**INDUSTRY INTERNSHIP REPORT**

On

# “Data Science Using Python”



An Internship report submitted in partial fulfillment of requirements for the award of degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

**By**

**G. SWAROOPA RANI (219X1A0530)**

**Under the esteemed guidance of**

**Sri R. Sandeep Kumar**

**Assistant Professor**

**Department of C.S.E.**

**Department of Computer Science and Engineering**

**G. PULLA REDDY ENGINEERING COLLEGE (Autonomous): KURNOOL**

**(Affiliated to JNTUA, ANANTHAPURAMU)**

**KURNOOL – 518007**

**2024 – 2025**

**Department of Computer Science and Engineering**

**G. PULLA REDDY ENGINEERING COLLEGE (Autonomous): KURNOOL**

**(Affiliated to JNTUA, ANANTHAPURAMU)**



**CERTIFICATE**

**This is to certify that the internship titled** ‘Data Science using Python’ **is a bonafide record of work carried out by** SWAROOPA RANI (219X1A0530) **and the Internship report is submitted in partial fulfillment of requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering.**

…………………………….. . ……………………………..

Internship Guide HEAD OF THE DEPARTMENT

Sri R. Sandeep Kumar Dr. N. Kasiviswanath

Assistant Professor Professor & Head of the Department,

Department of C.S.E., Department of C.S.E.,

G. Pulla Reddy Engineering College, G. Pulla Reddy Engineering College,

Kurnool. Kurnool.

**INTERNSHIP DETAILS**

**Name of the College :** G. Pulla Reddy Engineering College

**Department :** Computer Science and Engineering

**Name of the Student :** G. Swaroopa Rani

**Register Number :** 219X1A0530

**Name of the Faculty Guide :** SriR. Sandeep Kumar

**Duration of the Internship : From** 10-01-2025 **to** 10-05-2025

**Name and address of the Organization :** Ava Intern, Neraluru, Bangalore,

Karnataka

**Internship Domain :** Machine Learning and AI

**Internship Title :**  Data Science using Python

**Internship Stipend :** No Stipend

**Date of Internship Report Submission :** 30-04-2025

**Internship Coordinator**

**Dr. B. Geetha Vani**

**Professor**

**CSE Department**

**GPREC**

### DECLARATION

I **G Swaroopa Rani**, a student bearing the Roll No. **219X1A0530**, of the Department of Computer Science and Engineering, G. Pulla Reddy Engineering College, do hereby declare that the I have completed the mandatory internship titled **Data Science using Python** from **10-01-2025** to **10-05-2025** in **Ava Intern, Bangalore** under the Faculty Guide ship of **Sri R. Sandeep Kumar.**

**(Student Signature and Date)**

## Certificate from Organization

This is to certify that **G. Swaroopa Rani**, **219X1A0530** of **G. Pulla Reddy Engineering College** underwent Internship in **Ava Intern** from **10-01-2025** to **10-05-2025.**

The overall performance of the Intern during her internship is found to be good.



Authorized Signatory with Date and Seal

**Internship completion certificate**

****

**ACKNOWLEDGEMENT**

I wish to express my deep sense of gratitude to my internal supervisor **Sri R. Sandeep Kumar**, **Assistant Professor** of Computer Science and Engineering Department, G. Pulla Reddy Engineering College.

My sincere thanks to our Internship Coordinator **Dr. B. Geetha Vani** for her immaculate guidance, constant encouragement and cooperation which have made possible to bring out this internship work.

My sincere thanks to Miss. **Vollala Sindhu**, **Technical Industrial Mentor,** of Ava Intern, for helping me and giving me the required information needed for my internship work.

I am thankful to our Head of the Department **Dr. N. Kasiviswanath**, for his whole hearted support and encouragement during the internship.

We are grateful to our Internship Dean **Dr. Y. V. Siva Reddy** and to our respected Principal **Dr. B. Sreenivasa Reddy** for providing requisite facilities and helping us in providing such a good environment.

I wish to convey my acknowledgements to all the staff members of the Computer Science and Engineering Department for giving the required information needed for my internship work.

Finally, I wish to thank all our friends and well-wishers who have helped me directly or indirectly during the course of this internship work.

**Table of Contents**

Page No

**1**: **INTRODUCTION** **1**

1.1 About the Internship Company/Organization 2

1.2 Virtual/Offline Internship details 5

**2**: **WORK DONE DURING INTERNSHIP** **8**

2.1 About Course/Domain 9

2.2 Technologies learnt during internship 17

2.3 Assessments/Tasks assigned details 20

**3**: **PROJECTS/MODULES COMPLETED** **22**

3.1 About Project 1 23

3.2 About Project 2 34

**4**: **ACTIVITY LOG** **46**

**5**: **CONCLUSION 52**

**6: REFERENCES 55**

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **FIGURE NAME** | **PAGE NO** |
| 1. | Home page of Ava Intern website | 7 |
| 2. | Contact Page of Slash Mark website | 7 |
| 3. | Data Science Life Cycle | 9 |
| 4. | Flowchart of project (gender and age detection) | 22 |
| 5. | Output of project 1 (Female gender detection) | 31 |
| 6. | Output of project 1 (Male gender detection) | 31 |
| 7. | Flowchart of project (movie recommendation system) | 34 |
| 8. | Results of the project 2 commands | 39 – 4439 39 - 44 |
| 9. | Output of project 2 | 45 |

**1. INTRODUCTION**

**1. INTRODUCTION**

**1.1 About the Internship Company/Organization:**

​**Ava Intern** is an online education and career development platform based in Bengaluru, India, established in 2023. Operated by Ava intern Edu tech Private Limited, the company focuses on providing students and early-career professionals with hands-on, project-based internships and training programs to enhance industry readiness

Ava Intern is devoted to enhancing engineering education and life skills, and we are unwavering in our commitment to making quality education more affordable and accessible to all.

**Career growth and learning:**

At Ava Intern, we understand that career growth and continuous learning are the cornerstones of personal and professional development. Our company is not just a workplace; it's a dynamic learning environment designed to empower you to achieve your full potential. We believe that your journey with us should be a transformative experience. From day one, you'll have access to a wealth of resources, mentorship, and opportunities to expand your knowledge and skills. Whether you're just starting your career or seeking to advance to the next level, we provide the tools and support to help you reach your goals. Our commitment to your growth goes beyond the ordinary. We offer a range of professional development programs, workshops, and training opportunities to keep you at the forefront of your field. Additionally, our culture encourages innovation, creativity, and collaboration, providing the ideal backdrop for you to thrive and make a meaningful impact. At Ava Intern, your success is our success. Join us on a journey of continuous learning, career advancement, and personal growth, and let's shape a brighter future together.

### Core Offerings

* **Project-Based Internships**: Ava Intern offers internships that emphasize practical experience, aiming to bridge the gap between academic learning and industry requirements.​
* **Online Courses**: The platform provides a variety of online courses, including programs like Business Development Analyst (BDA), which cover topics such as business fundamentals, market analysis, and research methodologies.
* **Career Services**: Beyond training, Ava Intern offers career counseling, coaching, and placement services to support individuals in achieving their professional goals.

**Courses**:

​Ava Intern offers a diverse range of online and offline courses designed to equip students and professionals with practical, industry-relevant skills. Here's an overview of their course offerings:​

### Technical Courses

* **Full Stack Web Development**: Covers front-end and back-end development, including HTML, CSS, JavaScript, and server-side technologies.​
* **Data Structures & Algorithms (DSA)**: Focuses on fundamental concepts essential for coding interviews and software development.​
* **Data Science**: Teaches data analysis, machine learning, and statistical modeling techniques.​
* **Java Programming**: Offers in-depth knowledge of Java for building robust applications.​
* **Cloud Computing**: Provides insights into cloud service models, deployment, and management strategies.
* **VLSI Design**: Introduces Very-Large-Scale Integration design principles and methodologies. ​
* **Embedded Systems**: Covers the development and testing of embedded hardware and software systems. ​
* **UI/UX Design**: Focuses on user interface and user experience design for digital products. ​
* **Cybersecurity**: Teaches strategies to protect systems and networks from cyber threats.​
* **Artificial Intelligence (AI)**: Explores AI concepts, including machine learning and neural networks.​
* **Software Testing**: Covers testing methodologies to ensure software quality and reliability.​
* **AutoCAD**: Provides training in computer-aided design for engineering and architectural applications.​

### Non-Technical Courses

* **Digital Marketing**: Teaches online marketing strategies, SEO, and social media management.​
* **Human Resources (HR)**: Covers recruitment, employee relations, and HR management practices.​
* **Finance**: Focuses on financial analysis, investment strategies, and risk management. ​
* **Business Development Executive (BDE)**: Trains individuals in sales strategies, client acquisition, and market analysis.​
* **Business Development Analyst (BDA)**: Emphasizes analytical skills for market research and business strategy.​

**Additional Information**:

**Official Website**: <https://www.avaintern.com>

**Course Page** (Internship Domains): <https://www.avaintern.com/courses>

**Ava Intern – LinkedIn**: <https://www.linkedin.com/company/ava-intern>

**Login Page:** <https://www.avainternlms.in/login>

**1.2 Offline Internship details:**

## Training Phase

The internship began with a well-structured training phase designed to build a strong foundation in data-related tools and concepts. This phase combined theoretical learning with practical exposure, ensuring a balanced and hands-on learning experience. I was introduced to fundamental topics in data analysis, data visualization, and machine learning. The training sessions were interactive and included regular assignments, and mini tasks that reinforced the concepts taught. Through continuous practice and guidance from industry mentors, I was able to solidify my understanding and develop confidence in handling data and solving analytical problems. This foundational stage was crucial in preparing me for the challenges and responsibilities of the upcoming project phase.

## Project phase

Following the training phase, I transitioned into the project phase where I applied my acquired knowledge to real-world datasets. Under the mentorship of experienced professionals, I worked on practical problems involving various stages of the data pipeline.

