Artificial Intelligence and Machine Learning Internship

A Project Report submitted to the

GLOBAL NEXT CONSULTING INDIA PVT LTD

(Six – Week Internship Program)

By

PRIYANSHI BHANDARI

Under the Supervision of

Dr. ANURADHA GUPTA
(Project Director)

Duration of Internship :

23-March-2026 to 7-May-2026

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this report, **"AIML Internship (GNCIPL)"**, submitted as per the requirements for the ML engineer/ AI engineer/ Data engineer/ Data Analyst/ Data Science role.This is the result of original work carried out by me under the guidance of **Ms. Anuradha Gupta** during the time period from March 2026 to May 2026.

I further declare that this report represents authentic record of my own work and does not contain any falsely fabricated ideas, data, facts or sources. I also declare that I have adhered to all principles of academic honesty and integrity and that this report has not been submitted, either in part or in full, to any other institute, university, or organization for the award of any degree, diploma, or certification.

Priyanshi Bhandari

CERTIFICATE

This is to certify that the project report entitled **"AIML Internship Report"** has been carried out by **, Priyanshi Bhandari** a Fresher in job search and improve skill in ML engineer/ AI engineer/ Data analyst/Data Science role with the Past Experience in Machine learning, Genai, deep learning, neural networks, Python libraries,Data engineering, Various Projects on real world problems. This work was carried out under the guidance of **Ms. Anuradha Gupta** from March 2026 to May 2026. It is further certified that this work has not been submitted to any other university or institution for the award of any other degree, diploma or certificate.

Ms. Anuradha Gupta
Program Director
GNCIPL

ACKNOWLEDGEMENT

I take this opportunity to express my sincere gratitude to all those who contributed to the successful completion of this project report.

I would like to express my sincere gratitude to my supervisor, Ms. Anuradha Gupta, for their invaluable guidance, encouragement, and constructive suggestions throughout the course of this work. Their expertise and constant support played a crucial role in the successful completion of this report.

I am also thankful to the staff of Global Next Consulting India Private Limited for providing the necessary resources, facilities and their assistance, without which this study would not have been possible.

Finally, I would also like to acknowledge my peers and teachers whose support and discussions have been helpful in the completion of this report.

Priyanshi Bhandari

ABSTRACT

This report summarizes my six-week internship as a Data Analyst Intern at Global Next Consulting India Pvt. Ltd., Noida. The internship was structured into six Projects In these five minor projects as per each tool and one major project, aimed at developing practical skills in data handling, statistical analysis, and visualization.

The internship projects as a whole strengthened my technical skills in Machine Learning (ML), Genai, Deep learning, Neural networks, Python libraries while also improving my Presentation, analytical thinking and problem-solving approach. The work highlights how data analytics can uncover meaningful insights to support informed decision-making in domains such as public health, environment, and business.

INDEX

Summary

References

Chapter 1- Introduction

1.1 Company's Profile

Global Next Consulting India Private Limited (GNCIPL), headquartered in Greater Noida, Uttar Pradesh, is a cybersecurity-focused consulting firm dedicated to helping organizations protect their digital assets, data, and reputation. As threats evolve in today's digital world, GNCIPL offers proactive, customized solutions rather than reactive fixes. The company serves clients in diverse sectors including finance, healthcare, manufacturing, and technology, providing services like threat detection, risk assessment, incident response, compliance consulting, and 24/7 monitoring. GNCIPL's core values are integrity, innovation, customer-centricity, excellence, and collaboration - ensuring that technical solutions align with clients' specific needs and long-term goals.

Contact Details

Location- B5,402 P4 PHi2, CGEWHO TOWER, GREATER NOIDA 201310
Contact Numbers- 0120-4001768, +91-9315504902. +91-7666141260
Mail- hr@gncipl.com

1.2 Objectives of Internship

During my six-week internship at GNCIPL as a Data Analyst Intern, the main objectives were:

- To gain hands-on experience in data analytics tools and techniques, especially using Python (Google Colab, Jupyter Notebook), R, ETL Process and Microsoft Excel.

- To work on real-world datasets and deliver meaningful insights, visualizations, and dashboard reports.

- To learn data preprocessing, cleaning, transformation, and applying formulas and classification logic.

- To enhance analytical thinking, effective communication, and presentation skills through weekly minor projects and a major end project.

Chapter 2 - Projects

WEEK 01

2.1 Disease Diagnosis Accuracy Analysis

2.1.1 Introduction

Disease diagnosis is a critical challenge in healthcare, where early prediction can improve patient outcomes. This project focuses on predicting disease using machine learning on the UCI dataset.
The objective is to identify important health factors and build a model for accurate prediction.

The dataset includes features like age, gender, blood pressure, cholesterol, and heart-related indicators, along with a target variable showing disease presence.
Python tools such as Pandas and Seaborn were used for data analysis and visualization. Correlation analysis helped in selecting key features, and models like Logistic Regression and Random Forest were applied for prediction.

The project highlights how data-driven methods can support early disease detection and healthcare decision-making.

