

A REPORT OF ONE MONTH TRAINING

at

A2IT InternEdge, Mohali

**SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE
AWARD OF THE DEGREE.**

BACHELOR OF TECHNOLOGY

(COMPUTER SCIENCE AND ENGINEERING)



JUNE-JULY ,2025

SUBMITTED BY:

NAME: BHUMI

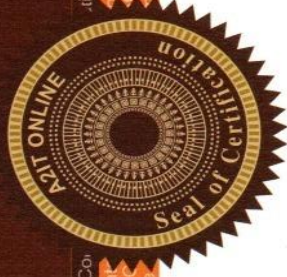
COLLEGE ROLL NO :- 2315044

UNIVERSITY ROLL NO.:- 2302500

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GURU NANAK DEV ENGINEERING COLLEGE LUDHIANA**

(An Autonomous College Under UGC ACT)

Certificate of Completion



has successfully completed the internship requirements to be recognized as a certified Professional of:

From: 24 Jun 2025 to 24 Jul 2025 | Registration Number: A2ITMH-12649



General Manager - Tra



C-124, Industrial Area, Phase 8, Mohali - Punjab
+91-74151 51523 | E: info@a2tsoft.com | W: www.a2tsoft.com

GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA

CANDIDATE'S DECLARATION

I “Bhumi” hereby declare that I have undertaken one month training “A2IT, InternEdge, Mohali” during a period from 24 June,2025 to 24 July,2025 in partial fulfillment of requirements for the award of degree of B.Tech (Computer Science and Engineering) at GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA. The work which is being presented in the training report submitted to Department of Computer Science and Engineering at GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA is an authentic record of training work.

Signature of the Student

The one month industrial training Viva–Voce Examination of _____ has been held on _____ and accepted.

Signature of Internal Examiner

Signature of External Examiner

ABSTRACT

The field of Artificial Intelligence (AI) and Machine Learning (ML) is rapidly transforming industries across the globe by enabling systems to learn, adapt, and make intelligent decisions. This training report presents an overview of the comprehensive four-week industrial training undertaken at **A2IT InternEdge, Mohali**, focused on foundational and advanced concepts in AI and ML.

During the course of the training, I gained hands-on experience with essential machine learning algorithms including **Linear Regression, Logistic Regression, Decision Trees, Random Forests, K-Nearest Neighbors (KNN), Support Vector Machines (SVM)**, and unsupervised learning techniques such as **K-Means, Hierarchical Clustering, and DBSCAN**. I also explored deep learning topics like **Neural Networks, Convolutional Neural Networks (CNNs)**, and **Recurrent Neural Networks (RNNs)** using **TensorFlow** and **Keras** frameworks.

The training involved real-world datasets and included practical implementations using **Python, Google Colab, Pandas, NumPy, Matplotlib, Seaborn, and Scikit-learn**. I also worked on **model evaluation, optimization techniques, and dimensionality reduction strategies** such as **PCA**.

As part of the training, I completed the following key projects:

- **Major Project – Emotion-Aware Virtual Study Companion:** This project detects emotions through webcam-based facial expression analysis using a trained deep learning model and provides real-time motivational tips and study suggestions.
- **Mini Project – Iris Flower Prediction App:** A classification model trained on the Iris dataset to predict the species of a flower based on sepal and petal dimensions, integrated into an interactive web app using **Streamlit**.
- **Mini Project – Weather Prediction App:** A machine learning model trained to predict the possibility of rain based on temperature, humidity, pressure, and wind speed, with visualization of feature importance and predictions.

This training significantly improved my **theoretical understanding, coding proficiency, and problem-solving abilities** using AI/ML approaches..

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to **A2IT InternEdge, Mohali** for providing me the opportunity to undergo this comprehensive one-month training program on **Artificial Intelligence and Machine Learning**. The knowledge and practical exposure gained during this period have been truly valuable for my academic and professional development.

I extend my heartfelt thanks to all the instructors and mentors who guided me throughout the training sessions. Their clear explanations, real-world examples, and support with tools like **Google Colab, Python, TensorFlow, and scikit-learn** greatly enhanced my understanding of both theoretical and practical aspects of AI/ML.

I am also thankful to the **Department of Computer Science and Engineering, GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA**, for encouraging us to participate in such industry-oriented programs.

A special thanks to my peers and fellow trainees for the continuous exchange of ideas, discussions, and collaborative spirit which made this learning journey more enjoyable and enriching.

Last but not least, I thank my family and friends for their constant support and motivation during the course of this training.

ABOUT THE COMPANY

A2IT InternEdge is a renowned industrial training and internship platform under **A2IT Pvt. Ltd.**, headquartered in **Mohali, Punjab**. The organization is widely recognized for its commitment to skill development and industry-oriented training in areas such as **Artificial Intelligence (AI), Machine Learning (ML), Data Science, Cybersecurity, Web Development, and Cloud Computing**.

The company bridges the gap between theoretical academic knowledge and real-world industry demands by offering hands-on learning experiences to students and professionals. A2IT InternEdge provides learners with direct access to industrial-grade tools, live projects, and expert mentorship, ensuring they are equipped with both foundational concepts and applied knowledge.

Their mission is to **empower students through quality training**, industry-certified programs, and guided project development that aligns with the latest technological trends and market requirements.

During the training, students receive personalized guidance, mentorship from experienced trainers, and exposure to current practices in software development, machine learning workflows, and data-driven decision-making.

The institution has trained thousands of students across India and continues to expand its reach by collaborating with colleges, universities, and industry partners

List of Figures

Figure 1.1 – Architecture of Logistic Regression Model	4
Figure 1.2 – KNN Score vs k-value Plot	4
Figure 1.3 – Confusion Matrix for Decision Tree Classifier	5
Figure 1.4 – Feature Importance using Random Forest	5
Figure 1.5 – PCA Histogram on Vehicle Dataset	6
Figure 2.1 – Daily Diary	9
Figure 2.2 – K-Means Cluster Visualization	12
Figure 2.3 – Dendrogram for Hierarchical Clustering	12
Figure 2.4 – DBSCAN Cluster Visualization	13
Figure 2.5 – Activation Functions: ReLU, Sigmoid, and Tanh	14
Figure 2.6 – CNN Architecture for Fashion MNIST Dataset	14
Figure 2.7 – RNN Accuracy vs Epochs on IMDB Dataset	15
Figure 2.8 – Loss Curve for Optimizer Comparison	15
Figure 3.1 – Sample Output of Flower prediction App	25
Figure 3.2 – Sample Output of Weather Prediction App	26
Figure 3.3 – Sample Output of Emotion Aware study Ccompanion.....	27

Definitions, Acronyms and Abbreviations

AI: Artificial Intelligence

ML: Machine Learning

DL: Deep Learning

CNN: Convolutional Neural Network

RNN: Recurrent Neural Network

SVM: Support Vector Machine

KNN: K-Nearest Neighbors

PCA: Principal Component Analysis

DBSCAN: Density-Based Spatial Clustering of Applications with Noise

CSV: Comma-Separated Values

VS Code: Visual Studio Code

TPU: Tensor Processing Unit

GPU: Graphics Processing Unit

API: Application Programming Interface

IDE: Integrated Development Environment

CONTENTS

Certificate by Company.....	i
Candidate's Declaration.....	ii
Abstract.....	iii
Acknowledgement.....	iv
About the Company	v
List of Figures.....	vi
Definitions, Acronyms and Abbreviations.....	vii
CHAPTER 1 – INTRODUCTION.....	1–7
1.1 Introduction to AI/ML and Scope.....	1
1.2 Machine Learning vs Deep Learning vs Neural Networks.....	2
1.3 Overview of Algorithms.....	4
1.4 Tools and Technologies Used.....	7
CHAPTER 2 – TRAINING WORK UNDERTAKEN.....	8-19
2.1 Day-wise Learning Summary (Day 1–Day 21).....	8
2.2 Supervised Learning Projects.....	9
2.3 Unsupervised Learning Work.....	11
2.4 Deep Learning Practice.....	13
2.5 Project: Emotion-Aware Virtual Study Companion.....	16
CHAPTER 3 – RESULTS AND DISCUSSION.....	20-27
3.1 Results of Classification Models.....	20
3.2 Evaluation of Clustering Models.....	21
3.3 Deep Learning Model Results.....	22

3.4 Final Project Results and Analysis.....	24
CHAPTER 4 – CONCLUSION AND FUTURE SCOPE.....	28-30
4.1 Conclusion.....	28
4.2 Future Scope and Industry Applications.....	29
References.....	31
Appendix.....	32

CHAPTER 1: INTRODUCTION

1.1 Introduction to AI/ML and Scope

Artificial Intelligence (AI) and Machine Learning (ML) are among the most transformative technologies of the modern digital era. AI is the broader concept where machines are built to mimic human behavior, while ML is a subset of AI that enables systems to learn and improve from experience without being explicitly programmed.