This included tasks such as data cleaning, preprocessing, exploratory data analysis, and drawing insights from patterns in data. I also gained experience in modeling and evaluating data to solve specific business challenges. This phase not only improved my technical skills but also strengthened my problem-solving, communication, and collaboration abilities. The hands-on experience gave me a clear understanding of how data-driven decisions are made in a professional environment, making it a highly valuable part of the internship.

After completing the learning phase, candidates get the opportunity to work on four live projects that are based on real-world scenarios. These projects are carefully designed to reinforce the concepts learned during training and allow candidates to apply their knowledge in a practical setting. Each project challenges different aspects of data handling, analysis, and visualization, providing hands-on experience that is essential for professional growth.

Working on these live projects not only helps in understanding how to tackle real-time problems but also builds confidence in using tools and techniques effectively. With the guidance of mentors, candidates learn how to clean and analyze data, draw meaningful insights, and present their findings clearly. These projects serve as a bridge between learning and industry application, giving candidates a strong foundation for future roles in the data domain.

### ****Program Structure****

**Duration**: Typically 12 weeks

**Training Hours**: Over 30 hours of instruction

**Modules**: 25+ modules covering theoretical and practical aspects

**Projects**: 2 hands-on projects to apply learned skills

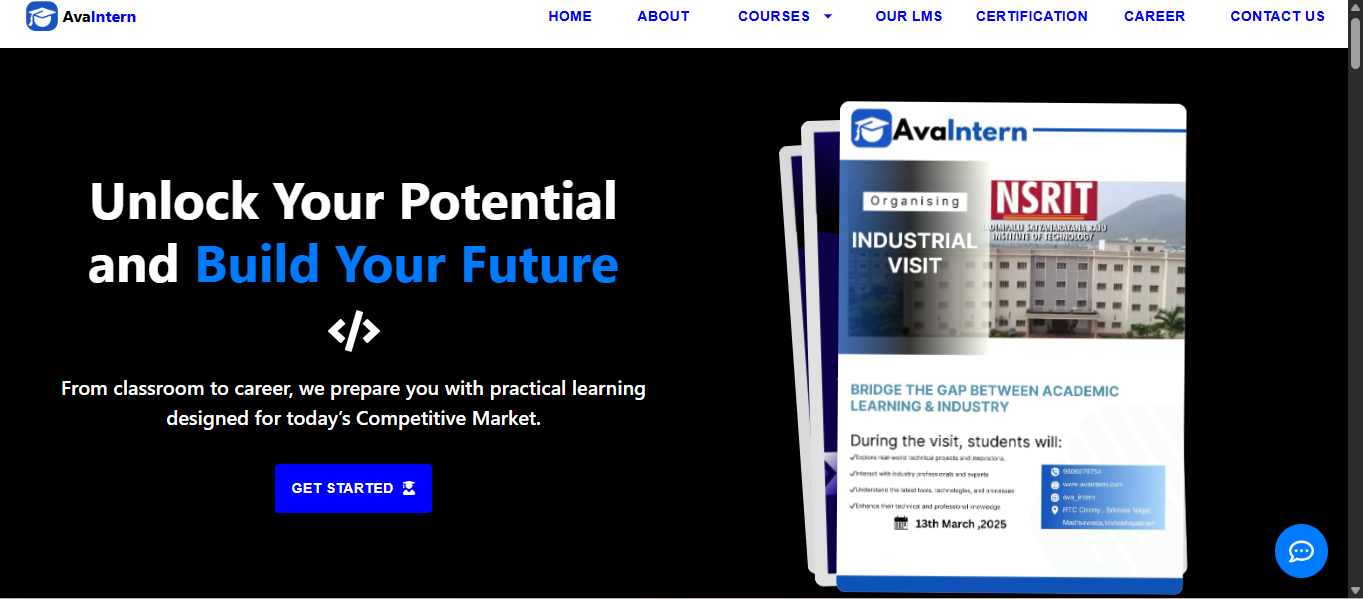
**Certification**: Industry-recognized certificate upon completion

**Mentorship**: Guidance from experienced professionals

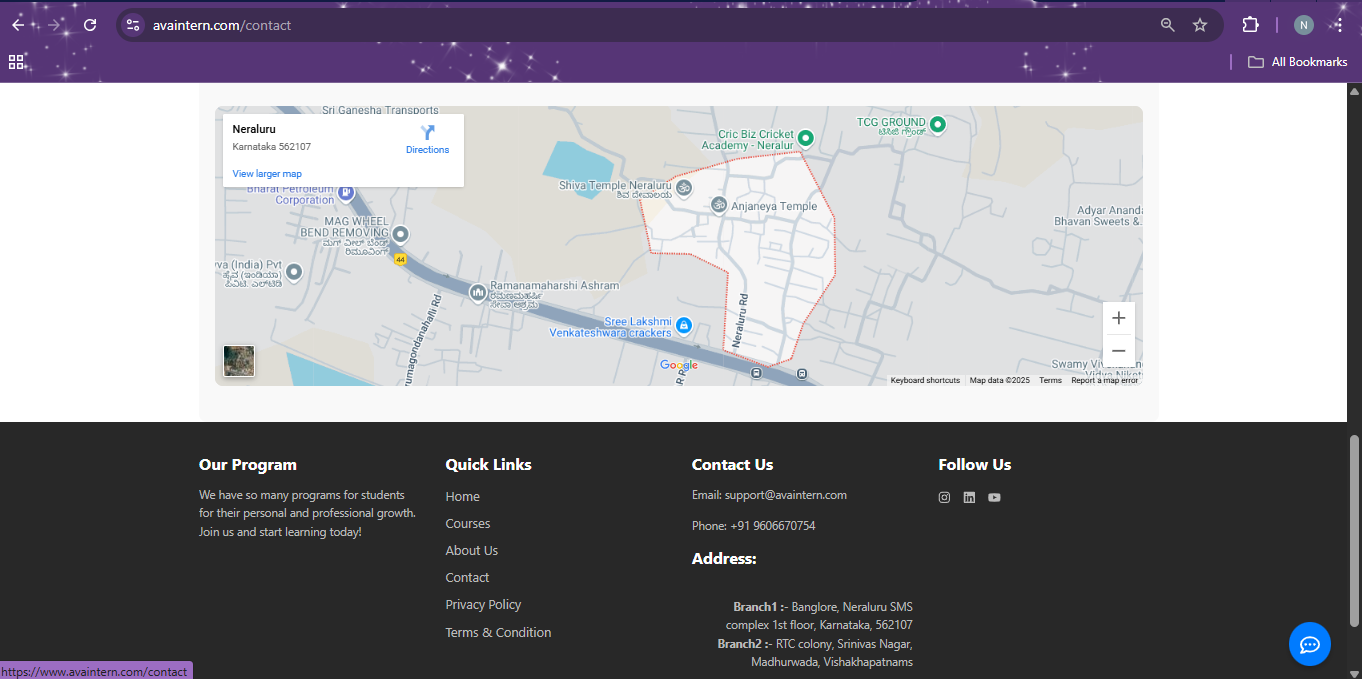
### ****Additional Benefits:****

* **Live Training Sessions**: Conducted via platforms like Google Meet or Zoom
* **Recorded Sessions**: Available through the Learning Management System (LMS) for review
* **Networking Opportunities**: Engage with recruiters and hiring managers through guided sessions
* **Placement Assistance**: Access to internship, freelance, and full-time job opportunities through the Ava Intern Placement Cell
* **Portfolio Development**: Inclusion of a minimum of two live projects in your portfolio.

**Ava Intern (snapshots):**

****

**Fig 1: Home Page of Ava Intern website**



**Fig 2: Contact Page of Ava Intern website**

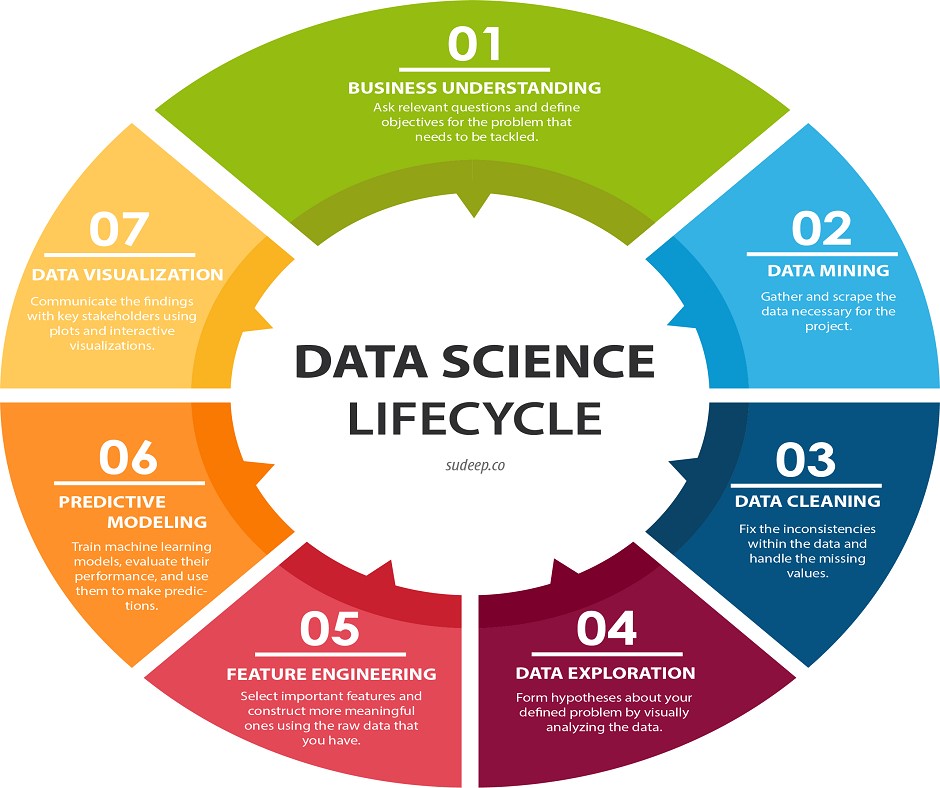
**2**: **WORK DONE DURING INTERNSHIP**

**2**: **WORK DONE DURING INTERNSHIP**

**2.1 About Domain:**

I have done the internship in Data Science using Python domain. Data Science is an interdisciplinary field that uses scientific methods, algorithms, processes, and systems to extract knowledge and insights from structured and unstructured data. It combines skills from computer science, mathematics, and statistics to understand and analyze real phenomena using data. Data science helps organizations make informed decisions by revealing trends, patterns, and predictions based on data analysis.