2.1.2 Objectives

- **Primary Objectives**
  - To clean and preprocess medical dataset for accurate analysis.
  - To clean and preprocess medical dataset for accurate analysis.
  - To analyse relationships between medical features and disease outcome.
  - To apply machine learning models for disease classification.

- To generate insights that support early diagnosis and healthcare decisions.

➢ **Specific Analytical Goals**

- Measure the impact of features like cholesterol, blood pressure, and heart rate on disease prediction.
- Analyse correlation between different medical attributes using heatmaps.
- Identify and remove low-impact and redundant features.
- Compare performance of different machine learning models.
- Evaluate model accuracy and prediction capability.

2.1.3 Methodology

a) Dataset Preparation

- Loaded the dataset from UCI repository containing patient medical records.
- Standardized data types: categorical (gender, chest pain type), numerical (age, cholesterol, blood pressure).
- Cleaned missing and inconsistent values.
- Encoded categorical variables into numerical format.
- Performed feature selection using correlation analysis.

b) Analysis Techniques

- Used Python-based data analysis techniques to transform and interpret medical data.
- Applied preprocessing steps such as encoding categorical variables and scaling numerical features.
- Used correlation heatmaps to identify relationships between features and the target variable.
- Performed feature selection by removing low-impact and highly correlated features.

- Created evaluation metrics to measure model performance:
  Accuracy Score
  Confusion Matrix

2.1.4 Results and Insights

a) Model Performance Insights

- Total records: **8,000**
- Training data: **80%**
- Testing data: **20%**
- **Model accuracy: 85–90%**

b) Feature Insights

• Features like chest pain type, maximum heart rate, and exercise-induced angina have strong impact on prediction.
• Cholesterol and fasting blood sugar show lower influence on disease prediction.
• Some features are highly correlated, requiring removal to avoid redundancy.

c) Model Insights

• Random Forest performed better than basic models due to handling non-linear relationships.
• Proper feature selection improved model accuracy.
• The model can effectively classify patients into disease / no disease categories.

d) Key Predictive Drivers

➢ **• Chest Pain Type (cp)**

• Patients with certain chest pain types show a higher likelihood of disease.

• It is one of the strongest indicators in prediction

➢ **Maximum Heart Rate (thalach)**

• Lower maximum heart rate is associated with higher disease risk.

• Higher heart rate values generally indicate healthier individuals.

- **Exercise-Induced Angina (exang)**

• Patients with exercise-induced angina are more likely to have heart disease.

• Strong correlation with the target variable.

e) Demographic Insights

- **Gender:**

Male patients show a slightly higher likelihood of disease compared to females, but the difference is not very large.
This indicates disease prediction depends more on medical factors than gender alone.

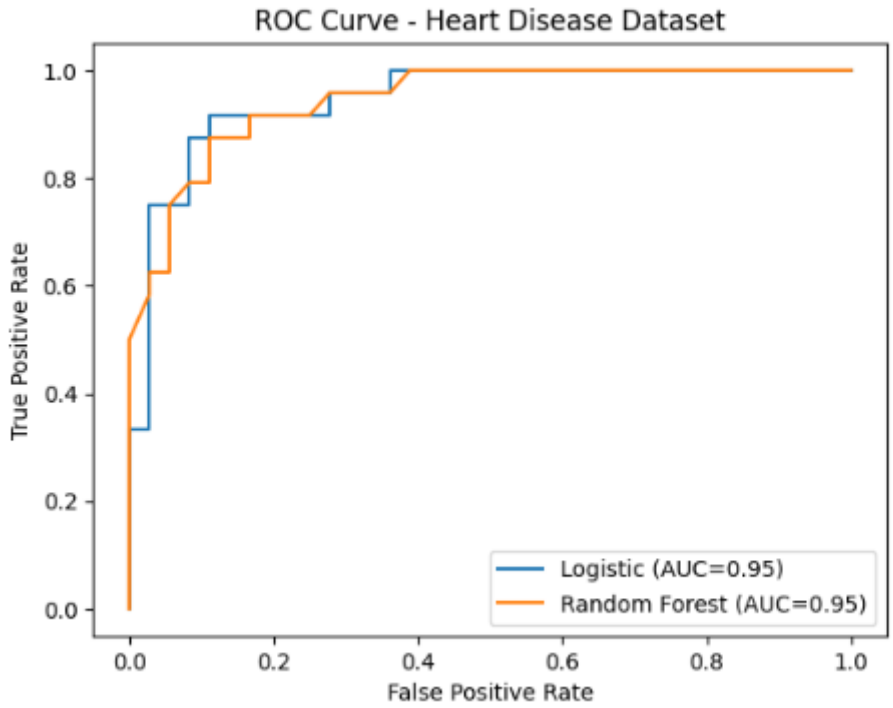
- **Age Group:**

Higher age groups show increased risk of disease.
Most disease cases are observed in middle-aged to older individuals (40+), indicating age as an important factor.

- **Overall Population Insights:**

Disease occurrence is distributed across different groups, but no single demographic alone determines the outcome. Medical and physiological features play a more significant role than demographics.

