accuracy and generalization while visualizing how data is separated in high-dimensional space. **Key Questions / Analysis / Interpretation to be Evaluated** 1. Can SVM effectively separate spam and non-spam emails based on textual features? 2. How does changing the kernel (linear, polynomial, RBF) affect classification performance? 3. What is the optimal decision boundary, and how is it determined? 4. How does margin maximization improve generalization? 5. What insights can be gained by visualizing high-dimensional data? **Supplementary Problems** Document classification. **Key Skills Addressed** Students will be able to create robust spam detection systems that improve email filtering, enhance cybersecurity, and moderate content, addressing challenges across various domains like communication and social media. **Applications** Students will understand how machine learning can predict customer churn, aiding industries like telecom, banking, and e-commerce in reducing customer attrition through data-driven strategies. **Learning Outcome** Upon completing this practical:		

	Students will develop an understanding of SVM principles, implement text classification pipelines, and interpret the performance and decision boundaries of SVM for high-dimensional data. **Dataset/Test Data** Dataset: SMS Spam Collection Dataset (UCI) Link: https://archive.ics.uci.edu/dataset/228/sms+spam+collection Features extracted from: email/SMS content, frequency of certain keywords, presence of special characters or links. Target: Spam or Ham **Tools/Technology** Python, pandas, numpy for data handling sklearn for ML models matplotlib, seaborn for visualization Jupyter Notebook or Google Colab.		
10.	**Model Evaluation & Cross-Validation**	2	**3**
	Heart Disease Prediction Using Model Evaluation & Cross-Validation Heart disease is one of the leading causes of death globally. Early and accurate prediction of heart disease based on patient health indicators can assist in timely diagnosis and treatment. This practical aims to build and evaluate a classification model (e.g., Logistic Regression, Decision Tree) using various performance metrics such as accuracy, precision, recall, F1-score, and ROC curve. To ensure robustness and generalizability of the model, k-fold cross-validation will be applied. The goal is to not only build a classifier but also interpret and compare the model's performance using different evaluation techniques. **Key Questions / Analysis / Interpretation to be Evaluated** 1. Is the model accurately predicting patients at risk of heart disease? 2. How do performance metrics vary when using different algorithms? 3. Which metric (precision, recall, F1) is more important in a medical context? 4. What does the ROC curve reveal about the model's discrimination ability? 5. How consistent is the model's performance across k-folds? **Supplementary Problems** Quality assurance in manufacturing. **Key Skills Addressed** These practical enables students to evaluate models with metrics like ROC and F1-score, apply k-fold cross-validation for reliability, and compare algorithms to ensure robust predictions. **Applications** Students will see how predictive models can assist in early detection of heart disease, supporting timely medical interventions and improving healthcare decision-making. **Learning Outcome** Upon completing this practical: Students will learn to train and assess classification models, apply evaluation techniques for reliability, and interpret results, preparing them to solve real-world healthcare problems with machine learning. **Dataset/Test Data** Dataset: UCI Heart Disease Dataset Link: https://archive.ics.uci.edu/dataset/45/heart+disease Features include: age, sex,		