Python is the most popular and beginner-friendly programming language for data science due to its simplicity and powerful data-related libraries. Python allows you to perform tasks like data cleaning, data analysis, machine learning, and data visualization efficiently.



**Fig 3 : Data Science Life Cycle**

**Technology Related to Data Science using Python:**

### ****Data Collection****

Before performing any analysis, we need to collect data from different sources such as files, APIs, web scraping, or databases. Python provides several tools for data collection:

* **CSV / Excel / JSON / Text Files** – Common file formats that can be imported using libraries like pandas.
* **Requests** – A Python module used for accessing web data and APIs.
* **BeautifulSoup / Scrapy** – Libraries used for web scraping.
* **SQLAlchemy / PyMySQL / SQLite3** – Python connectors for fetching data from relational databases.

### ****Data Cleaning & Preparation****

Real-world data is often messy. Data cleaning is the process of removing errors, handling missing values, and transforming data into a usable format.

* **Pandas** – The most widely used library for data manipulation. It supports data filtering, transformation, and analysis.
* **NumPy** – Used for numerical operations and efficient array handling.
* **OpenPyXL / xlrd** – Libraries to read and write Excel files.

### ****Exploratory Data Analysis (EDA)****

EDA is the process of analyzing data sets to summarize their main characteristics, often using visual methods.

* **Matplotlib** – A basic plotting library for line, bar, pie, and scatter plots.
* **Seaborn** – Built on top of Matplotlib, it provides attractive statistical plots like heatmaps, histograms, and boxplots.
* **Plotly** – A library for creating interactive visualizations and dashboards.

### ****Statistical Analysis & Feature Engineering****

Statistical methods help understand the data and extract meaningful features:

* **SciPy / Statsmodels** – Libraries for performing statistical tests and regression analysis.
* **Feature Engineering** – Involves transforming raw data into features that better represent the underlying problem, such as scaling, encoding, or creating new variables.

### ****Machine Learning & Modeling****

Machine learning is at the core of predictive analytics. It involves training models on data to make predictions.

* **Scikit-learn** – A user-friendly machine learning library that supports classification, regression, clustering, and more.
* **XGBoost / LightGBM** – Advanced libraries for efficient and high-performing models, especially in competitions.
* **TensorFlow / PyTorch** – Libraries used for building deep learning models and neural networks.

### ****Big Data & Real-Time Processing****

For large datasets or real-time data, Python integrates with big data tools:

* **PySpark** – Interface for Apache Spark that allows processing large datasets in distributed systems.
* **Dask** – Parallel computing library for handling data that doesn’t fit into memory.

### ****Data Storage & Database Integration****

Working with databases is crucial in most data science workflows:

* **SQL (MySQL, PostgreSQL)** – Used for handling structured relational data.
* **MongoDB** – NoSQL database for storing unstructured data.
* **Firebase / Google Sheets API** – For storing or reading data from cloud services.

### ****Model Deployment & Reporting****

Once the model is built, it needs to be shared or deployed for others to use.

* **Flask / FastAPI** – Web frameworks used to deploy machine learning models as APIs.
* **Streamlit / Dash** – Tools for building interactive data science web apps with minimal code.
* **Jupyter Notebook / Google Colab** – Ideal for sharing research and documentation with code, charts, and text.

## ****Skills Needed for Data Science Using Python****

To become a successful data science professional using Python, one must develop a combination of technical programming skills, analytical thinking, and data storytelling capabilities. Below is a breakdown of the essential skills, tools, and knowledge areas required.

### ****1. Essential Programming & Tools****

#### **Python (Core Language):**

Python is the backbone of data science due to its simplicity and powerful libraries. It allows quick development of prototypes and has extensive support for data manipulation, visualization, and machine learning.

* Easy to learn syntax.
* Extensive community and library support.
* Ideal for scripting, automation, and algorithm implementation.

#### **Jupyter Notebook / Google Colab:**

These are browser-based environments designed for data science and machine learning tasks.

* **Jupyter Notebook**: Lets you combine live code, visualizations, and narrative text in one document.
* **Google Colab**: Offers similar features with free access to GPUs/TPUs, especially useful for deep learning tasks.

#### **Git and GitHub for Version Control:**

Version control is essential for collaboration and tracking project changes.

* **Git**: A tool to manage versions of code locally.
* **GitHub**: A cloud-based platform to host and share your code, collaborate with others, and maintain project history.

### ****2. Libraries & Frameworks****

#### **Pandas, NumPy (For Data Handling)**

* **Pandas**: Offers high-level data structures like Data Frames that make it easier to clean, filter, and analyze tabular data.
* **NumPy**: Provides support for numerical data and array operations, forming the foundation for many scientific computing tasks.

#### **Matplotlib, Seaborn, Plotly (For Data Visualization)**

* **Matplotlib**: A low-level plotting library used for basic charts such as line, bar, scatter, and pie charts.
* **Seaborn**: Built on top of Matplotlib and adds aesthetically pleasing charts, especially for statistical data.
* **Plotly**: An advanced and interactive visualization library useful for dashboards and web-based plots.

**Scikit-learn, XGBoost (For Machine Learning)**

* **Scikit-learn**: A robust library with simple and efficient tools for data mining and machine learning. It supports classification, regression, clustering, model evaluation, and more.
* **XGBoost**: A gradient boosting framework optimized for performance and accuracy, often used in data science competitions like Kaggle.

#### **TensorFlow, PyTorch (For Deep Learning)**

* **TensorFlow**: Developed by Google, it is widely used for neural network modeling, especially in production environments.
* **PyTorch**: Developed by Facebook, it is known for its simplicity and dynamic computation graph, making it suitable for research and development.

#### **Flask, Streamlit (For Deployment)**

* **Flask**: A lightweight web framework used to turn models into RESTful APIs, enabling integration into web applications.
* **Streamlit**: A fast and simple way to turn Python scripts into shareable data science web apps, great for showcasing your work to others.

### ****3. Other Important Skills****

#### **SQL for Querying Databases**

Structured Query Language (SQL) is used to extract and manage data stored in relational databases.

* Helps in retrieving, filtering, and joining data from large datasets.
* Often used to pre-process data before feeding it into a machine learning model.

#### **Statistics & Probability**

A solid foundation in statistics is necessary to understand data distributions, correlations, and hypotheses.

* Helps in interpreting the results of machine learning models.
* Useful in feature selection, performance evaluation, and data interpretation.

#### **Problem Solving and Critical Thinking**

Analyzing problems, identifying patterns, and thinking logically are essential in:

* Framing the right questions.
* Building efficient models.
* Optimizing results for accuracy and relevance.

#### **Communication and Storytelling with Data**

Being able to explain your findings to non-technical stakeholders is just as important as the technical analysis.

* Use of clear visualizations, dashboards, and presentations.
* Telling the “story” behind the data—why it matters and what actions should be taken.

## ****Types of Data Science Roles****

In the field of data science, there are multiple specialized roles that work together to handle, process, analyze, and extract insights from data. Each role focuses on a specific aspect of the data science lifecycle. Below are the most common roles in detail:

### ****1. Data Analyst****

A **Data Analyst** is primarily responsible for exploring and interpreting raw data to discover meaningful trends and insights that can support business decision-making. Their work often involves cleaning data, performing exploratory data analysis (EDA), and summarizing the results using statistical techniques and visualizations. **Tools used** are Microsoft Excel, SQL for querying databases, and Python for scripting and automation**. Main responsibilities** are cleaning and preparing data for analysis, Identifying trends, correlations, and patterns in datasets, Creating **dashboards** and **reports** using tools like Tableau, Power BI, or Matplotlib/Seaborn in Python, Supporting business teams with insights that aid in strategic decisions. The main **goal** is to enable companies to make data-driven decisions by presenting clear and actionable findings.

### ****2. Data Engineer****

A **Data Engineer** designs, builds, and maintains the data architecture (such as databases and large-scale processing systems) that enables the organization to store and access massive amounts of data efficiently. **Tools and Technologies** are SQL, Python, Apache Spark, Hadoop, Kafka, Airflow, and cloud platforms like AWS, Azure, or GCP**. Main responsibilities** are building **ETL (Extract, Transform, Load)** pipelines to move and transform data across systems, designing and managing **data warehouses** and **data lakes**, ensuring data quality, integrity, and availability for analysts and scientists, optimizing systems for **scalability** and **performance**. The main goal is to provide reliable infrastructure for storing and accessing data across various teams and applications.

### ****3. Data Scientist****

A **Data Scientist** blends statistical knowledge, programming expertise, and domain understanding to develop models that generate predictions or recommendations from data. **Tools used** are Python, R, Jupyter Notebooks, Scikit-learn, TensorFlow, SQL, and visualization libraries. **Main responsibilities are** understanding business problems and framing them as data science questions, performing **feature engineering** and **model training** using machine learning techniques, interpreting model outputs and validating performance using metrics like accuracy, precision, recall, and ROC-AUC, communicating findings through reports or interactive dashboards. **The main goal is t**o extract actionable insights and build predictive models that help businesses solve complex challenges.