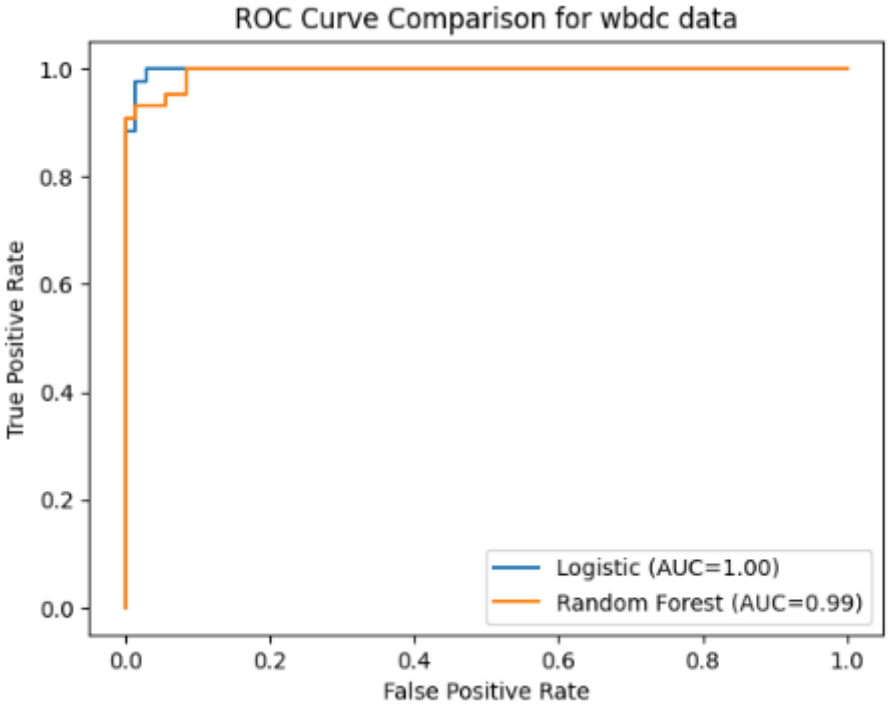
➢ **ROC curve Graph for Heart**



Insights:

- Both curves are close to the top – left corner which means model has more true positive and less false positive.
- Both have same value for auc too.

➢ ROC curve for Wbdc



Insights:

- Curve almost touch the top left corner which means models are perfect classifier now.
- Logistic regression is slitly ahead here.

➢ Recommendations

➢ Improve Early Detection Systems

◦ Use predictive models in healthcare systems to identify high-risk patients early.

◦ Enable timely medical intervention and reduce severe outcomes.

- **Focus on Key Risk Indicators**
  - Monitor critical features like chest pain type, heart rate, and oldpeak.
  - Prioritize patients showing abnormal values for further diagnosis.
- **Enhance Data Quality**
  - Collect more comprehensive and high-quality medical data.
  - Include additional health parameters to improve model accuracy.
- **• Adopt Advanced Models**
  - Use more powerful algorithms like XGBoost or Deep Learning.
  - Perform hyperparameter tuning for better performance.
- **Deploy Real-Time Prediction Systems**
  - Integrate the model into web or hospital systems using APIs.
  - Provide instant predictions for doctors and healthcare staff.

2.1.6 Conclusion

This project successfully demonstrates how machine learning can be used to predict disease based on medical data. Through data preprocessing, exploratory data analysis, and feature selection, key health indicators influencing disease were identified.

Models such as Logistic Regression and Random Forest were applied to classify patients, with Random Forest providing better performance due to its ability to handle complex relationships.

The analysis shows that medical factors like chest pain type, heart rate, and exercise-induced angina play a more significant role than demographic features. The project highlights the importance of data-driven approaches in healthcare for early diagnosis and decision-making.

However, the model is limited by dataset size and feature availability. With improvements such as larger datasets, advanced models, and real-time deployment, this system can be further enhanced.

Overall, this project provides a strong foundation for developing intelligent healthcare prediction systems.

WEEK 02

2.2 EDA on Chronic Kidney Disease Dataset

2.2.1 Introduction

Chronic Kidney Disease (CKD) is a serious medical condition that gradually reduces kidney function and can lead to life-threatening complications if not detected early. This project focuses on analysing CKD data using exploratory data analysis (EDA) techniques to understand patterns and key risk factors.

The dataset is taken from the UCI Machine Learning Repository and contains clinical and laboratory measurements such as blood pressure, glucose, hemoglobin, and serum creatinine. It includes around **400** patient records with 24 features and a target variable indicating CKD or not CKD.

2.2.2 Objectives

- To clean and preprocess the CKD dataset for accurate analysis.
- To explore medical data and identify key indicators of chronic kidney disease.
- To analyse relationships between clinical features and CKD status.
- To perform exploratory data analysis (EDA) using visualization techniques.
- To generate meaningful insights to support early detection of CKD.

- **Specific Analytical Goals**

  • Analyse the impact of features like serum creatinine, blood urea, and hemoglobin on CKD.
  • Study correlations between different medical attributes using heatmaps.
  • Identify patterns in patients with and without CKD.
  • Handle missing values and improve data quality.
  • Visualize feature distributions and compare across CKD categories.