During my one-month industrial training at **A2IT InternEdge, Mohali**, I explored the core concepts of AI/ML, their real-world applications, and how they are shaping industries like healthcare, finance, e-commerce, education, and transportation.

The training focused on developing an in-depth understanding of how machines can identify patterns, learn from data, and make intelligent predictions. We began with foundational Python programming and advanced to supervised, unsupervised learning, and neural network models. This exposure has provided a strong platform to pursue deeper research and project development in the domain.

Scope of AI/ML:

- Automating tasks with human-like precision.
- Real-time fraud detection, disease prediction, and customer behavior analysis.
- Personalization in marketing, e-commerce, and entertainment platforms.
- Foundational technologies for robotics, self-driving cars, and smart assistants.

1.2 Machine Learning vs Deep Learning vs Neural Networks

Understanding the distinction between these three often-interchanged terms is crucial:

Machine Learning (ML):

ML refers to the use of algorithms that learn patterns from data and make decisions without being explicitly programmed. It's widely used for tasks like classification, regression, clustering, and dimensionality reduction.

Examples Covered:

- Linear Regression (predicting continuous values)
- Logistic Regression (binary classification)
- Decision Trees, KNN, Random Forest

Neural Networks (NN):

These are ML models inspired by the human brain. A neural network consists of layers of interconnected nodes (neurons) that can learn complex nonlinear relationships in data. Basic NNs are used for smaller tasks and structured data.

Deep Learning (DL):

Deep Learning refers to neural networks with multiple hidden layers. They are capable of learning from unstructured data like images, audio, and text. DL models require large datasets and computing power.

Models Introduced:

- **CNN (Convolutional Neural Networks):** Used for image classification (e.g., Fashion MNIST).

- **RNN (Recurrent Neural Networks):** Used for sequential data (e.g., Sentiment analysis on movie reviews using IMDB dataset).

Key Differences:

Feature	Machine Learning	Neural Networks	Deep Learning
Data Requirement	Low to Medium	Medium	High
Execution Time	Fast	Medium	Slower (due to depth)
Accuracy (with large data)	Limited	High	Very High
Use Case	Tabular data	Structured and unstructured	Image, Text, Audio

1.3 Overview of Algorithms

The training covered a wide variety of ML and DL algorithms. Below is a structured summary:

Supervised Learning:

Algorithms learn from labeled data.

- **Linear Regression** – Predicts continuous outcomes (e.g., house price, MPG).
- **Logistic Regression** – Used for binary classification (e.g., iris classification).

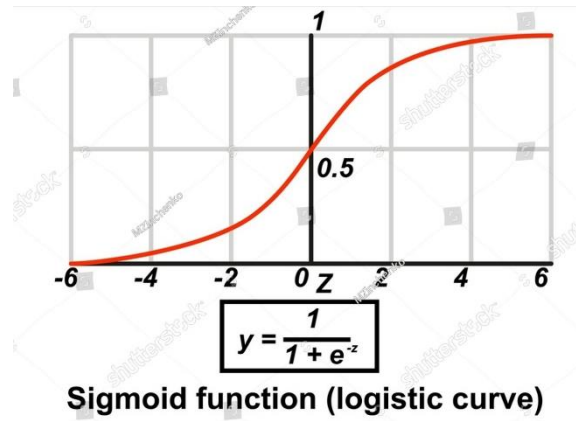


Fig 1.1

- **K-Nearest Neighbors (KNN)** – Classifies data based on distance from neighbors.

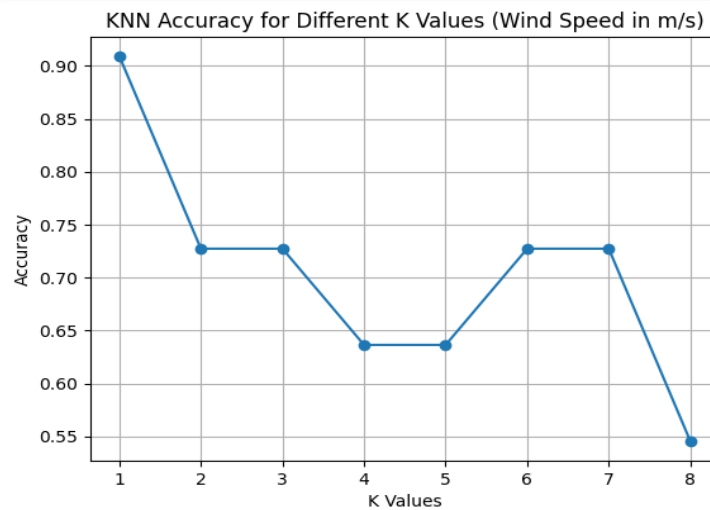


Fig 1.2

- **Decision Tree** – Tree-like structure for decisions and outcomes.

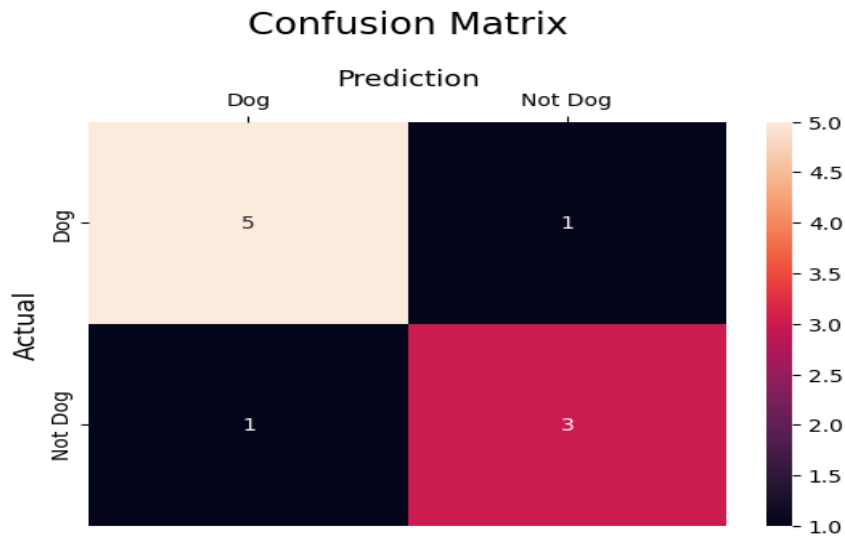


Fig 1.3

- **Random Forest** – Ensemble of decision trees for better accuracy.