### ****4. Machine Learning Engineer****

A **Machine Learning Engineer** focuses on developing and deploying machine learning models into production environments. While a data scientist may focus on building a prototype model, an ML engineer ensures that it runs efficiently in real-world applications. **Tools and Frameworks** TensorFlow, PyTorch, Scikit-learn, Docker, Kubernetes, MLflow, and REST APIs**. Main responsibilities** are writing **production-level code** to integrate models into applications, monitoring model performance and **retraining** when necessary, ensuring **scalability, latency,** and **reliability** of deployed models, collaborating with data scientists to transition prototypes to production-ready systems. **The main goal is to** operationalize machine learning by making sure models are robust, scalable, and maintainable over time.

**2.2 Technologies learnt during internship:**

I have completed a Data Science internship using Python, and it was an enriching experience that helped me gain hands-on exposure to a wide range of tools, libraries, and technologies used in the field of data science. The internship covered every critical stage of a data science workflow—from data collection and preprocessing to model building and result visualization. Here's a detailed breakdown of the key technologies and tools I learned:

* **Python**: Mastered core concepts, scripting, and automation using lists, dictionaries, and control structures.
* **Jupyter Notebook & Google Colab**: Used for interactive coding, documentation, and collaboration.
* **Git & GitHub**: Learned version control, collaboration, and project management using repositories and branches.
* **Pandas & NumPy**: Handled data preprocessing, analysis, and numerical computations efficiently.
* **Matplotlib, Seaborn & Plotly**: Created static and interactive visualizations for data exploration and presentations.
* **Scikit-learn & XGBoost**: Built and evaluated machine learning models; explored hyperparameter tuning techniques.
* **TensorFlow & PyTorch (Basics)**: Gained introductory exposure to deep learning workflows.
* **Flask & Streamlit**: Deployed models as APIs and interactive dashboards.
* **SQL**: Queried relational databases for data extraction and manipulation.
* **Statistics & Probability**: Applied statistical methods for data interpretation and hypothesis testing.
* **Analytical Thinking & Communication**: Strengthened problem-solving skills and data storytelling through reports and dashboards.
* Internship Projects Phase After the training phase, the internship projects began, providing an opportunity to apply the learned concepts in a practical setting and gain hands-on experience.

**2.3 Assessments/Tasks assigned details:**

As part of the structured learning process during the internship, a written assignment was conducted at the end of each month. These assignments were designed to assess the understanding of concepts taught throughout that month. Each assignment included theoretical questions based on Python programming, data analysis, machine learning, and deep learning topics. The questions tested both conceptual clarity and the ability to express ideas related to data science practices in a structured format.

### ****Assignment 1 – Month 1: Python Programming and Data Handling****

In the first month of the internship, we were introduced to the fundamentals of Python programming, including variables, data types, control structures, functions, and basic input/output operations. Alongside Python basics, we also learned how to work with essential data science libraries like NumPy and Pandas. At the end of the month, a written assignment was given to evaluate our understanding of these foundational concepts. The assignment included descriptive and code-based questions such as explaining the difference between lists and NumPy arrays, handling missing data using Pandas, writing basic scripts to read CSV files, and explaining how groupby() works. We were required to write the answers clearly and include syntax examples to show practical implementation. This assignment helped reinforce our grasp on core Python features and data manipulation techniques, setting the groundwork for more advanced topics in data science.

### ****Assignment 2 – Month 2: Data Visualization and Introduction to Machine Learning****

In the second month, the focus of the internship shifted toward data visualization and the basic principles of machine learning. We gained hands-on experience with libraries such as Matplotlib and Seaborn, learning to create a variety of visual plots to better understand data distributions, correlations, and trends. We also began exploring supervised learning, including linear and logistic regression models. A written assignment was given at the end of the month to assess our understanding of these new topics. The questions included the importance of data visualization in data science, differences between Matplotlib and Seaborn, the concept of supervised learning with examples, and the steps involved in building and evaluating a regression model. We also explained the significance of splitting datasets into training and testing sets.

In addition to theoretical learning, we also began focusing on project work during the second half of this month. We started working on both minor and major projects that applied the concepts we had learned so far, such as data analysis, visualization, and building basic machine learning models. This project development continued actively through the third month, allowing us to implement practical data science workflows and gain hands-on experience with real-world datasets.

**3. PROJECTS**

**3. PROJECTS**

**3.1 PROJECT 1: Gender and age detection:**

**1. Abstract:**

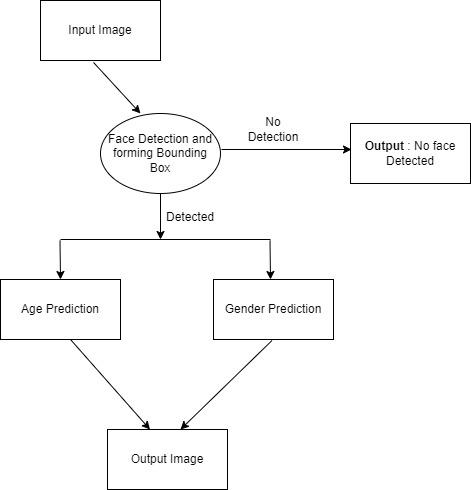
Gender and age detection is a crucial application of computer vision with uses in surveillance, marketing, and human-computer interaction. This project leverages deep learning models using OpenCV’s DNN module to accurately classify gender and estimate age from facial images. A pre-trained face detection model is employed to locate faces within an image or a webcam feed, followed by two specialized convolutional neural networks (CNNs) to determine gender (Male/Female) and categorize age into predefined groups. The system enhances detection accuracy by implementing padding and resizing techniques to improve face localization. The final results are displayed on the image with bold, clearly visible text. This approach provides a real-time, efficient, and scalable solution for demographic analysis, enabling its deployment in diverse real-world applications.

**2. Introduction:**

In recent years, computer vision and deep learning have revolutionized the field of human feature recognition. Gender and age detection is an essential application of these technologies, widely used in security systems, personalized marketing, social media analysis, and human-computer interaction. This project implements an automated gender and age detection system using OpenCV’s Deep Neural Network (DNN) module and pre-trained deep learning models.

The system works by first detecting faces in an image or live webcam feed using a pre-trained face detection model. Once a face is identified, two convolutional neural networks (CNNs) analyze the face to determine gender (Male/Female) and estimated age group. The results are then displayed directly on the image, providing a real-time, efficient, and scalable solution. The project also integrates image preprocessing techniques such as resizing and padding to improve detection accuracy. By leveraging pertained deep learning models, this project provides a robust, real-time, and user-friendly system for age and gender classification, making it suitable for a wide range of applications, from security enhancements to user experience personalization.

**Flowchart:**

****

**Fig 4: Flowchart for project (gender and age detection)**

**3. Research Methodology:**

**3.1 Basic Requirements:**

To run this project successfully, the following software and hardware requirements must be met:

* Operating System: Windows (Adequately equipped machine)
* Python Version: 2.7 - 3.6
* Libraries: OpenCV (cv2)
* Development Environment: PyCharm Community Edition (or any other Python IDE)
* Hardware: Webcam (at least 2.0 MP) for real-time detection

**3.2 Algorithm Workflow:**

The algorithm is structured into four main components:

* Input
* Face Detection
* Face Processing (Age & Gender Classification)
* Output

**3.2.1 Input:**

The system is designed to provide a fast and efficient method for obtaining input. Users can choose between two methods:

* Webcam Input: The system can directly capture live data from the webcam, allowing real-time age and gender detection.
* Image Input: The user can specify an image file (JPEG format) as input. This method allows batch processing of multiple images.

The input is processed to ensure proper face detection and classification. If an image is provided, it is read, resized, and analyzed. If no image is given, the system defaults to real-time detection via the webcam.

**3.2.2 Face Detection:**

* Uses Haar Classifier and a deep learning model to detect faces in images or video streams.
* Trained on a diverse dataset to handle variations in lighting, pose, and facial features.
* Bounding box adjustments improve accuracy.

**3.2.3 Face Processing (Age & Gender Classification):**

* Uses a CNN-based deep learning model with three convolutional layers.
* Gender Classification: Predicts Male or Female with a confidence score.
* Age Classification: Estimates age into predefined groups: (0-2), (4-6), (8-12), (15-20), (25-32), (38-43), (48-53), (60-100).

**3.2.4 Output:**

* Real-time Display: Overlays age and gender info on the webcam feed.
* Image Output: Saves the processed image with detection results in JPEG format.
* Displays "No Face Detected" if no face is found

**4. Steps for Practicing Gender & Age Detection in Python:**

1. Download and Extract Files

* The folder contains pre-trained models for face detection, age classification, and gender classification, along with sample images.
* Model files include:
* Face detection: opencv\_face\_detector.pbtxt,

opencv\_face\_detector\_uint8.pb

* Age classification: age\_deploy.prototxt,

age\_net.caffemodel

* Gender classification: gender\_deploy.prototxt,

gender\_net.caffemodel

2. Parse Image Argument

* Use the argparse library to get the image path from the command prompt.
* This allows the user to run the program with an image file instead of a live webcam feed.

3. Initialize Models

* Load pre-trained deep learning models for face, age, and gender detection.
* These models are based on CNN (Convolutional Neural Networks) and trained on large datasets for high accuracy.

4. Set Model Parameters

* Define mean values for normalizing images.
* Specify age groups: (0-2), (4-6), (8-12), (15-20), (25-32), (38-43), (48-53), (60-100).
* Define gender labels: Male, Female.

5. Load the Deep Learning Models

* Use OpenCV’s cv2.dnn.readNet() function to load:
* Face detection model (faceNet)
* Age classification model (ageNet)
* Gender classification model (genderNet)

6. Capture Video Stream

* If no image is provided, the program captures real-time video from a webcam.
* Video processing ensures live age and gender detection.
* Set padding of 20 pixels for better face detection.

7. Process Each Frame

* Read frames from the webcam or load the input image.
* If an image is used, process it immediately.
* If using a webcam, continuously read and process frames until a key is pressed.

8. Detect Faces (highlightFace())

* Create a blob from the image and pass it through the face detection model.
* The model detects faces and returns bounding box coordinates.
* Confidence Threshold: If confidence is above 70%, the face is considered detected.
* A green rectangle is drawn around each detected face.