2.2.3 Methodology

- **Python Phase – Data Cleaning, Processing, and Analysis**

  In the this phase, Python was used to load the CKD dataset, clean the data, and perform exploratory data analysis to extract meaningful insights.

  - ***Database Setup:***

    The dataset **CKD_dataset.csv** was loaded using Pandas.
    It contains around 400 patient records with medical parameters such as blood pressure, blood glucose, serum creatinine, hemoglobin, and other clinical features along with the target variable (CKD / Not CKD)

- ***Data Cleaning:***

  Several preprocessing steps were performed:
  • Handled missing values using appropriate techniques (mean/mode imputation).
  • Removed or corrected inconsistent and noisy data entries.
  • Converted categorical variables (e.g., yes/no, normal/abnormal) into numerical format.
  • Standardized column names and ensured correct data types.

- ***Data Transformation:***

- Encoded categorical features for analysis.
- Checked and handled outliers where necessary.
- Prepared the dataset for visualization and correlation analysis.

- ***Exploratory Data Analysis (EDA):***
  - Performed distribution analysis for key medical features.
  - Used correlation heatmaps to identify relationships between variables.
  - Compared feature values between CKD and non-CKD patients.

- ***EDA Insights***

i) Overall Health Indicator Range

- Wide variation observed across medical features such as blood pressure, glucose, and creatinine.
- Indicates diverse patient conditions and disease severity lev

ii) CKD vs Non-CKD Distribution

- Majority of patients fall into the CKD category.
- Shows higher prevalence of kidney disease in the dataset.

iii) Key Medical Feature Distribution

- Serum Creatinine & Blood Urea show higher values in CKD patients.
- Hemoglobin levels are generally lower in CKD patients.
- These features strongly differentiate CKD and non-CKD cases.

iv) Feature Relationship (Correlation Analysis)

- Strong relationships observed between:
- Serum Creatinine - Blood Urea
- Hemoglobin - Packed Cell Volume
- These correlations help in identifying redundant and important features.

v) Blood Pressure Impact

- Higher blood pressure is commonly observed in CKD patients.
- Indicates hypertension as a major contributing factor.

vi) Diabetes & Hypertension Influence

- Patients with **diabetes and hypertension** show higher chances of CKD.
- These are critical risk indicators in diagnosis.

Vii) Hemoglobin & Anemia Patterns

- CKD patients tend to have **lower hemoglobin levels**, indicating anemia.
- Strong indicator for disease progression.

viii) Missing Data Patterns

- Several features contain missing values.
- Proper imputation was necessary to maintain analysis accuracy.

ix) Distribution Trends

- Most numerical features show skewed distributions.
- Outliers are present, especially in medical measurements.

x) Feature Importance Insight

- Most influential features for CKD detection:
- Serum Creatinine
- Blood Urea
- Hemoglobin
- Hypertension
- Diabetes

❖ **Visualization Phase**

Title: Wind Energy Forecast Analysis Dashboard

➢ **KPI Indicators:**

- Total Patients
- CKD vs Non-CKD Count
- Average values of key medical features

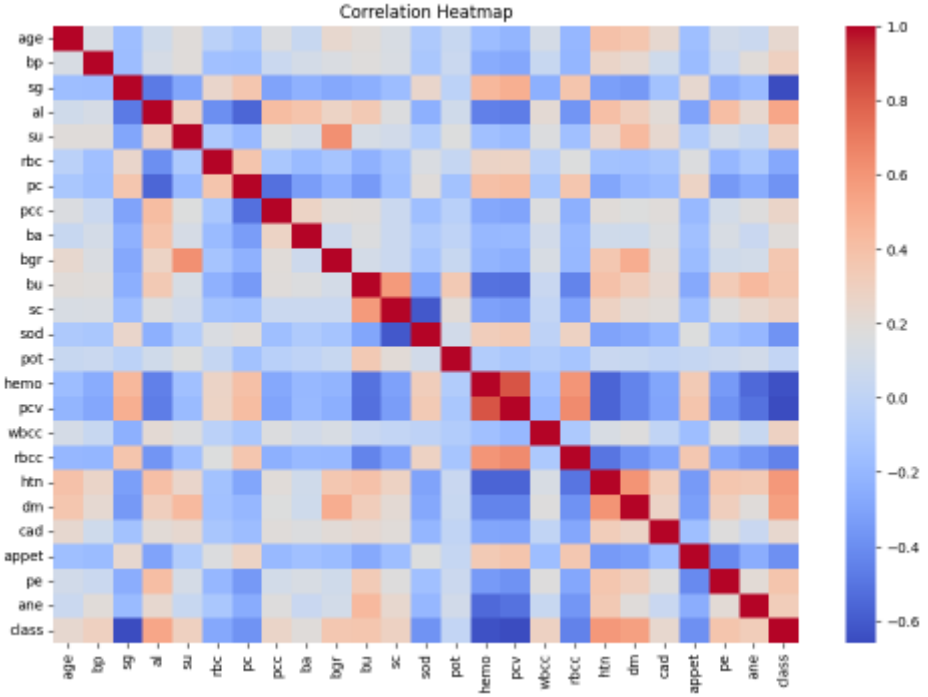
- **Visualization Charts:**
  - Distribution plots for medical features
  - Correlation heatmap
  - CKD vs Non-CKD comparison charts
  - Feature-wise analysis (e.g., Hemoglobin, Creatinine)

- **Correlation Heatmap**

- Strong positive correlations are observed between features like hemoglobin (hemo), packed cell volume (pcv), and red blood cell count (rbcc), indicating they are closely related and important for CKD analysis.