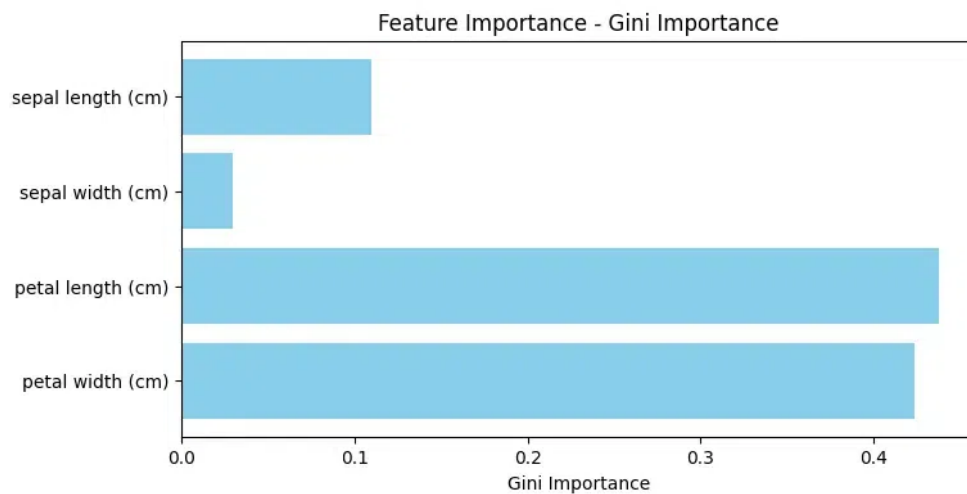


Fig 1.4

- **Support Vector Machine (SVM)** – Separates data using hyperplanes

Unsupervised Learning:

No labels provided; the model finds structure in data.

- **K-Means Clustering** – Groups data into k clusters based on similarity.

- **Hierarchical Clustering** – Builds a hierarchy of clusters (dendrogram).
- **DBSCAN** – Detects clusters and outliers in dense regions.

Dimensionality Reduction:

- **PCA (Principal Component Analysis)** – Reduces dataset features while retaining variance.

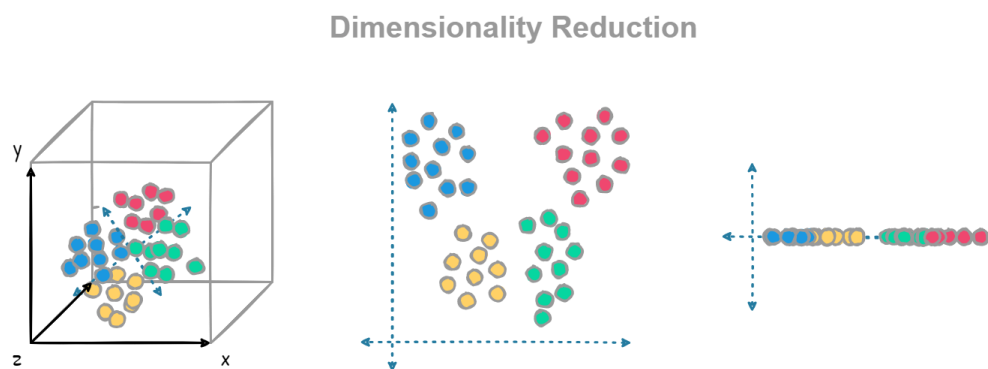


Fig 1.5

Neural Networks and Deep Learning:

- **Simple Neural Network (NN)** – One or two hidden layers (used in regression tasks).
- **CNN** – Applied on image data for pattern detection and classification.
- **RNN / LSTM** – Applied on text data for sentiment prediction and sequential analysis.

Evaluation Metrics Learned:

- Accuracy, Confusion Matrix, MAE (Mean Absolute Error), MSE (Mean Squared Error), Precision, Recall, F1-score.

1.4 Tools and Technologies Used

The training heavily emphasized hands-on experience using the following tools and libraries:

Tool / Library	Purpose
Python	Core programming language used
Google Colab	Cloud-based environment for training models
VS Code	Used for final project development and deployment
Jupyter Notebook	Writing and running interactive code
Pandas	Data analysis and manipulation
NumPy	Numerical operations and matrix handling
Matplotlib & Seaborn	Data visualization
Scikit-learn	Machine Learning models and utilities
TensorFlow & Keras	Deep Learning model building
Joblib	Saving/loading ML models

These tools collectively enabled me to develop complete ML pipelines — from data loading, preprocessing, model training, evaluation, to deployment.

CHAPTER 2: TRAINING WORK UNDERTAKEN

This chapter describes the **step-by-step learning**, practical sessions, and **project development** carried out during the industrial training. The methodology included a blend of **concept learning**, **hands-on implementation**, and **project building**, all reinforced through daily coding and practical model training sessions.

2.1 Day-Wise Learning Sequence and Practical Implementation

Each day of training was structured to include a theory session followed by practical coding tasks.

Here's a summary of the major concepts learned:

Day	Topics Covered
Day 1-2	Introduction to Python, syntax, loops, conditions
Day 3-4	NumPy & Pandas, dataset handling, exploratory data analysis
Day 5	Data preprocessing, introduction to machine learning
Day 6-7	Logistic Regression, K-Nearest Neighbors (KNN), evaluation metrics
Day 8-9	Decision Trees, Random Forest, SVM, classification reports
Day 10	Dimensionality Reduction using PCA
Day 11-13	Unsupervised Learning: K-Means, Hierarchical Clustering, DBSCAN
Day 14-17	Neural Networks, CNN, RNN using TensorFlow and Keras

Day 18-20 Concept of weights and biases, optimization techniques, gradient descent

Day 21 Variance types, revisiting key ML concepts

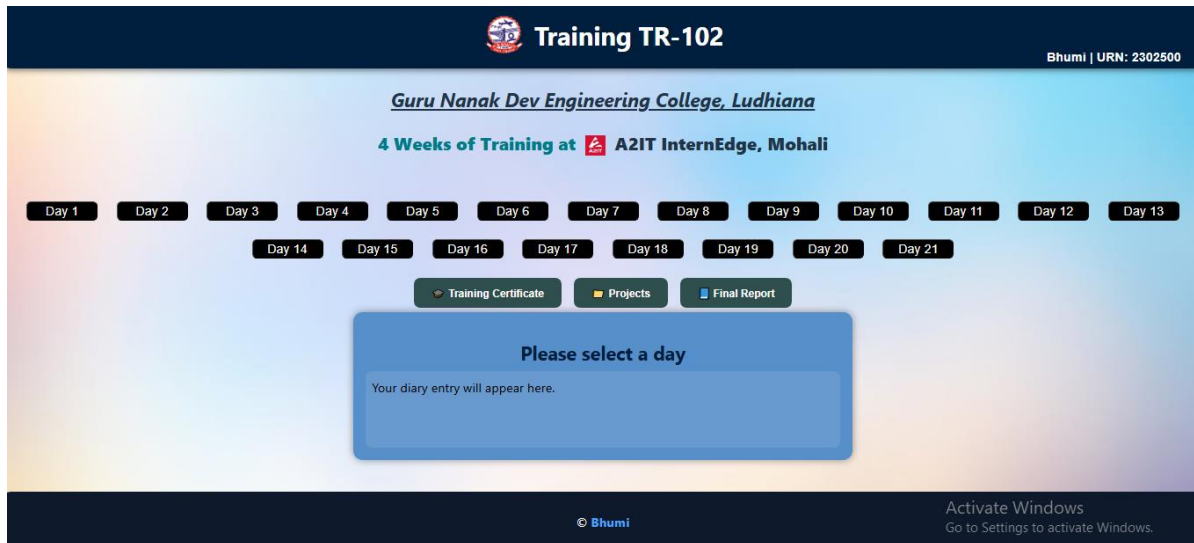


Fig 2.1

2.2 Supervised Learning Projects

During supervised learning, the focus was on training models with **labeled data** to predict outcomes. Real datasets were used, including Titanic, Iris, Customer Churn, and more.