	chest pain type, resting blood pressure, cholesterol, fasting blood sugar, etc. Target: Presence or absence of heart disease **Tools/Technology** Python pandas, numpy for data handling sklearn for modeling, evaluation, and cross-validation matplotlib, seaborn for plotting metrics and ROC curves Jupyter Notebook / Google Colab.		
11.	**K-Means and DBSCAN**	2	**4**
	Customer Segmentation Using K-Means and DBSCAN Businesses often deal with large, diverse customer bases. Segmenting customers based on behavior allows companies to tailor marketing strategies, improve customer experience, and optimize resources. These practical aims to apply K-Means and DBSCAN clustering algorithms to a retail customer dataset to group customers with similar purchasing behavior. By analyzing features like annual income and spending score, we will explore clustering techniques, evaluate performance using inertia and silhouette scores, and visualize the clusters. This hands-on practical demonstrates how unsupervised learning can uncover hidden patterns in data without labeled outcomes. **Key Questions / Analysis / Interpretation to be Evaluated** 1. How many natural customer segments are there in the dataset? 2. What is the optimal number of clusters (for K-Means)? 3. How does DBSCAN perform compared to K-Means for density-based clustering? 4. What do inertia and silhouette scores reveal about clustering performance? 5. How can clusters be interpreted for business decision-making? **Supplementary Problems** Image compression and segmentation. **Key Skills Addressed** By completing this practical, students will apply centroid-based and density-based clustering techniques, evaluate cluster quality using inertia and silhouette scores, and visualize data to derive meaningful insights for business. **Applications** Students will learn to segment customers effectively, enabling businesses in retail and e-commerce to design targeted marketing strategies, improve customer satisfaction, and detect anomalies. **Learning Outcome** Upon completing this practical: Students will understand clustering algorithms, evaluate their performance, and interpret clusters to support business decisions with unsupervised learning techniques. **Dataset/Test Data** Dataset: Mall Customer Segmentation Dataset (Kaggle) https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial-in-python Features: CustomerID, Gender, Age, Annual Income, Spending Score Optional: Normalize Annual Income and Spending Score for DBSCAN **Tools/Technology**		

	Python pandas, numpy for data manipulation sklearn for K-Means, DBSCAN, silhouette score matplotlib, seaborn for visualization scipy, PCA from sklearn.decomposition for dimensionality reduction Jupyter Notebook / Google Colab.		
12.	**Principal Component Analysis (PCA)**	2	**4**
	Face Recognition Feature Reduction using PCA Face recognition systems process high-dimensional image data, which increases computational cost and may lead to overfitting. This practical focuses on using Principal Component Analysis (PCA) to reduce the dimensionality of facial image datasets while retaining essential features. The goal is to apply PCA to compress the dataset, visualize it in 2D using principal components, and understand the impact of eigen decomposition and explained variance. This helps in improving model efficiency without significant loss of accuracy. Through this, learners explore how PCA simplifies data while preserving structure and key information. **Key Questions / Analysis / Interpretation to be Evaluated** 1. How does PCA reduce the dimensionality of image data? 2. What percentage of variance is retained by top components? 3. How are eigenvectors and eigenvalues used in PCA? 4. How do lower-dimensional visualizations help in understanding data clusters? 5. What are the trade-offs between compression and information loss? **Supplementary Problems** Handwriting recognition. **Key Skills Addressed** Through this practical, students will standardize high-dimensional data, apply PCA for feature reduction, and analyze explained variance to balance dimensionality and performance. **Applications** Students will explore how dimensionality reduction optimizes face recognition systems and other high-dimensional tasks, reducing computation costs while maintaining essential data for accurate predictions. **Learning Outcome** Upon completing this practical: Students will learn to implement PCA, visualize data in reduced dimensions, and evaluate the trade-offs between data compression and accuracy for real-world applications. **Dataset/Test Data** Dataset: Olivetti Faces Dataset (sklearn) https://scikit-learn.org/stable/datasets/real_world.html#olivetti-faces-dataset Features: Pixel values of grayscale facial images Labels (optional): Person identities (used for classification post PCA) **Tools/Technology** Python numpy, pandas for data handling sklearn.decomposition.PCA for dimensionality reduction matplotlib, seaborn for 2D plotting and visualizations sklearn.datasets for loading image datasets.		