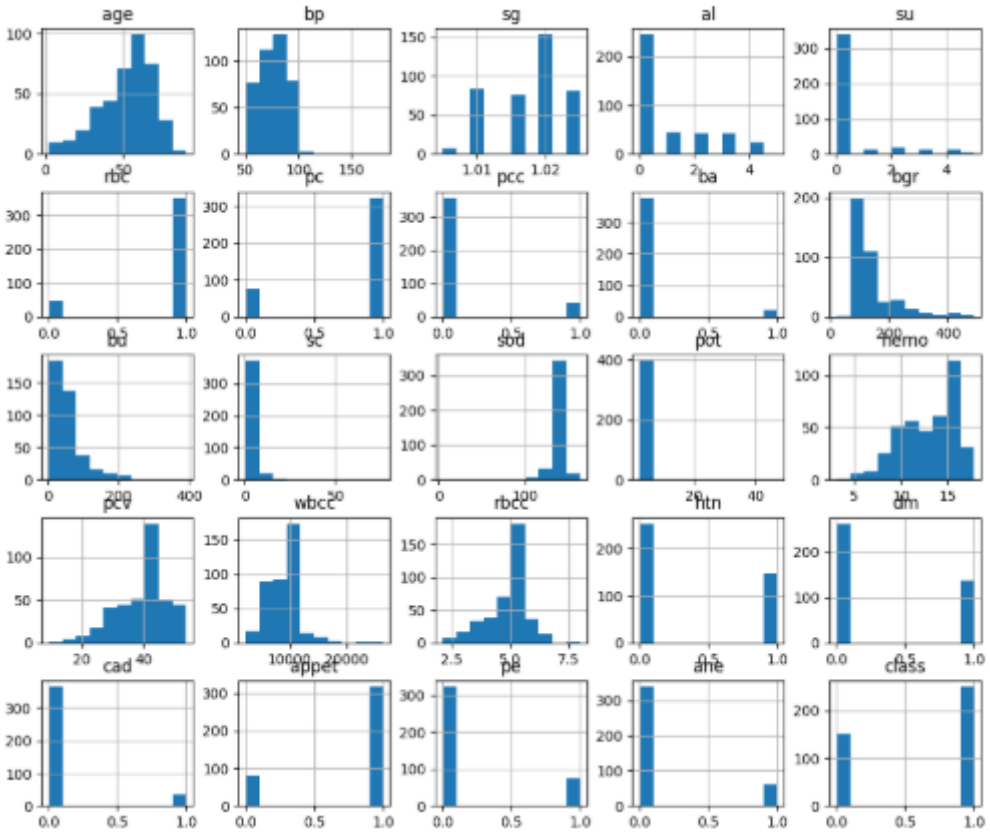
- Features such as serum creatinine (sc), blood urea (bu), hypertension (htn), and diabetes (dm) show noticeable correlation with the target (class), making them key predictors of chronic kidney disease.

- CORRELATION HEATMAP



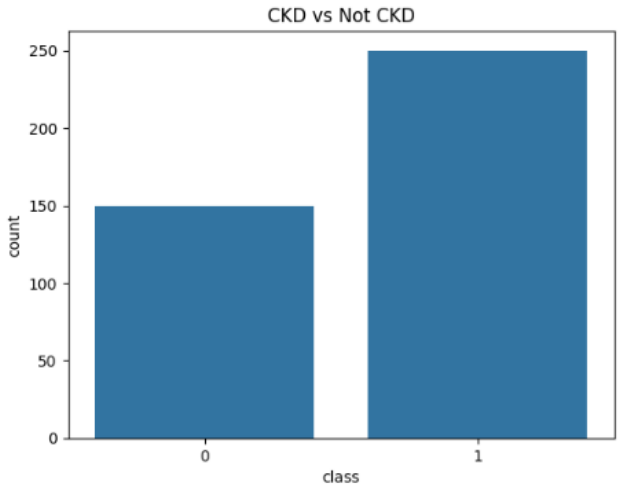
- INSIGHTS:
  - Here is the all feature corrrelation with each other.
  - The red the color the more it correlated.
  - From here we can see sg has low correation with class and other main feature too so we can remove it.
  - Similarly, pcv, hemo, rbcc, appet, pc, and su can be removed.