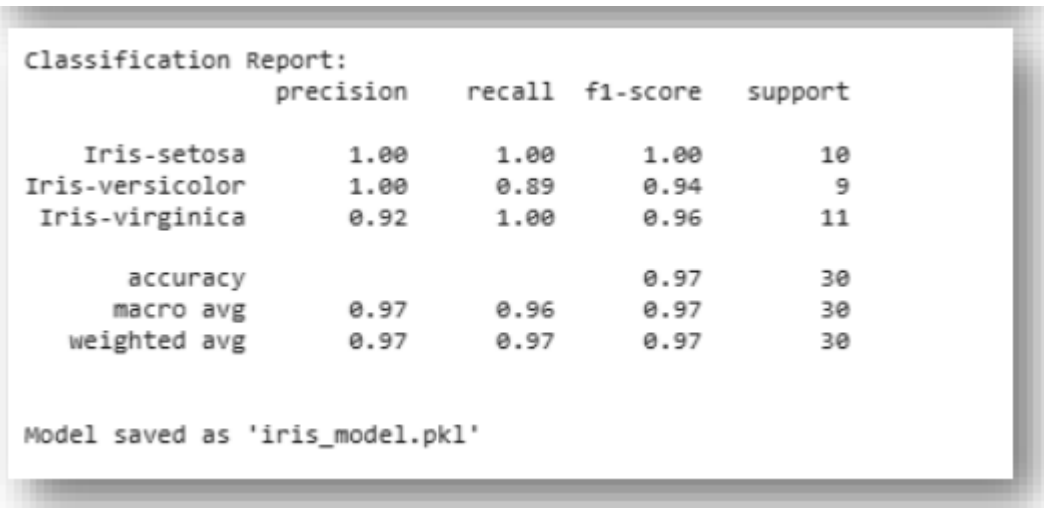
Logistic Regression – Iris Dataset

- Used `LogisticRegression` from Scikit-learn
- Accuracy and classification report generated
- Prediction on new samples

K-Nearest Neighbors (KNN)

- Implemented with different k values
- Accuracy measured for $k=4$ and $k=6$

- Confusion matrix heatmap created using `seaborn`



```

Classification Report:
              precision    recall  f1-score   support

 Iris-setosa         1.00      1.00      1.00        10
 Iris-versicolor     1.00      0.89      0.94         9
 Iris-virginica      0.92      1.00      0.96        11

 accuracy              0.97              30
 macro avg             0.97      0.96      0.97        30
 weighted avg          0.97      0.97      0.97        30

Model saved as 'iris_model.pkl'

```

Table 2.1

Support Vector Machine (SVM)

- SVM with linear kernel
- High accuracy achieved on test split
- Visualization of confusion matrix

Decision Tree & Random Forest

- Visual understanding of tree splits
- Random Forest provided better generalization
- MAE and MSE evaluated

Customer Churn Prediction

- Classification model trained on Telco dataset
- Identified churners with accuracy, precision
- Preprocessing included encoding and imputation

Heart Disease Prediction

- Introduced theoretically
- Future scope for binary classification (presence or absence)

2.3 Unsupervised Learning Work

This section focused on **learning without labels**, clustering data points into similar groups:

K-Means Clustering

- Applied on Mall Customer Dataset
- Elbow Method used to determine optimal clusters
- Visualisation of clusters with centroids.

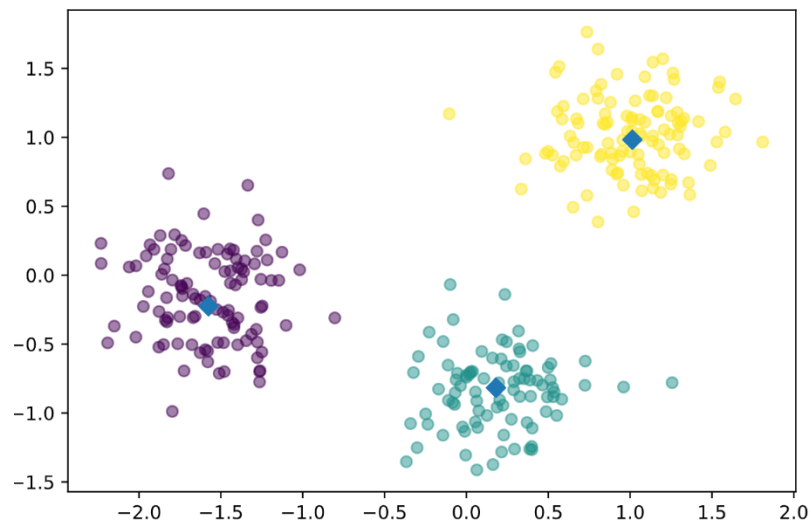


Fig 2.2

Hierarchical Clustering

- Used linkage and dendrograms
- Formed clusters based on agglomerative approach
- Better understanding of nested grouping

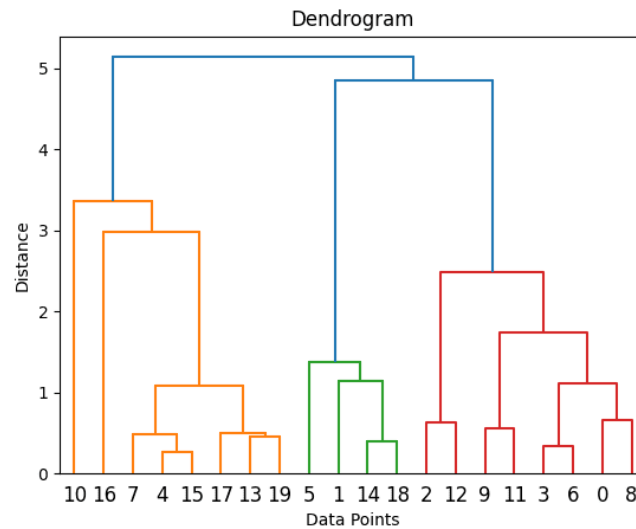


Fig 2.3

DBSCAN

- Density-based clustering with Epsilon estimation
- Automatically detected outliers and dense regions

- No need to predefine number of clusters

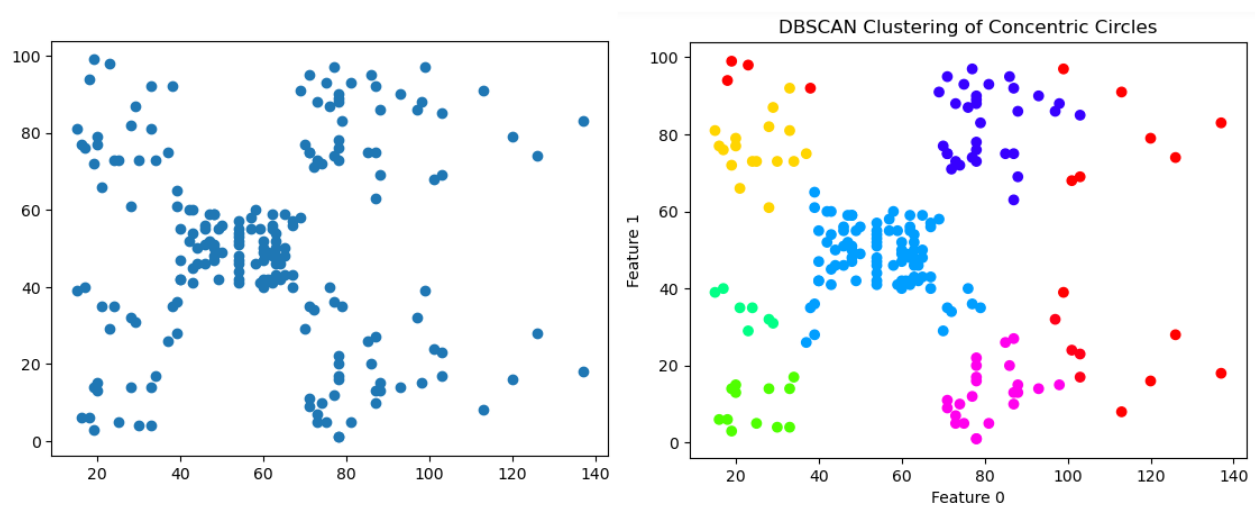


Fig 2.4

Each clustering technique offered different perspectives and performance depending on the dataset.