9. Extract Face & Process Data

* Extract the detected face region from the image.
* Convert the face into a 4D blob, scale it, resize it, and normalize it.

10. Predict Gender

* Pass the face through the gender classification model.
* The model outputs probabilities for Male and Female categories.
* The label with the highest confidence is assigned.

11. Predict Age

* Pass the face through the age classification model.
* The model classifies the face into one of the 8 predefined age groups.
* The most probable age range is assigned to the face.

12. Display Output

* Overlay gender and age predictions on the image.
* If using an image file, the output is displayed in an OpenCV window.
* If using a webcam, the output is shown in real-time with predictions updated on each frame.
* If no face is detected, display: "No Face Detected".

**5. Code Implementation:**

import cv2

import os

# Provide an image path OR set it to None for webcam mode

image\_path = r"C:\\picture\\ff.jpg" # Change this to None for webcam

#r"C:\\picture\\girl1.jpg" to detect for an image

# Model files

faceProto=r"C:\\Users\DELL\\OneDrive\Desktop\\example\\models\\opencv\_face\_detector.pbtxt"

faceModel=r"C:\\Users\DELL\\OneDrive\Desktop\\example\\models\\opencv\_face\_detector\_uint8.pb"

ageProto=r"C:\\Users\DELL\\OneDrive\Desktop\\example\\models\\age\_deploy.prototxt"

ageModel=r"C:\\Users\DELL\\OneDrive\Desktop\\example\\models\\age\_net.caffemodel"

genderProto=r"C:\\Users\DELL\\OneDrive\Desktop\\example\\models\\gender\_deploy.prototxt"

genderModel=r"C:\\Users\DELL\\OneDrive\Desktop\\example\\models\\gender\_net.caffemodel"

# Check if model files exist

for model\_file in [faceProto, faceModel, ageProto, ageModel, genderProto, genderModel]:

    if not os.path.exists(model\_file):

        raise FileNotFoundError(f"Error: Model file '{model\_file}' not found. Please check the file path.")

# Load models

faceNet = cv2.dnn.readNet(faceModel, faceProto)

ageNet = cv2.dnn.readNet(ageModel, ageProto)

genderNet = cv2.dnn.readNet(genderModel, genderProto)

MODEL\_MEAN\_VALUES = (78.4263377603, 87.7689143744, 114.895847746)

ageList = ['(0-2)', '(4-6)', '(8-12)', '(15-20)', '(25-32)', '(38-43)', '(48-53)', '(60-100)']

genderList = ['Male', 'Female']

def highlightFace(net, frame, conf\_threshold=0.7):

    frameOpencvDnn = frame.copy()

    frameHeight, frameWidth = frame.shape[:2]

    blob = cv2.dnn.blobFromImage(frameOpencvDnn, 1.0, (300, 300), [104, 117, 123], True, False)

    net.setInput(blob)

    detections = net.forward()

    faceBoxes = []

    for i in range(detections.shape[2]):

        confidence = detections[0, 0, i, 2]

        if confidence > conf\_threshold:

            x1 = int(detections[0, 0, i, 3] \* frameWidth)

            y1 = int(detections[0, 0, i, 4] \* frameHeight)

            x2 = int(detections[0, 0, i, 5] \* frameWidth)

            y2 = int(detections[0, 0, i, 6] \* frameHeight)

            padding\_x = int((x2 - x1) \* 0.15)  # 15% of face width

            padding\_y = int((y2 - y1) \* 0.15)  # 15% of face height

            x1, y1 = max(0, x1 - padding\_x), max(0, y1 - padding\_y)

            x2, y2 = min(frameWidth - 1, x2 + padding\_x), min(frameHeight - 1, y2 + padding\_y)

            faceBoxes.append([x1, y1, x2, y2])

            cv2.rectangle(frameOpencvDnn, (x1, y1), (x2, y2), (0, 255, 0), 2)

    return frameOpencvDnn, faceBoxes

def detect\_age\_gender(frame):

    resultImg, faceBoxes = highlightFace(faceNet, frame)

    if not faceBoxes:

        print("No face detected")

        return frame  # Return original frame if no face detected

    for faceBox in faceBoxes:

        face = frame[max(0, faceBox[1] - 20): min(faceBox[3] + 20, frame.shape[0] - 1),

                     max(0, faceBox[0] - 20): min(faceBox[2] + 20, frame.shape[1] - 1)]

        blob = cv2.dnn.blobFromImage(face, 1.0, (227, 227), MODEL\_MEAN\_VALUES, swapRB=False)

        genderNet.setInput(blob)

        genderPreds = genderNet.forward()

        gender = genderList[genderPreds[0].argmax()]

        ageNet.setInput(blob)

        agePreds = ageNet.forward()

        age = ageList[agePreds[0].argmax()]

        print(f'Gender: {gender}, Age: {age[1:-1]} years')

        cv2.putText(resultImg, f'{gender}, {age}', (faceBox[0], faceBox[1] - 10),

                    cv2.FONT\_HERSHEY\_SIMPLEX, 0.8, (0, 255, 255), 2, cv2.LINE\_AA)

    return resultImg

if image\_path:

    # Detect from an image

    if not os.path.exists(image\_path):

        raise FileNotFoundError(f"Error: Image file '{image\_path}' not found. Please check the file path.")

    img = cv2.imread(image\_path)

    processed\_img = detect\_age\_gender(img)

    cv2.namedWindow("Age and Gender Detection", cv2.WINDOW\_NORMAL)

    cv2.resizeWindow("Age and Gender Detection", 800, 600)

    cv2.imshow("Age and Gender Detection", processed\_img)

    cv2.waitKey(0)

    cv2.destroyAllWindows()

else:

    # Detect from webcam (only one frame)

    video = cv2.VideoCapture(0)

    video.set(cv2.CAP\_PROP\_FRAME\_WIDTH, 1280)

    video.set(cv2.CAP\_PROP\_FRAME\_HEIGHT, 720)

    hasFrame, frame = video.read()

    video.release()  # Release immediately after capturing a frame

    if hasFrame:

        processed\_img = detect\_age\_gender(frame)

        cv2.namedWindow("Age and Gender Detection", cv2.WINDOW\_NORMAL)

        cv2.resizeWindow("Age and Gender Detection", 800, 600)

        cv2.imshow("Age and Gender Detection", processed\_img)

        cv2.waitKey(0)

        cv2.destroyAllWindows()

Here we have given an image path in the code, by taking that picture it will detect the gender of the person present in the image and the age of the person in the given age groups (0-2, 4-6). If we provide None value in the image path it will detect through webcam and provide the result of the person present infront of the webcam. The output of the image gender and age detection is given below:

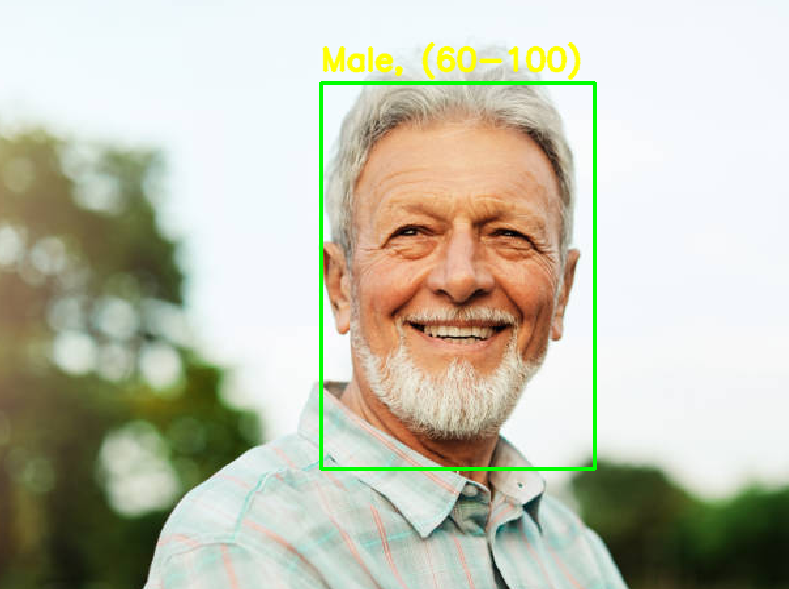
**Output:**

1. Gender: Female, Age: 25-32 years



**Fig 4: Output of project 1(Female gender detection)**

2. Gender: Male, Age: 60-100 years



**Fig 5: Output of project 1 (Male gender detection)**

**6. Use Cases:**

Several uses cases for this project includes the following:

* + Identification of the target audience in marketing organisation.
  + In Recruitment procedure, to verify legitimacy of the applicants.
  + Verification of authentic person applying for government IDs.
  + Classification of human resources in bulk.

**7. Conclusion:**

This project provides a real-time and automated gender and age detection system using deep learning. By leveraging CNN models, the system ensures accurate classification across diverse face variations. The modular design allows easy integration into applications requiring facial analysis, such as biometric authentication, audience analytics, and interactive user experiences. This implementation can be extended to include additional features like emotion detection, ethnicity classification, and facial expression analysis, making it a powerful tool in the field of computer vision and AI-based analytics.

**3.2 PROJECT 2: BASIC PORTFOLIO WEBSITE**

A movie recommendation system suggests movies to users based on their interests. You’ve seen it on Netflix, Amazon Prime, etc. — like when Netflix says "Because you watched Inception, you might like Interstellar."

**Abstract:**

In today's digital era, users are overwhelmed by the sheer volume of movies available across various platforms. A Movie Recommendation System aims to address this challenge by suggesting movies that align with a user's preferences, thus enhancing their viewing experience. This system leverages data-driven approaches such as collaborative filtering, content-based filtering, and hybrid methods to analyze user behavior, movie attributes, and ratings. By processing large datasets, the system identifies patterns and predicts movies that users are most likely to enjoy. Advanced techniques, including machine learning algorithms and natural language processing, are employed to further refine recommendations. The project also integrates APIs like TMDb (The Movie Database) to fetch real-time information and visually appealing posters, ensuring an engaging and dynamic user interface.