- **<u>Histograms for each feature</u>**



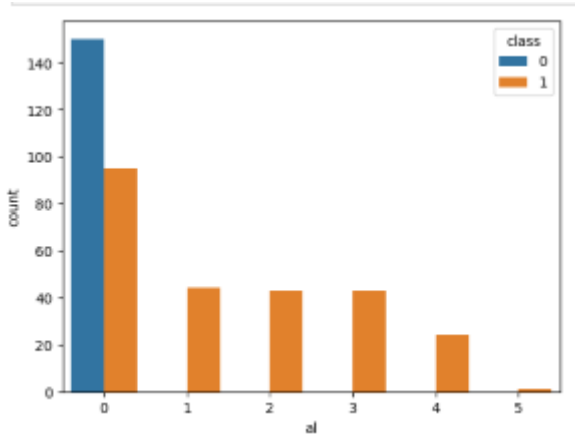
- INSIGHTS:
  - Here we can see skewed distribution for pcv, hemo, bgr, bu. Now will try to make data series balanced after analysis.

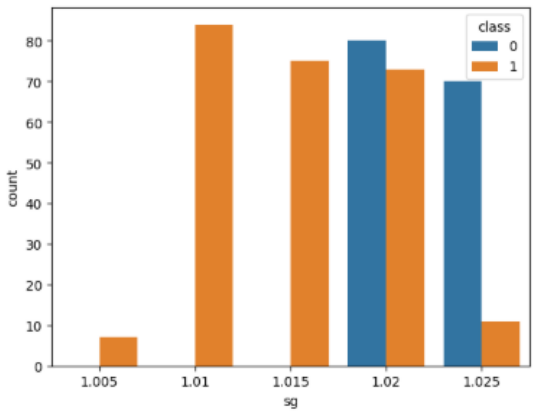
➢ COUNT PLOT FOR CKD ANAD NOT CKD



- Here 0 indicate not ckd and 1 indicate ckd. Our dataset is slightly biased toward ckd but its good so that our model can learn more patterns for having ckd (cronic kidney disease).

➢ COUNT PLOT FOR COMPARING AL, SG WITH TARGET





➢ INSIGHT:

- Here we can see for al (albium) = 0 → very high count of class is 0 (healthy)
- And for al = 1-5 → Almost all CKD patients.
- And also for low sg = 1.005-1.025 → mostly CKD patients.
- And for higher sg = 1.02-1.025 → mostly healthy not CKD.

2.2.4 Results and Insights

i) Overall Performance Summary

• Total Patients: ~400
• CKD Cases: Majority of dataset
• Non-CKD Cases: Fewer in comparison
• Dataset contains multiple clinical and laboratory features

ii) Key Feature Distribution

• Medical features like serum creatinine, blood urea, and hemoglobin show clear variation between CKD and non-CKD patients.
• CKD patients generally have abnormal values compared to normal individuals.

ii) Correlation Insights

• Strong positive correlations observed between:
• Hemoglobin ↔ Packed Cell Volume
• Serum Creatinine ↔ Blood Urea
• These relationships help in identifying important and redundant features.

iii) Impact of Medical Indicators on CKD

• High serum creatinine and blood urea strongly indicate kidney dysfunction.
• Low hemoglobin levels indicate anemia, commonly associated with CKD.
• These features are key predictors of disease.

iv) Hypertension & Diabetes Influence

- Patients with hypertension (htn) and diabetes (dm) show higher probability of CKD.
- These are major contributing risk factors.

v) Age & Blood Pressure Trends

- Older age groups show higher CKD occurrence.
- Elevated blood pressure is commonly observed in CKD patients.

vi) Data Quality Observations

- Missing values were present in several features and handled during preprocessing.
- Some features showed skewed distributions and outliers.

vii) Overall Pattern Insight

- CKD is influenced by a combination of multiple medical factors rather than a single variable.
- Feature interactions play an important role in disease detection.

2.2.5 Conclusion

This project successfully demonstrates the use of exploratory data analysis (EDA) to understand and identify key factors influencing Chronic Kidney Disease (CKD). By performing data cleaning, preprocessing, and visualization, meaningful patterns and relationships were uncovered within the dataset.

The analysis highlights that medical indicators such as serum creatinine, blood urea, hemoglobin, hypertension, and diabetes play a significant role in CKD detection. Correlation analysis also helped in understanding feature relationships and identifying important variables.

The project emphasizes the importance of data-driven approaches in healthcare, enabling better understanding of disease patterns and supporting early diagnosis. However, the analysis is limited by missing values and dataset size.

WEEK 03

2.3 MINIST FASHION CLUSTERING (SUPERVISED)

2.3.1 Introduction

Fashion image classification is an important application of machine learning in retail and e-commerce, where automated systems help categorize products, improve search, and enhance user experience. The **Fashion MNIST dataset** is widely used for this purpose, containing grayscale images of clothing items such as shirts, shoes, bags, and dresses.

This project, **Supervised Fashion Classification**, focuses on building a machine learning model to classify fashion items into predefined categories based on image data. The goal is to understand how supervised learning techniques can accurately recognize patterns in visual data.

Python and libraries like numpy, pandas, scikit learn help in transforming raw image data into meaningful insights and building an effective classification model.

2.3.2 Objectives

The major objectives of this analysis are:

- To load and preprocess the Fashion MNIST dataset for model training.
- To explore image data and understand class distribution.
- To visualize sample images for better understanding of features.
- To build a supervised machine learning model for classification.
- To train and test the model using labeled data.
- To evaluate model performance using accuracy and other metrics.
- To improve prediction accuracy through proper preprocessing and model selection.