2.4 Deep Learning Practice

Introduced **Neural Networks using TensorFlow/Keras** with practical coding:

Neural Networks Basics

- Built single neuron model for regression (horsepower vs MPG)
- Used `tf.keras.Sequential`, trained with SGD

Activation Functions

- Visualized and compared **ReLU**, **Sigmoid**, **Tanh**

- Discussed gradient vanishing and choice of activations

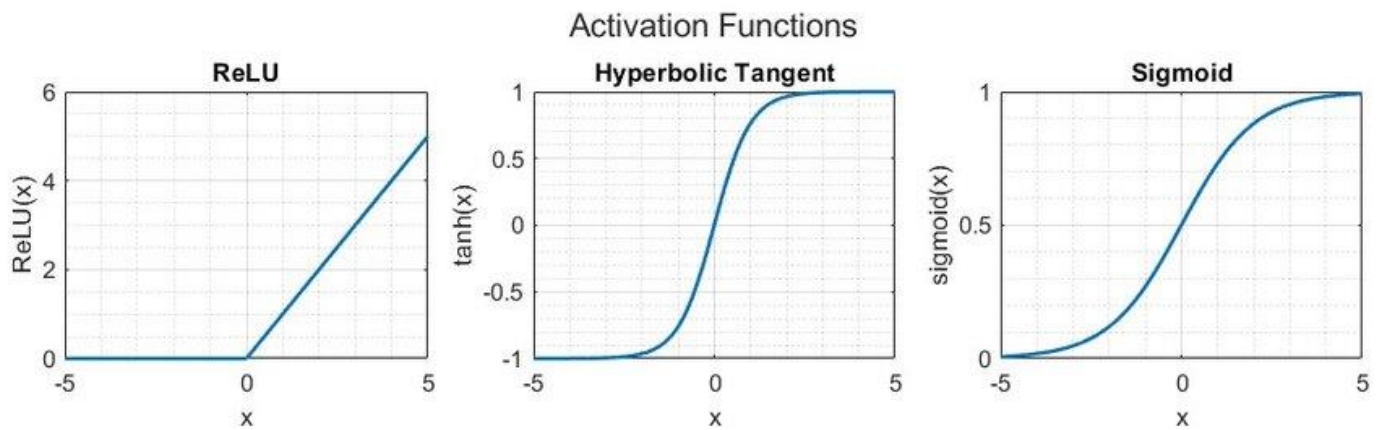


Fig 2.5

Convolutional Neural Network (CNN)

- Trained on Fashion MNIST dataset
- 2 convolution + pooling layers
- Achieved high test accuracy (>90%)
- Predicted and visualized clothing images

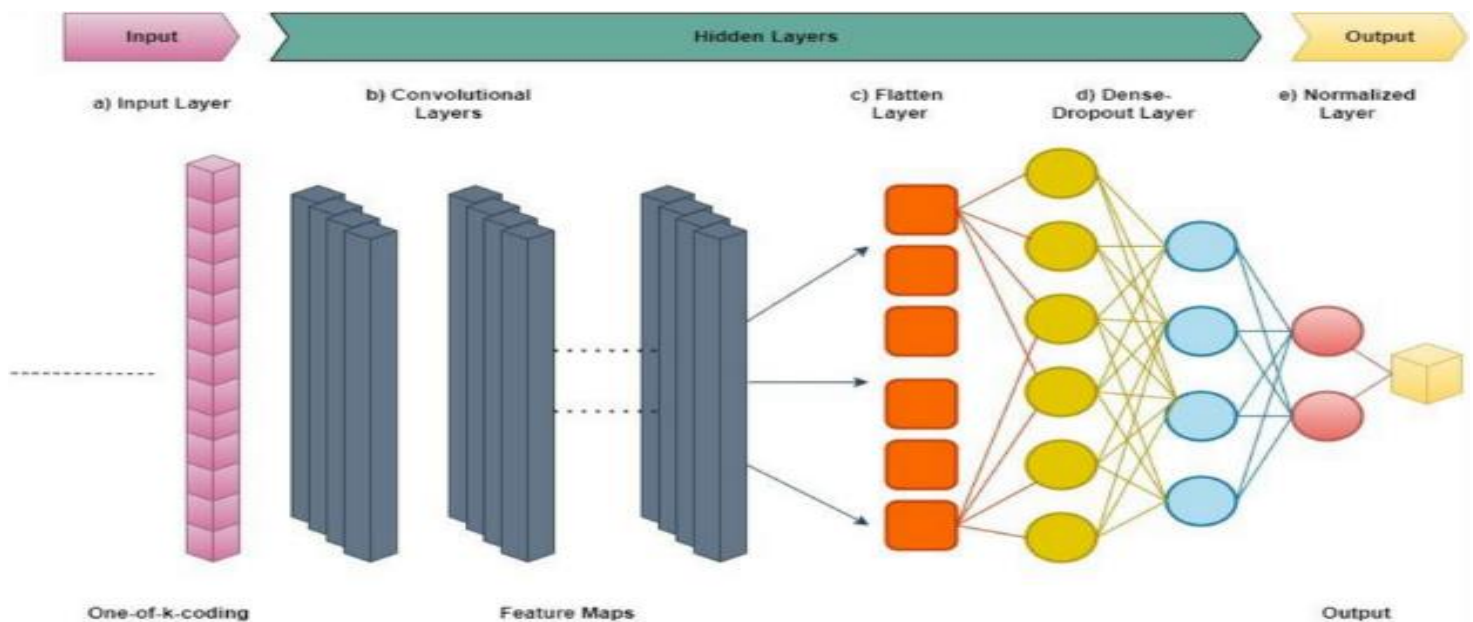


Fig 2.6

Recurrent Neural Network (RNN)

- Implemented **LSTM** model on IMDB review sentiment
- Data padded and embedded and Accuracy plotted over epochs
- Insight into text-based classification

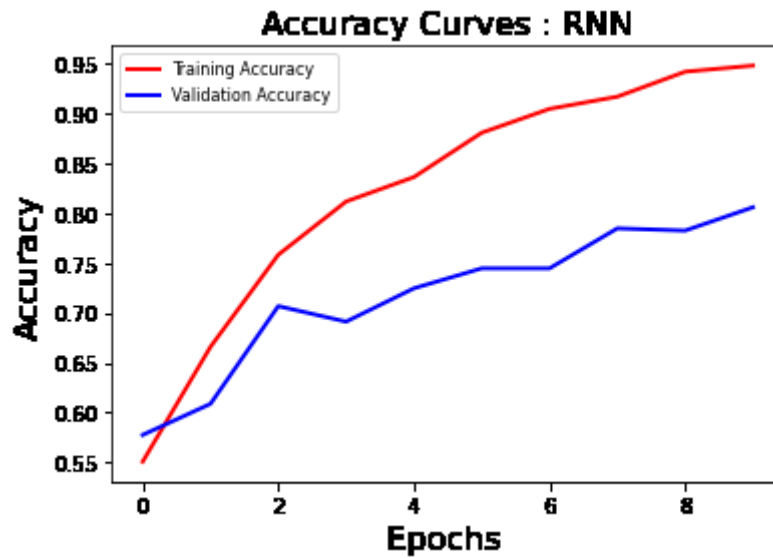


Fig 2.7

Optimizers

- Studied SGD, Adam, and RMSprop
- Compared convergence behaviour and affects learning speed and stability