### 1. Introduction

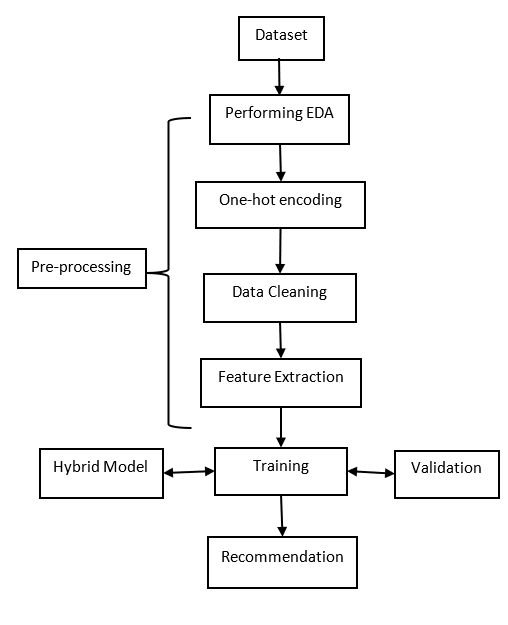
The Movie Recommendation System was developed to suggest personalized movie choices to users based on their interests and past behavior. With the explosion of online streaming platforms, users often find it difficult to pick the next movie to watch. This project aims to solve this problem by building an intelligent recommendation engine that offers tailored suggestions.

### 2. Objective

The primary objectives of the Movie Recommendation System were:

* To help users find movies similar to their interests quickly and accurately.
* To implement a scalable, efficient, and easy-to-use recommendation engine.
* To improve user interaction by integrating movie posters, metadata, and an appealing user interface.
* To gain hands-on experience with machine learning models, similarity algorithms, and API integration.

**Flowchart:**

****

**Fig 7: Flowchart of project (Movie recommendation system)**

### 3. Technologies Used

* **Programming Language**: Python
* **Libraries**: Pandas, NumPy, Scikit-learn, Streamlit (for frontend), Requests (for API integration)
* **API**: TMDb (The Movie Database) API for fetching movie posters and metadata
* **Machine Learning Techniques**:
* Content-Based Filtering
* Collaborative Filtering (conceptual understanding)
* Cosine Similarity
* Nearest Neighbors algorithms

### 4. Methodology

### 4.1 Dataset Preparation

* **Data Source**: Used a Kaggle dataset containing thousands of movie records with features like title, genre, cast, crew, and keywords.
* **Data Cleaning**:
* Handled missing values.
* Standardized and merged important features into a unified textual format.

### 4.2 Feature Engineering

* Created a **'bag of words'** by combining relevant columns such as:
  + Genre
  + Director
  + Cast
  + Movie Keywords
* This enriched feature set allowed better content-based matching.

### 4.3 Similarity Measurement

* Calculated **cosine similarity** between movie vectors.
* Stored similarity scores to instantly fetch the most related movies for any selected title.

### 4.4 API Integration

* Connected to **TMDb API** to retrieve dynamic movie posters and additional metadata like release date, ratings, and descriptions.
* Managed API keys securely and optimized calls to stay within rate limits.

### 4.5 Front-End Development

* Built a **Streamlit web application**:
  + User-friendly dropdown for movie selection.
  + Instantly displayed recommended movie titles alongside corresponding posters.
  + Added titles, descriptions, and ratings to make recommendations informative.

## 5. System Architecture

* The user selects a movie through the Streamlit web interface.
* The recommendation model processes the selection and finds similar movies.
* Poster images and movie metadata are fetched using the TMDb API.
* Results are displayed dynamically to the user with movie titles and posters.

### 6. Outcome

* Created a fully functional and visually appealing Movie Recommendation System.
* Successfully deployed a web-based tool that can recommend 5–10 movies within seconds based on a user's choice.
* Improved knowledge of machine learning algorithms and how they can be applied in real-world recommendation scenarios.
* Gained practical experience in integrating APIs into applications to enrich user experience.

### 7. Challenges Faced

* **Data Cleaning**: Handling missing or inconsistent movie data required careful preprocessing.
* **Poster Fetching Errors**: Managing errors when movies were not found in TMDb and providing fallback options.
* **Similarity Tuning**: Fine-tuning the model to avoid recommending irrelevant movies with low similarity scores.
* **API Rate Limits**: Dealing with TMDb’s API call limits and optimizing requests to stay within usage limits.

### 8. Learning Outcomes

* Deepened understanding of content-based filtering techniques.
* Improved skills in Python libraries related to data processing and machine learning.
* Gained hands-on experience in API integration and web app development using Streamlit.
* Learned how to build a complete end-to-end project, from data preprocessing to deployment.

**App.py:**

import streamlit as st

import pickle

import requests

movies = pickle.load(open("movies\_list.pkl",'rb'))

similarity = pickle.load(open("similarity.pkl",'rb'))

movies\_list=movies['title'].values

st.header("Movie recommendation System")

import streamlit as st

selectvalue=st.selectbox("Select movies from dropdown",movies\_list)

def fetch\_poster(movie\_id):

     url="https://api.themoviedb.org/3/movie/3?api\_key=c7ec19ffdd3279641fb606d19ceb9bb1&language=en-US"

     requests.get(url)

     data=data.json()

     poster\_path=data['poster\_path']

    full\_path = "https://www.wallpaperflare.com/static/925/818/454/the-godfather-movies-vito-corleone-godfather-wallpaper.jpg"+poster\_path

     return full\_path

def recommend(movie):

    index=movies[movies['title']==movie].index[0]

    distance=sorted(list(enumerate(similarity[2])), reverse=True,key=lambda vector:vector[1])

    recommend\_movie=[]

    recommend\_poster=[]

    for i in distance[1:6]:

          recommend\_movie.append(movies.iloc[i[0]].title)

          recommend\_poster.append(fetch\_poster(movies\_id))

    return recommend\_movie,recommend\_poster

if st.button("Show recommend"):

     movie\_name, movies\_poster  =recommend(selectvalue)

     col1,col2,col3,col4,col5=st.columns(5)

     with col1:

          st.text(movie\_name[0])

          st.image(movie\_poster[0])

     with col2:

          st.text(movie\_name[1])

          st.image(movie\_poster[1])

     with col3:

          st.text(movie\_name[2])

          st.image(movie\_poster[2])

     with col4:

          st.text(movie\_name[3])

          st.image(movie\_poster[3])

     with col5:

          st.text(movie\_name[4])

          st.image(movie\_poster[4])

**Main.ipynb**

* import pandas as pd
* movies = pd.read\_csv('datasetm.csv', encoding='latin1')
* movies.head()

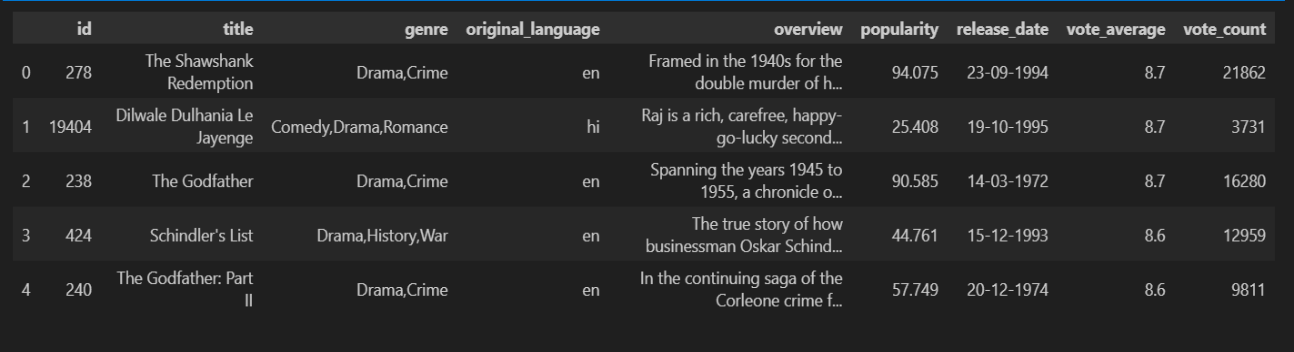


Fig 8.1: Result of command 1

* movies.describe()

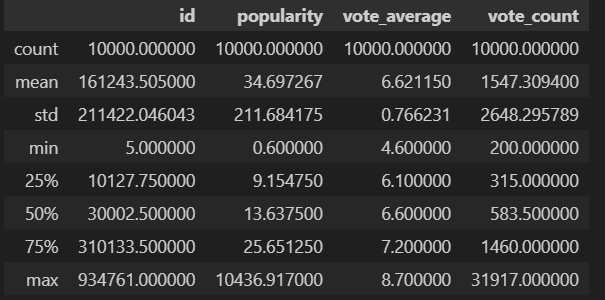


Fig 8.2: Result of command 2

* movies.info()

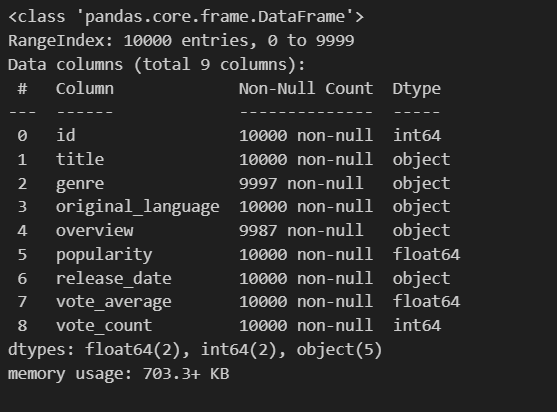


Fig 8.3: Result of command 3

* movies.column()