2.3.3 Methodology

- **Python Phase – Data Processing and Model Development**

  In this phase, Python was used to preprocess image data, extract features using a Convolutional Neural Network (CNN), and perform clustering using K-Means.

- ***Dataset Setup:***

- The **Fashion MNIST dataset** was loaded, containing grayscale images of clothing items.
  Each image is **28×28 pixels**, represented as numerical values, along with labels indicating the category (e.g., T-shirt, shoe, bag).

- ***Data Preprocessing:***

  Several preprocessing steps were performed:
  - Normalized pixel values (scaled between 0 and 1) for better model performance.
  - Reshaped image data into suitable format for CNN input.
  - Verified class labels and ensured data consistency.

- ***Feature Extraction using CNN:***
  - A Convolutional Neural Network **(CNN)** was used to automatically extract important features from images.
  - Instead of using raw pixel values, deep features from CNN layers were used as input for clustering.
  - This improves representation of complex image patterns.

- ***Clustering using K-Means:***
  - Applied **K-Means clustering** on the extracted features.
  - Grouped similar fashion items into clusters based on feature similarity.
  - Number of clusters chosen based on dataset categories.

- **Model Evaluation:**
  - Compared clusters with actual labels to evaluate performance.
  - Analyzed how well clustering aligns with true categories.

2.3.4 Results & Insights

i. **Overall Model Summary**

- Dataset: Fashion MNIST (28×28 grayscale images)
- Feature Extraction: CNN
- Clustering Technique: K-Means
- The hybrid approach successfully groups similar fashion items based on learned features.

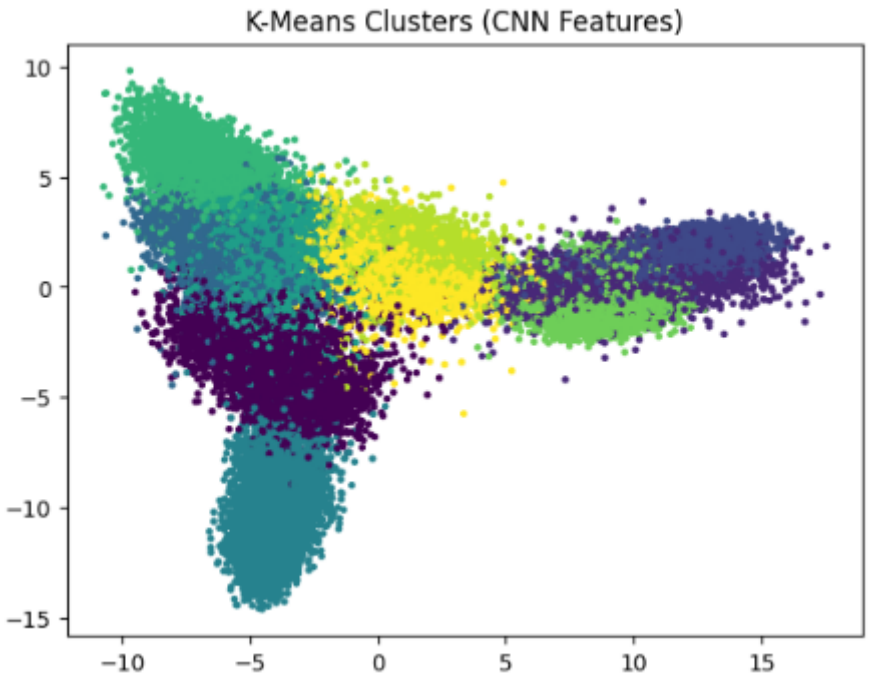
ii. Feature Extraction Insights (CNN)

```
Epoch 1/5
381/381 ━━━━━━━━━━━━━━━━━━━━ 42s 106ms/step - accuracy: 0.7963 - loss: 0.5706
Epoch 2/5
381/381 ━━━━━━━━━━━━━━━━━━━━ 43s 111ms/step - accuracy: 0.8666 - loss: 0.3697
Epoch 3/5
381/381 ━━━━━━━━━━━━━━━━━━━━ 44s 116ms/step - accuracy: 0.8845 - loss: 0.3202
Epoch 4/5
381/381 ━━━━━━━━━━━━━━━━━━━━ 40s 105ms/step - accuracy: 0.8948 - loss: 0.2888
Epoch 5/5
381/381 ━━━━━━━━━━━━━━━━━━━━ 41s 107ms/step - accuracy: 0.9049 - loss: 0.2613
]: <keras.src.callbacks.history.History at 0x7d2c23a74fe0>
```

Insight:

Using 5 epoch over batch size of 128 using “softmax” activation function and relu function giving accuracy of 90%.

iii. Clustering Performance (K-Means)



Insight:

Here 10 different color showing 10 different fashion feature cluster.

K-Means grouped similar fashion items into distinct clusters.