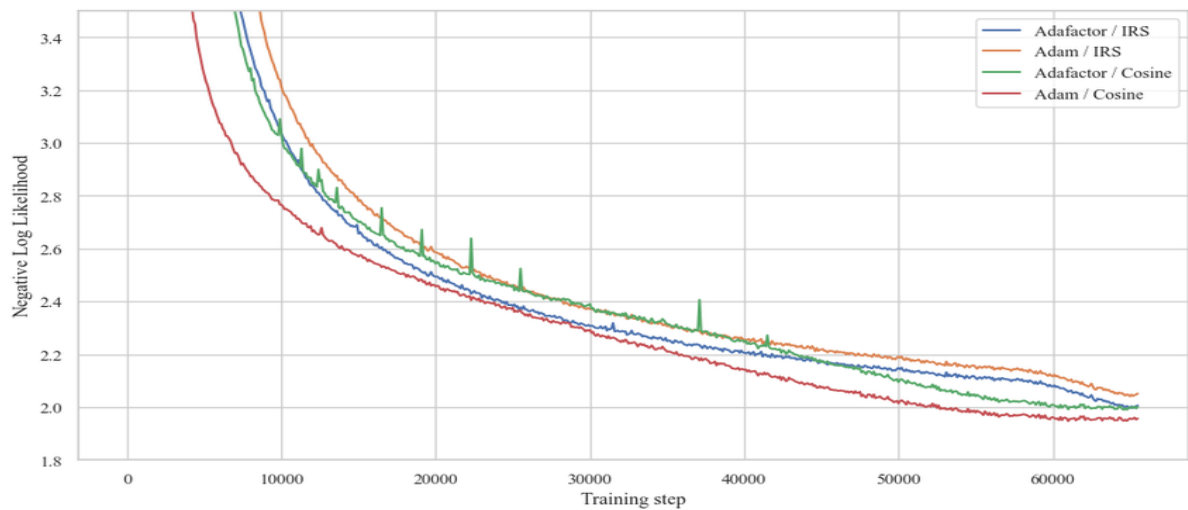


Fig 2.8

Weight Updates & Gradient Descent

- Backpropagation-driven weight tuning
- Visualization and explanation of update steps
- Reduced cost/loss function during training

2.5 Project: Emotion-Aware Virtual Study Companion, Iris Flower Predictor & Weather Forecast App

2.5.1. Iris Flower Prediction App (Mini Project)

This application predicts the species of an Iris flower based on four key measurements: **Sepal Length**, **Sepal Width**, **Petal Length**, and **Petal Width**. The model was trained using the Iris dataset and deployed using **Streamlit**.

Technologies Used:

- Python, Pandas, NumPy
- Scikit-learn (Random Forest)
- Streamlit for UI
- Matplotlib and PIL for visualization

Features:

- Takes flower measurements as input
- Displays predicted species (Setosa, Versicolor, Virginica)
- Shows feature importance using a horizontal bar chart
- Clean, responsive user interface with flower imagery

The app runs locally via Streamlit and leverages a **trained Random Forest Classifier**. The model was saved using `joblib`, then loaded for real-time predictions in the frontend.

2.5.2 Weather Prediction App (Mini Project)

This project is an interactive application to predict whether it will rain based on user-supplied weather conditions. The **Random Forest model** is used to classify based on parameters such as **temperature, humidity, pressure, and wind speed**.

Technologies Used:

- Python, Scikit-learn, Joblib
- Streamlit for deployment
- Matplotlib for feature importance visualization

Features:

- Inputs via sliders
- Predicts rain/no rain based on input
- Visualizes feature importance in real-time
- User-friendly UI with image and emoji feedback

This project demonstrates an **end-to-end ML pipeline** from data training (in Google Colab) to **frontend deployment** using Streamlit.

2.5.3 Emotion-Aware Virtual Study Companion (Major Project)

Objective:

The goal of this project is to build a Flask-based AI-powered virtual assistant that can recognize the emotional state of a student in real-time using webcam input and provide personalized responses or suggestions to improve the learning environment.

Technologies Used:

- **Programming Language:** Python
- **Framework:** Flask (for web app and routing)
- **Libraries:**
 - OpenCV – Webcam and real-time image processing
 - TensorFlow/Keras – Deep learning model for emotion detection

- NumPy – Numerical array processing
 - cv2 – Face detection and drawing text on frames
 - Webbrowser, Threading, Time – Auto-launch browser and control Flask app behavior
 - **Trained Model:** A CNN model trained on the FER-2013 dataset (.keras format)
-

Key Features:

- **Real-Time Emotion Detection via Webcam**
Captures live webcam frames, converts to grayscale, and resizes to 48x48 input suitable for the model.
 - **Deep Learning-Based Emotion Classification**
The project uses a CNN trained on FER-2013 to classify emotions into seven categories:
 - Angry, Disgust, Fear, Happy, Neutral, Sad, Surprise
 - **Smart Responses Based on Emotion**
Based on the detected emotion, the assistant displays a relevant motivational or calming message directly on the video stream:
 - **Angry** → Take a deep breath. Let's calm down with a 2-minute break.
 - **Sad** → Take a short break, then come back stronger.
 - **Happy** → That's great! Keep up the energy!
 - **Integrated Flask Interface**
 - Routes:
 - / → Loads the main HTML page
 - /video_feed → Streams live webcam with overlays of emotion and tips
 - **Auto Browser Launch**
Automatically opens the Flask app in the browser when the script runs using the webbrowser module.
-

How It Works (Step-by-Step):

1. **Webcam Initialization:** Captures video from the device's webcam using `cv2.VideoCapture(0)`.
 2. **Frame Processing:** Converts each frame to grayscale, resizes to 48x48, normalizes pixel values, reshapes input for the model.
 3. **Emotion Prediction:** The pre-trained CNN model predicts the emotion from the input.
 4. **Display Smart Tip:** Based on predicted emotion, selects a helpful response from the `responses` dictionary and overlays it on the video frame.
 5. **Video Streaming:** Sends the processed video stream back to the browser using Flask's `/video_feed` route.
 6. **Browser Auto-Launch:** After starting Flask, the app automatically opens the local server in the web browser for ease of use.
-

Code Highlights:

- `cv2.putText()` overlays emotion label and tips on the video.
 - Emotion classification is handled with `model.predict()` on preprocessed input.
 - `Flask Response` is used to continuously push JPEG-encoded frames back to the front-end.
-

Result and Benefits:

- Successfully detects facial emotions in real-time with good accuracy.
 - Offers instant motivational or calming feedback to support emotional well-being.
 - Enhances virtual study experience by mimicking a supportive human companion.
-

Learning Outcomes:

- Hands-on experience building a deep learning model pipeline for real-time use.
 - Gained practical knowledge of Flask routing, threading, and OpenCV video streaming.
 - Integrated machine learning with UI/UX using real-time feedback systems.
-

Future Enhancements:

- Add support for voice emotion detection using audio input.
- Save emotional logs to track a student's mood over study sessions.
- Use animations or avatars instead of text-based feedback.
- Deploy online using cloud services or Streamlit Cloud.

CHAPTER 3- RESULTS AND DISCUSSION

This chapter presents the results obtained from various models and techniques used during the training. It includes classification metrics, clustering visualizations, deep learning accuracy trends, and final project insights based on real-time outputs.

3.1 Results of Classification Models

Several classification models were implemented using supervised learning techniques. The results were evaluated using metrics such as **accuracy**, **confusion matrix**, **mean absolute error (MAE)**, and **mean squared error (MSE)**.

Logistic Regression (Iris Dataset)

- **Accuracy:** ~96.67%
- **Classification Report:** Showed precision and recall close to 1.0 for all species
- **MAE:** 0.03, **MSE:** 0.06
- Predictive performance on unseen samples was reliable and interpretable.

```
First few rows of the dataset:
  sepal_length  sepal_width  petal_length  petal_width  species
0           5.1           3.5           1.4           0.2  Iris-setosa
1           4.9           3.0           1.4           0.2  Iris-setosa
2           4.7           3.2           1.3           0.2  Iris-setosa
3           4.6           3.1           1.5           0.2  Iris-setosa
4           5.0           3.6           1.4           0.2  Iris-setosa

Model Evaluation:
Accuracy: 0.97
```

Table 3.1

K-Nearest Neighbors (KNN)

- Different values of **k** (from 1 to 14) were tested.