13.	**Artificial Neural Networks (ANN)**	2	**5**
	Handwritten Digit Recognition using Artificial Neural Networks (ANN) Handwritten digit recognition is a classic problem in the field of pattern recognition and deep learning. These practical aims to build an Artificial Neural Network (ANN) using a Multi-Layer Perceptron (MLP) to classify images of handwritten digits from the MNIST dataset. The model will learn from pixel-level image data through forward propagation, adjusting weights using backpropagation, and applying activation functions such as ReLU and Softmax. The objective is to train the ANN to achieve high accuracy in predicting digits (0–9), understand the working of neural layers, and interpret training behavior using loss and accuracy curves. **Key Questions / Analysis / Interpretation to be Evaluated** 1. How does the MLP architecture (hidden layers, neurons) affect performance? 2. What role do activation functions play in learning non-linear patterns? 3. How does the network learn through forward and backward passes? 4. What is the effect of learning rate and number of epochs? 5. How well does the trained ANN generalize to unseen digit images? **Supplementary Problems** Automated postal address reading. **Key Skills Addressed** This practical will involve training an ANN using forward and backward propagation, understanding activation functions, and tuning hyperparameters to optimize model performance **Applications** Students will explore how dimensionality reduction optimizes face recognition systems and other high-dimensional tasks, reducing computation costs while maintaining essential data for accurate predictions. **Learning Outcome** Upon completing this practical: Students will gain the ability to implement and train ANNs, analyze training behavior, and achieve high accuracy in recognizing handwritten digits using modern deep learning frameworks. **Dataset/Test Data** Dataset: MNIST Handwritten Digit Dataset (60,000 training, 10,000 test images) Features: 28x28 grayscale images of digits (0–9) Target: Digit label (0–9) **Tools/Technology** Python TensorFlow or PyTorch for ANN implementation Keras (if using TensorFlow) for high-level APIs matplotlib, seaborn for plotting training metrics Jupyter Notebook / Google Colab for hands-on environment.		
14.	**Convolutional Neural Network (CNN)**	2	**5**
	To implement a Convolutional Neural Network (CNN) for the binary classification of images of dogs and cats. The goal is to build a model that can accurately distinguish between images of dogs and cats. In this case study, we will use the Kaggle Dogs vs. Cats dataset, which consists of 25,000 labeled images of dogs and cats. The dataset can be downloaded from the following link: https://www.kaggle.com/c/dogs-vs-cats/data Dataset:		

• Training set: 25,000 images (12,500 dogs and 12,500 cats)
• Test set: To be split from the training set or use the provided test set for evaluation.

Tasks to be Performed:

1. Dataset Preparation:
   • Download and extract the dataset.
   • Split the dataset into training and validation sets.
   • Preprocess the images (resizing, normalization, and augmentation).
2. Model Building:
   • Define the architecture of the CNN.
   • Choose appropriate layers (convolutional layers, pooling layers, dense layers, etc.).
   • Implement the model using a deep learning framework (e.g., TensorFlow/Keras, PyTorch).
3. Model Training:
   • Compile the model with appropriate loss function and optimizer.
   • Train the model on the training set and validate it on the validation set.
   • Monitor the training process using metrics like accuracy and loss.
4. Model Evaluation:
   • Evaluate the model's performance on the test set.
   • Analyze the results using confusion matrix, precision, recall, and F1-score.
5. Model Optimization:
   • Implement techniques to improve model accuracy (e.g., hyperparameter tuning, regularization, dropout).
   • Retrain and evaluate the optimized model. 6. Model Deployment (Optional):
   • Save the trained model. • Implement a simple application to use the model for real-time image classification.

Key Questions / Analysis / Interpretation to be Evaluated

1. Understanding the Dataset:
   • How is the dataset structured, and what preprocessing steps were necessary?
   • What challenges did you encounter during data preprocessing?
2. Model Architecture:
   • What architecture did you choose for the CNN, and why?
   • How did you decide on the number of layers and their types?
3. Training Process:
   • What loss function and optimizer were used, and why?
   • How did you split the dataset for training and validation?
   • What metrics were used to monitor the training process?
4. Model Performance:
   • What was the accuracy of the model on the validation and test sets?
   • What do the confusion matrix and other evaluation metrics indicate about the model's performance?
5. Optimization Techniques:
   • What optimization techniques did you implement, and how did they affect the model's performance?
   • What were the best hyperparameters found during tuning?
6. Deployment and Application:

	• How can the trained model be deployed for practical use? • What are the potential applications of the model in real-world scenarios? 7. Reflection: • What were the main challenges you faced during the implementation of the CNN? • How would you improve the project if given more time or resources? **Supplementary Problems** 3 class problem **Key Skills Addressed** Training and evaluating deep learning models Visualizing and interpreting model performance Performing hyperparameter tuning (epochs, learning rate, batch size) Applications Image Classification **Learning Outcome** Implement an CNN using modern frameworks like Keras or PyTorch. Apply forward and backward propagation effectively during training. Analyze training curves and optimize model parameters. Evaluate and interpret classification results on real-world datasets. Dataset/Test Data The dataset can be downloaded from the following link: https://www.kaggle.com/c/dogs-vs-cats/data Dataset: • Training set: 25,000 images (12,500 dogs and 12,500 cats) • Test set: To be split from the training set or use the provided test set for evaluation. **Tools/Technology** Keras, matplotlib		
14.	**Q-Learning algorithm** To implement the Q-Learning algorithm for solving a reinforcement learning problem and evaluate its performance. Tasks to be Performed 1. Setup Environment: • Install the OpenAI Gym toolkit. • Load the Frozen Lake environment. • Understand the environment's state and action space. 2. Implement Q-Learning Algorithm: • Initialize the Q-table with zeros. • Define the hyperparameters: learning rate (alpha), discount factor (gamma), and exploration rate (epsilon). • Implement the Q-Learning update rule. • Implement an epsilon-greedy policy for action selection. 3. Training the Agent: • Run the Q-Learning algorithm for a fixed number of episodes. • Update the Q-values based on the experiences gained by the agent. • Monitor the agent's performance over time. 4. Evaluation: • Test the trained Q-Learning agent in the environment.	2	**5**

• Measure the agent's performance using metrics such as average reward per episode.

5. Analysis:
   • Analyze how different hyperparameters affect the learning process.
   • Compare the performance of the Q-Learning agent with other baseline methods if available.

Key Questions / Analysis / Interpretation to be Evaluated

1. How does the Q-Learning algorithm update the Q-values?
2. What is the role of the learning rate (alpha) and how does it affect the learning process?
3. How does the discount factor (gamma) influence the Q-Learning algorithm?
4. What is the epsilon-greedy policy and why is it used in Q-Learning?
5. How does the exploration rate (epsilon) impact the agent's learning and performance?
6. What challenges did you encounter while implementing the Q-Learning algorithm and how did you address them?
7. How does the performance of the Q-Learning agent change with different values of alpha, gamma, and epsilon?
8. Can the Q-Learning algorithm be applied to other environments? If yes, what changes would be required in the implementation?
9. Discuss the importance of choosing the right hyper parameters for the Q-Learning algorithm.
10. How does the Q-Learning algorithm compare to other reinforcement learning algorithms?
11. Analyse the reward value for the different actions.

Supplementary Problems

Frozen lake problem

Key Skills Addressed

Implementation of Q-Learning algorithm Hyperparameter tuning (α, γ, ε) Policy design using epsilon-greedy strategy

Applications

Spam detection
News topic classification
Social media content moderation
Customer feedback analysis

Learning Outcome

Students will understand and implement a tabular Q-Learning algorithm Students will analyze how learning parameters affect agent performance Students will develop the ability to design, train, and evaluate RL agents

Dataset/Test Data

Environment: Frozen Lake v1 from OpenAI Gym Predefined 4x4 or 8x8 grid map with slippery surface, holes, and a goal No external dataset; environment provides states, actions, and rewards dynamically.

Prepared By:	Prof. Dharmendrasinh Rathod, Dr. Amit Thakkar and Dr. Jigar Sarda