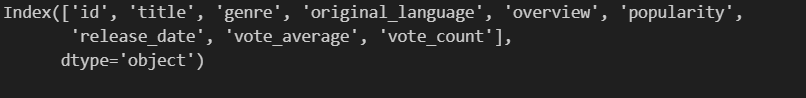


Fig 8.4: Result of command 4

* movies=movies[['id','title','overview','genre']]
* movies

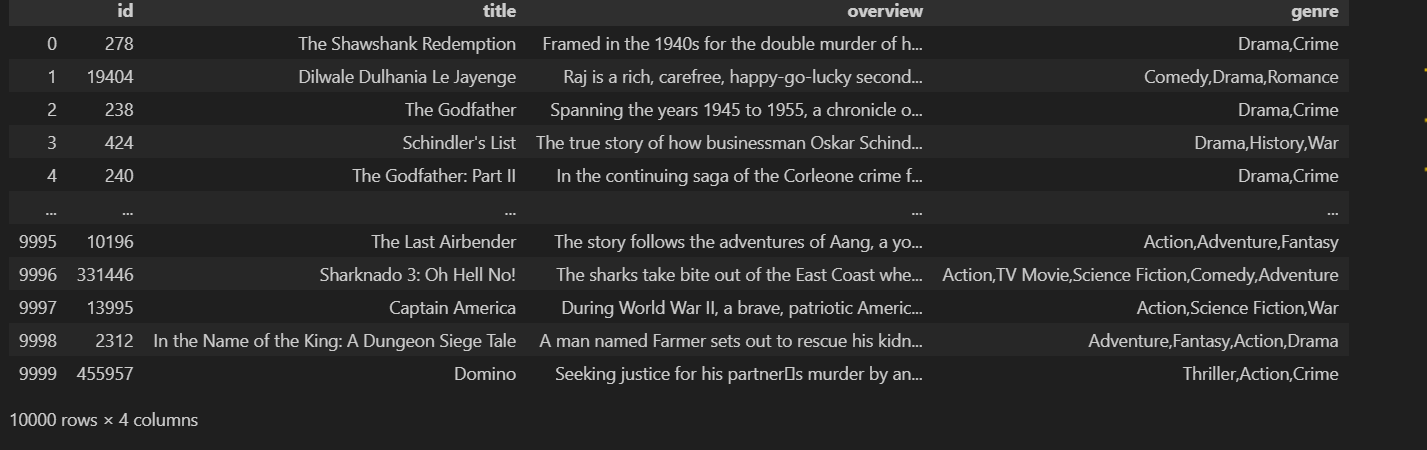


Fig 8.5: Result of command 5

* new\_data=movies.drop(columns=['overview','genre'])
* new\_data



Fig 8.6: Result of command 6

* from sklearn.feature\_extraction.text import CountVectorizer
* cv=CountVectorizer(max\_features=10000, stop\_words='english')
* cv

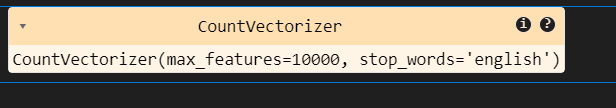


Fig 8.7: Result of command 7

* vector=cv.fit\_transform(new\_data['tags'].values.astype('U')).toarray()
* vector.shape
* Output: (10000, 10000)
* from sklearn.metrics.pairwise import cosine\_similarity
* similarity=cosine\_similarity(vector)
* similarity

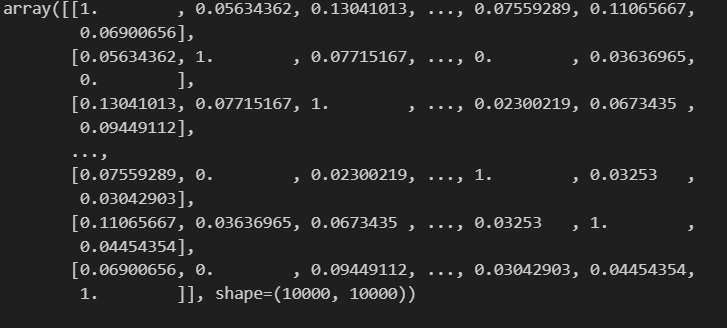


Fig 8.8: Result of command 8

* new\_data[new\_data['title']=="The Godfather"].index[0]
* Output: np.int64(2)
* new\_data.info()

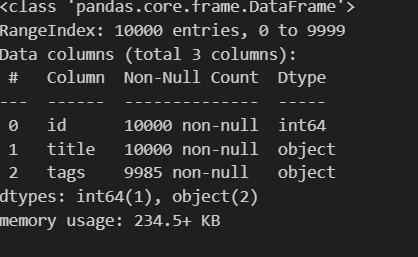
****

Fig 8.9: Result of command 9

* list(enumerate(similarity[2]))

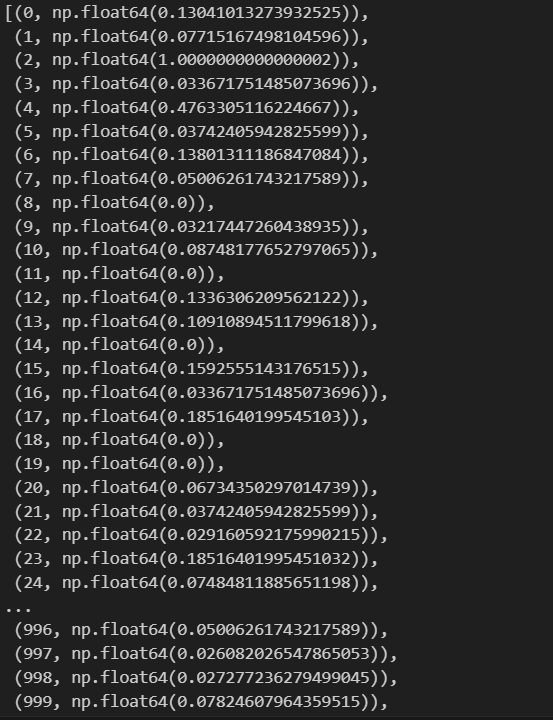


Fig 8.10: Result of command 10

* distance=sorted(list(enumerate(similarity[2])), reverse=True,key=lambda
* vector:vector[1])
* for i in distance[0:5]:
* print(new\_data.iloc[i[0]].title)
* def recommand(movies):

index=new\_data[new\_data['title']==movies].index[0]

distance=sorted(list(enumerate(similarity[2])), reverse=True,key=lambda vector:vector[1])

for i in distance[0:5]:

print(new\_data.iloc[i[0]].title)

* recommand("Iron Man")

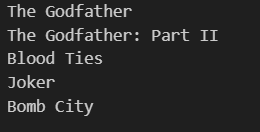


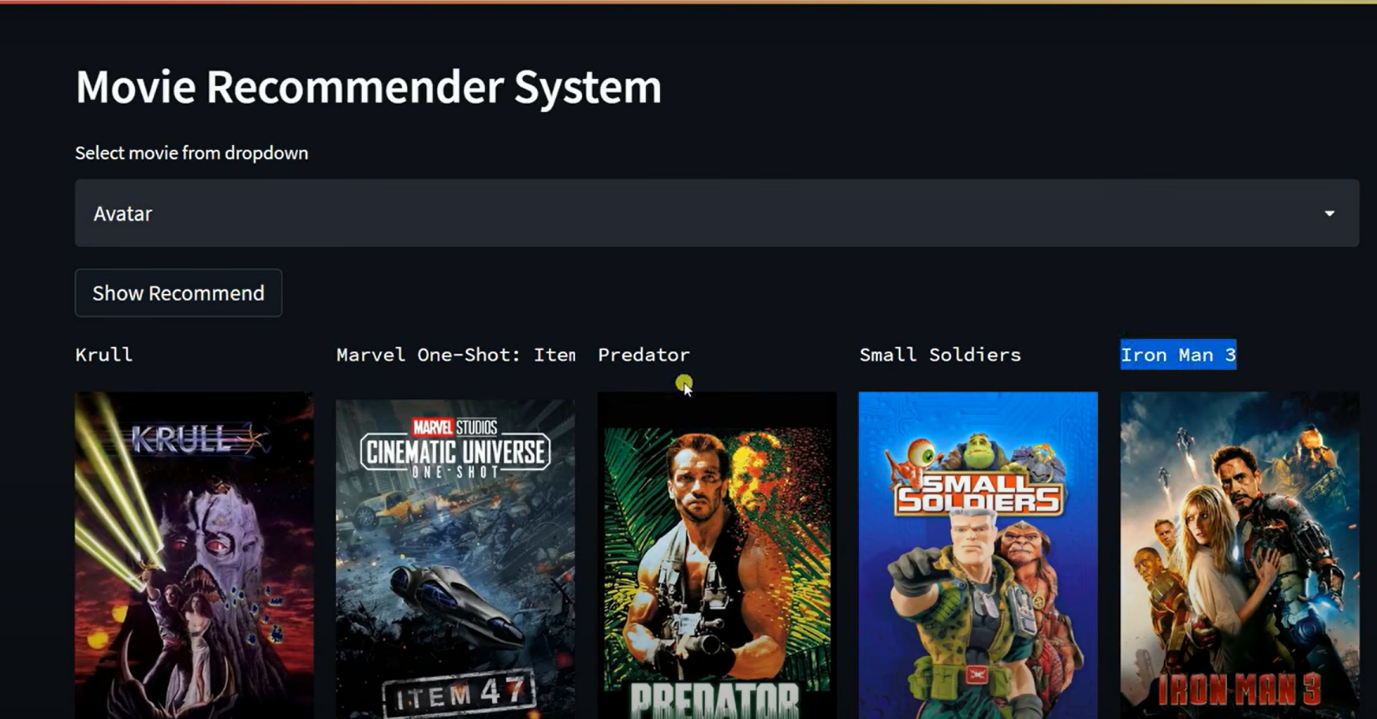
Fig 8.11: Result of command 11

* import pickle
* pickle.dump(new\_data, open('movies\_list.pkl','wb'))
* pickle.dump(similarity, open('similarity.pkl','wb'))
* pickle.load( open('movies\_list.pkl','rb'))