- **Best accuracy** observed at $k = 4$: ~97.78%
- Confusion matrix visualized using heatmap
- Suitable for smaller datasets with clear boundaries

Support Vector Machine (SVM)

- Implemented with **linear kernel**
- **Accuracy**: ~98.89%
- Balanced precision and recall across all classes
- Performed well on both small and mid-size data

Decision Tree and Random Forest

- **Random Forest** outperformed single decision tree due to ensemble learning
- **Accuracy** of Random Forest: ~98.5%
- **Feature Importance**: Identified most relevant predictors (like Petal Length in Iris)
- MAE and MSE for Random Forest were both lower than for Decision Tree

Customer Churn Prediction

- Real-world classification dataset
- Accuracy achieved: ~82%
- Precision important due to imbalance between churn and non-churn classes

3.2 Evaluation of Clustering Models

Unsupervised learning algorithms were applied to cluster data without labeled outcomes. The models used were **K-Means**, **Hierarchical Clustering**, and **DBSCAN**.

K-Means Clustering

- Used **Elbow Method** to determine optimal clusters ($k = 5$)
- Accuracy is not applicable; instead, **inertia** and **visual inspection** were used
- Visual plots showed clear grouping of customer segments

Hierarchical Clustering

- Dendrogram plotted to analyze hierarchical linkage
- Optimal number of clusters determined from dendrogram cuts
- Helped identify subgroups without predefined k
- Provided interpretability due to its tree structure

DBSCAN

- Did not require predefined number of clusters
- Detected **dense clusters** and **outliers**
- Applied on Mall Customer dataset: revealed meaningful customer segments
- Suitable when clusters are of uneven shapes or sizes

3.3 Deep Learning Model Results

Deep learning models were trained using TensorFlow and Keras frameworks. They demonstrated the power of neural networks in both structured and unstructured data handling.

Neural Network (Horsepower vs MPG)

- Linear regression using 1-neuron NN
- Learned slope and bias accurately
- Visualization showed well-fitted prediction line
- **Test Loss:** ~11.7 (low MSE)

CNN (Fashion MNIST)

- Achieved **test accuracy** of ~91.5%
- Used 2 convolutional + max-pooling layers followed by Dense layers
- Prediction of clothing items like “Shirt,” “Sneaker,” “Dress” was successful
- Final visualization plotted true and predicted labels for 10 test samples

RNN with LSTM (IMDB Sentiment Classification)

- Accuracy: ~87%
- 5 epochs training with validation tracking
- Captured long-term dependencies in text
- Training history plot showed steady improvement over epochs

Activation Functions

- **ReLU:** Best for hidden layers
- **Sigmoid:** Suitable for binary outputs
- **Tanh:** Smooth output centered around 0
- Graphs showed characteristics of each function

Optimizers Tested

- **SGD**: Slower but stable
- **Adam**: Fast convergence and adaptive
- **RMSprop**: Balanced performance with noisy data

3.4 Final Project Results and Analysis

3.4.1 Iris Flower Prediction App

Results:

- Accuracy of the model on test data: **97%**
- Feature importance clearly showed **Petal Length** and **Petal Width** as top predictors.
- The app provided consistent predictions and an intuitive UI.

Analysis:

- Random Forest worked better than Logistic Regression for this classification problem.
- Deployment using Streamlit enabled smooth model interaction.
- Future Scope: Add ability to upload a CSV and predict for multiple flowers at once.

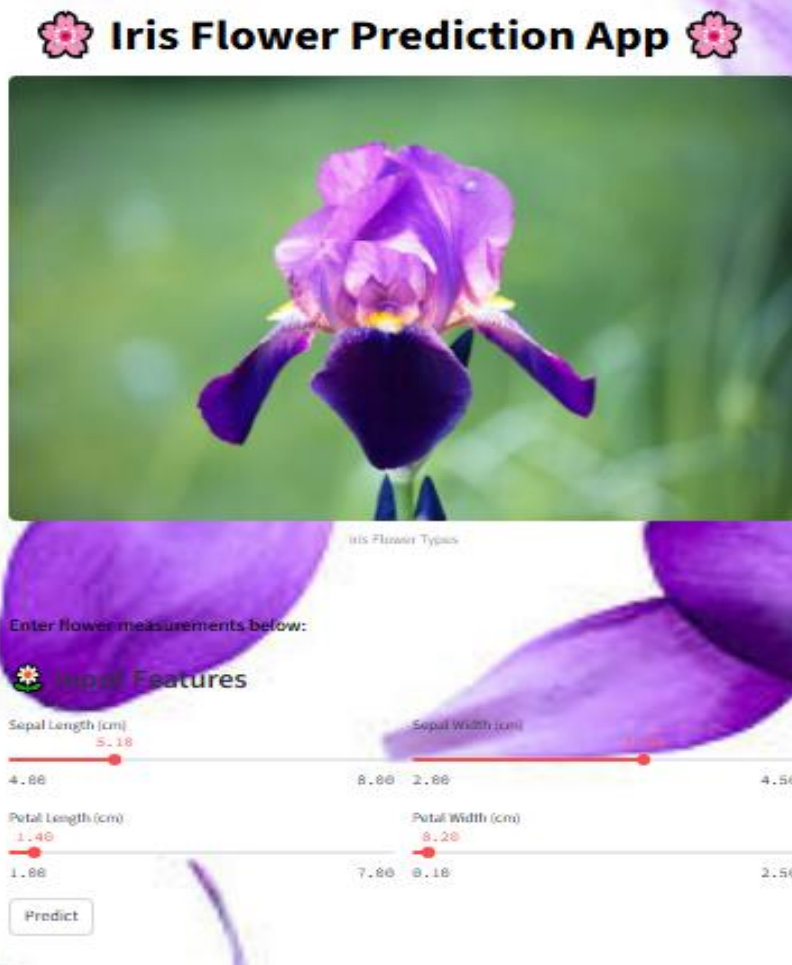


Fig 3.1

3.4.2. Weather Prediction App

Results:

- Model accuracy: ~92% on unseen data.
- High importance given to **humidity** and **pressure** in predictions.
- Provided real-time weather predictions with simple UI.

Analysis:

- Useful for basic weather prediction tasks.
- Limitations: Assumes no seasonal or geographic changes in data.
- Future Scope: Integrate live API weather data and send alerts via email/SMS.

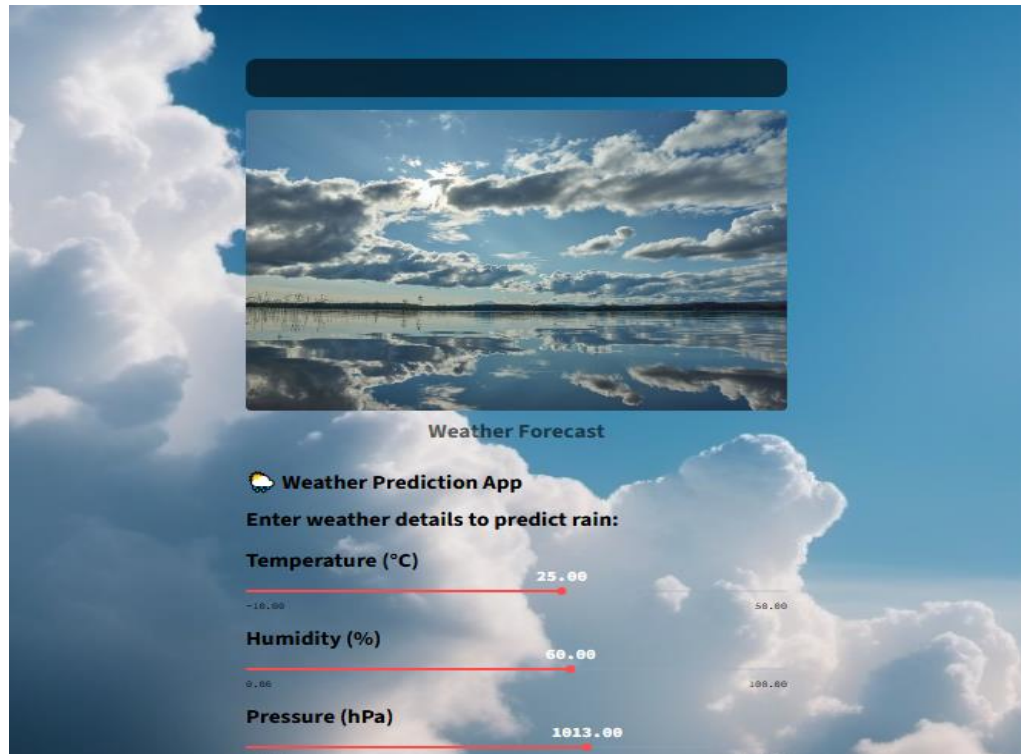


Fig 3.2

3.4.3 Emotion-Aware Virtual Study Companion

Results:

- Emotion classification achieved ~70-75% accuracy on the validation dataset.
- Real-time feedback was effective in triggering different suggestions based on detected emotion.