Fig 8.12: Result of command 12

**OUTPUT/RESULT:**



**Fig 9: Output of project 2**

**4. ACTIVITY LOG**

**Template for Chapter 4**

**ACTIVITYLOGFORTHEWEEK**

|  |  |  |  |
| --- | --- | --- | --- |
| **weeks** | **Description of the weekly activity** | **Learning Outcome** | **Person In- Charge Signature** |
| Week–1 | The internship started with an introduction to Python programming, laying the groundwork for all future learning. The sessions covered fundamental topics such as Python syntax, data types, variables, conditional statements, loops, and functions. We also discussed the importance of Python’s simplicity and versatility across different domains like web development, data science, and automation. This foundational week was critical for those unfamiliar with programming or transitioning from another language. Along with theoretical sessions, we practiced writing small scripts and solving basic problems, which helped in strengthening our understanding and improving our logical thinking skills. | We gained a solid understanding of Python basics, including how to write clean, efficient, and structured code. This week laid the groundwork for all future learning in the internship. |  |
| Week-2 | In the second week, we explored Python's built-in data structures such as **lists, tuples, dictionaries**, and **sets**, and learned about their characteristics and operations. We also practiced **file input/output operations**, including reading from and writing to different file formats like **.txt**, **.csv**, and **.json**. In addition, we were introduced to **exception handling** and learned to manage errors effectively using try-except blocks to build more reliable programs. | We became proficient in manipulating data using Python data structures, understood how to handle files programmatically, which is essential in data preprocessing tasks. |  |
| Week-3 | We were introduced to NumPy and Pandas, two of the most essential libraries for data analysis in Python. We worked with arrays, performed mathematical operations, and used Pandas for loading, cleaning, and transforming datasets. This week involved hands-on work with real datasets like sales records and customer data. | We learned how to handle large datasets using NumPy arrays and Pandas Data Frames. We also practiced data wrangling techniques including handling missing data, filtering, merging, and reshaping data. |  |
| Week-4 | This week focused on understanding data through visualizations. We used **Matplotlib** and **Seaborn** to plot bar charts, histograms, scatter plots, box plots, and heatmaps. We also explored various customization techniques such as changing colors, adding titles, labels, and legends to make the graphs presentation-ready. Special emphasis was placed on choosing the right type of plot for different kinds of data. Additionally, we learned how effective visualizations can reveal hidden patterns, trends, and outliers that are not easily noticeable in raw datasets. | We developed the ability to visually analyze data, identify trends, and communicate insights effectively through informative graphs and plots. This skill is crucial in data science, as clear visualizations often help in better decision-making and storytelling with data. |  |
| Week-5 | In the fifth week, we were introduced to the fundamental concepts of Machine Learning. We studied supervised and unsupervised learning, types of ML algorithms, model selection, and evaluation metrics like accuracy, precision, recall, and F1-score. We also discussed real-world applications of machine learning and how different algorithms are suited for different types of problems. | We built our first ML models using Scikit-learn and gained a conceptual understanding of how machines learn from data. |  |
| Week-6 | We deep-dived into **supervised algorithms** such as **Linear Regression, Logistic Regression, Decision Trees,** and **Random Forest**. We trained models on datasets like housing prices and classification tasks using labeled data. Through this, we gained hands-on experience in evaluating model performance using metrics like **accuracy, RMSE,** and **confusion matrices.** We also learned how to fine-tune model parameters to improve predictions and reduce overfitting. | We learned how to train and evaluate regression and classification models, perform feature engineering, and understand overfitting and underfitting. |  |
| Week-7 | This week, we explored unsupervised learning techniques such as K-Means Clustering, Hierarchical Clustering, and PCA (Principal Component Analysis). We applied these techniques to group unlabeled data and reduce dimensionality. Visualizing clusters helped us interpret patterns in the data more effectively. These methods also provided insights into how data can be organized without predefined labels. | We understood how to find hidden patterns in data without labels and how to preprocess and scale features appropriately for clustering tasks. |  |
| Week-8 | We learned about various model evaluation techniques such as cross-validation, confusion matrix, ROC-AUC, and hyperparameter tuning using GridSearchCV. The emphasis was on improving model performance. We also understood the importance of selecting the right evaluation method based on the problem type and dataset characteristics. | We became proficient in evaluating and fine-tuning machine learning models to improve their generalization capabilities and performance on unseen data. |  |
| Week-9 | We stepped into the field of **Deep Learning** by first learning about neural networks, including their structure and how they function. Key concepts such as **activation functions optimizers** and **loss functions** were introduced to help us understand how models learn and improve over time. We also studied **backpropagation**, the core algorithm that updates network weights based on error gradients. Using **TensorFlow** and **Keras**, we implemented basic neural networks, which gave us practical experience in building, training, and evaluating models. This foundation prepared us for exploring more advanced deep learning architectures in later stages of the internship. | We gained a foundational understanding of deep neural networks and successfully built and trained simple neural networks on image and numeric datasets. |  |
| Week-10 | We focused on Convolutional Neural Networks (CNNs) for image processing tasks. We covered convolution, pooling, and basic architecture design to understand how CNNs extract features from images. Using datasets like MNIST and CIFAR-10, we trained models for tasks like digit and object recognition. We also applied simple data augmentation techniques to improve model performance. | We learned how to create and train convolutional networks to perform image classification tasks with high accuracy using TensorFlow/Keras. |  |
| Week-11 | In this week, we explored basic NLP techniques including tokenization, stopword removal, and text vectorization using TF-IDF. We also studied Recurrent Neural Networks (RNNs) for sequence modeling and text generation. | We became familiar with preprocessing text data and developing models for tasks like sentiment analysis and sequence prediction. |  |
| Week-12 | In the final week, we consolidated our learning through two projects. A **Minor Project** was completed at the end of the Machine Learning module, involving prediction and classification on real-world tabular datasets. For the **Major Project**, we developed a complete end-to-end solution using Deep Learning — involving either a computer vision task or NLP-based application, such as image recognition or spam detection. | By completing the projects, we demonstrated our ability to independently design, develop, train, and evaluate ML and DL models, integrating all the skills acquired throughout the internship. |  |

**5. CONCLUSION**

**5. CONCLUSION**

The Data Science using Python internship has been a highly enriching and transformative journey, equipping me with the theoretical foundations, practical skills, and hands-on experience necessary to thrive in the data-driven industry. Over the course of twelve weeks, I not only gained a strong command over the Python programming language but also explored the complete lifecycle of data science projects—from data collection and cleaning to visualization, machine learning, and deep learning model deployment.

The internship began with an in-depth introduction to Python, which served as the cornerstone for all upcoming tasks. Gradually, we transitioned into specialized libraries like NumPy and Pandas for data manipulation, followed by Matplotlib and Seaborn for effective data visualization. This initial phase helped me develop a critical understanding of how to handle and interpret large datasets.

As we progressed, the focus shifted towards core concepts in Machine Learning. Here, I learned to build predictive models, evaluate them using metrics like accuracy and F1-score, and optimize their performance through techniques like cross-validation and hyperparameter tuning. The supervised and unsupervised learning modules helped me understand real-world applications such as regression analysis, classification, and clustering in various domains.

In the final phase, the internship delved into Deep Learning, where I worked with neural networks, CNNs, and RNNs using TensorFlow and Keras. This part was especially engaging as I could see how deep learning models are applied to complex problems like image recognition and natural language processing.

Working on a major project towards the end allowed me to apply all the concepts learned throughout the internship in a cohesive and meaningful way. The hands-on projects—both minor and major—were instrumental in solidifying my understanding and gave me a glimpse into the real-world challenges faced in the data science field. These projects not only honed my technical abilities but also improved my problem-solving, critical thinking, and collaborative skills.

Overall, this internship has significantly boosted my confidence and interest in the field of Data Science. It has laid a strong foundation for my future endeavors and has inspired me to continue learning and exploring emerging technologies. I now feel more prepared to take on data-centric roles in both academic and industrial settings. The journey from understanding basic Python scripts to implementing complex neural networks has been incredibly rewarding, and I am grateful for the structured guidance and exposure this program provided.

Beyond the technical skills, the internship also emphasized the importance of soft skills such as effective communication, teamwork, and time management. Regular presentations, peer reviews, and collaborative assignments simulated a real workplace environment, helping me articulate my ideas clearly and work efficiently within a team. The mentorship and feedback received throughout the program were invaluable in refining both my technical output and professional demeanor. This holistic learning approach has not only made me a better data scientist but also a more adaptable and well-rounded individual, ready to contribute meaningfully to any data-driven initiative.

**6. REFERENCES**

## ****6. References****

* <https://www.mdpi.com/2079-9292/10/20/2470>
* <https://iopscience.iop.org/article/10.1088/1742-6596/2649/1/012032/pdf>
* <https://openaccess.thecvf.com/content_cvpr_2013/papers/Pepikj_Occlusion>
* [Patterns\_for\_2013\_CVPR\_paper.pdf](https://openaccess.thecvf.com/content_cvpr_2013/papers/Pepikj_Occlusion_Patterns_for_2013_CVPR_paper.pdf)
* <https://www.sciencedirect.com/science/article/abs/pii/S0304389408015938>
* [Explosion pressures of hydrocarbon–air mixtures in closed vessels -](https://www.sciencedirect.com/science/article/abs/pii/S0304389405007545) [ScienceDirect](https://www.sciencedirect.com/science/article/abs/pii/S0304389405007545)
* <http://eemj.icpm.tuiasi.ro/pdfs/vol18/full/no4/14_293_Nan_18.pdf>
* <https://iopscience.iop.org/article/10.1088/1742-6596/1004/1/012029/pdf>
* <https://www.mdpi.com/2079-9292/12/10/2323>
* <https://link.springer.com/article/10.1007/s11042-022-13153-y>
* <https://ieeexplore.ieee.org/abstract/document/10473783>
* <https://app.roboflow.com/scrap-yard-project-zp6ho>