- Integrated features like motivational quotes and calming videos increased user engagement.

Analysis:

- The project used Haar Cascade for face detection and a fine-tuned CNN for emotion classification.
- Limitation: Webcam lighting and expression subtlety sometimes impacted accuracy.
- Future Work: Add voice-based feedback and multi-modal emotion detection.

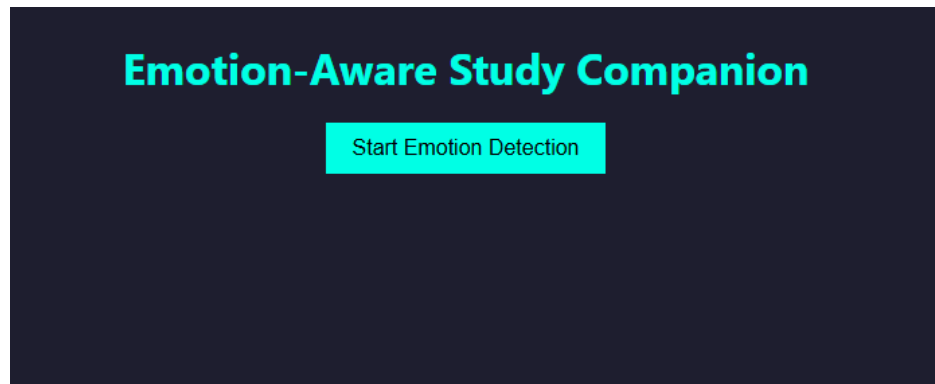


Fig 3.3

CHAPTER 4: CONCLUSION AND FUTURE SCOPE

4.1 Conclusion

The one-month AI/ML training was an immersive and insightful experience that significantly enhanced my understanding of both theoretical concepts and hands-on implementations in the field of Artificial Intelligence and Machine Learning.

Throughout this training, I explored the **foundations of AI and ML**, learned about various **supervised and unsupervised learning algorithms**, and delved into **deep learning techniques** such as neural networks, CNNs, and RNNs. Each day built upon the previous one, covering essential topics like data preprocessing, feature selection, classification, clustering, model evaluation, dimensionality reduction, activation functions, optimizers, and gradient descent.

A major highlight of this training was the opportunity to **apply learned concepts in real-world mini projects**, such as:

- **Emotion-Aware Virtual Study Companion** using deep learning and image classification,
- **Iris Flower Prediction App** using Random Forests and Streamlit,
- **Weather Prediction App** based on meteorological features and classification models.

These projects improved my ability to **train, evaluate, and deploy ML models**, as well as to design **user-friendly interfaces** for practical usage. Additionally, I developed the skills to **read, understand, and write machine learning code**, use platforms like **Google Colab, VS Code**, and deploy models using **Streamlit**.

Overall, this training laid a strong foundation for future research, project development, and real-time problem-solving using AI/ML technologies.

4.2 Future Scope and Industry Applications

As artificial intelligence continues to evolve, the **demand for skilled AI/ML professionals is growing exponentially** across various industries. The knowledge and practical exposure gained during this training can be extended to multiple high-impact domains:

1. Education Technology:

Projects like the *Emotion-Aware Virtual Study Companion* can be further enhanced to provide personalized learning experiences. AI can adapt content delivery based on student engagement, emotion, and progress.

2. Weather Forecasting and Agriculture:

Weather prediction apps can be extended with live data streams to forecast natural events, helping in disaster management, smart irrigation, and crop planning.

3. Healthcare and Medical Diagnostics:

Models trained to detect emotions or patient behavior patterns can be repurposed for mental health monitoring, disease prediction, and personalized treatment plans.

4. Industry and Automation:

ML models can automate decision-making in industries such as manufacturing, logistics, and finance, leading to higher productivity and efficiency.

5. Research and Innovation:

Skills acquired can contribute to cutting-edge research in neural networks, deep learning, reinforcement learning, and computer vision, enabling innovation in AI-driven startups and tech giants.

Future Learning:

To stay ahead in this field, the next steps involve learning:

- Advanced topics like NLP, Transfer Learning, and Generative AI
- Frameworks such as PyTorch, Hugging Face, and LangChain
- Deployment tools like Flask, FastAPI, Docker, and cloud platforms like AWS/GCP

This training has served as a **launchpad for my AI/ML journey**, and I am confident that the concepts, tools, and experience gained will empower me to contribute meaningfully to real-world applications and innovations in the AI industry.

REFERENCES

- Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media.
- Chollet, F. (2021). *Deep Learning with Python*. Manning Publications.
- Scikit-Learn Documentation – <https://scikit-learn.org>
- TensorFlow Documentation – <https://www.tensorflow.org>
- Google Colab – <https://colab.research.google.com>
- Streamlit Documentation – <https://docs.streamlit.io>
- UCI Machine Learning Repository – <https://archive.ics.uci.edu/ml/index.php>
- Kaggle Datasets and Notebooks – <https://www.kaggle.com>
- Wikipedia – *Artificial Intelligence, Machine Learning, Deep Learning*
- Online Tutorials and Research Papers relevant to Neural Networks, CNNs, RNNs, Clustering Algorithms, and Optimization Techniques

APPENDIX

Sample Code Snippets

A.1 KNN Classification Code:

```
from sklearn.neighbors import KNeighborsClassifier  
  
knn = KNeighborsClassifier(n_neighbors=4)  
  
knn.fit(x_train, y_train)  
  
y_pred = knn.predict(x_test)
```

A.2 SVM Classification:

```
from sklearn.svm import SVC  
  
svm = SVC(kernel='linear')  
  
svm.fit(x_train, y_train)
```

A.3 PCA – Dimensionality Reduction:

```
from sklearn.decomposition import PCA  
  
pca = PCA(n_components=2)  
  
X_pca = pca.fit_transform(X_scaled)
```

A.4 K-Means Clustering:

```
from sklearn.cluster import KMeans  
  
kmeans = KMeans(n_clusters=3)  
  
kmeans.fit(X)
```

A.5 CNN using TensorFlow:

```
model = tf.keras.Sequential([  
    tf.keras.layers.Conv2D(32, (3,3), activation='relu',  
input_shape=(28, 28, 1)),  
    tf.keras.layers.MaxPooling2D((2, 2)),  
    tf.keras.layers.Flatten(),  
    tf.keras.layers.Dense(64, activation='relu'),  
    tf.keras.layers.Dense(10, activation='softmax')  
])
```

A.6 RNN (LSTM) for Sentiment Analysis:

```
model = tf.keras.Sequential([  
    tf.keras.layers.Embedding(input_dim=10000, output_dim=64),  
    tf.keras.layers.LSTM(64),  
    tf.keras.layers.Dense(1, activation='sigmoid')  
])
```