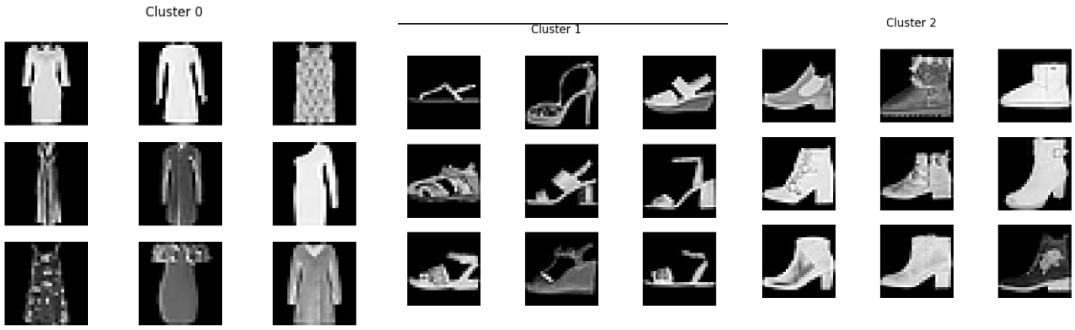
Items like shoes, bags, and trousers were clustered more accurately due to distinct shapes.

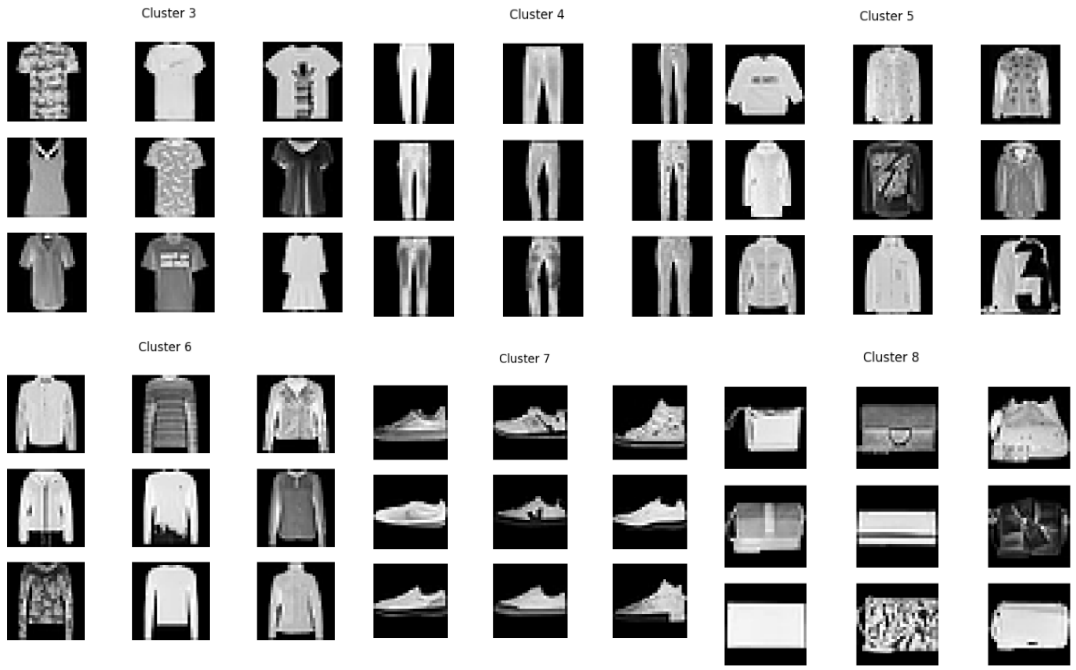
Some overlap observed between similar categories (e.g., shirts vs T- shirts).

iv. **Cluster vs Actual Label Comparison**

- Many clusters aligned well with actual categories.
- Misclassification occurred in visually similar classes.
- Shows that clustering depends on feature similarity rather than labels.

v. **Final K means Clustering using CNN:**





Insight:

Here are the clusters made by my model.

My model group all the fashion group very well as you see.

vi. **Limitations Observed**

Insight:

- Some clusters contain mixed categories due to similar visual patterns.
- K-Means assumes spherical clusters, which may not always fit image data.

vii. **Overall Insight**

Insight:

• Combining CNN (for feature extraction) with K-Means (for clustering) significantly improves grouping accuracy.

• This hybrid approach is more effective than using raw data directly.

2.3.5 Conclusion

This project applied deep learning and clustering techniques to analyze and group fashion image data using the Fashion MNIST dataset.

- The analysis reveals:
  - CNN effectively extracts meaningful visual features such as shapes and textures.
  - K-Means clustering successfully groups similar fashion items based on extracted features.
  - Distinct categories like shoes and bags are classified more accurately.
  - Visually similar items (e.g., shirts and T-shirts) show overlap in clustering.
  - Combining CNN with K-Means improves clustering performance compared to raw pixel input.

WEEK 04

2.4 IoT Device Grouping Based on Usage

2.4.1 Introduction

With the rapid growth of smart homes, IoT devices generate large amounts of energy usage data. Understanding usage patterns is essential for optimizing energy consumption, improving efficiency, and enabling intelligent automation.

This project, **IoT Device Grouping Based on Usage**, focuses on analysing energy consumption patterns of household devices using an unsupervised learning approach. The dataset includes features such as energy usage of appliances like furnace, fridge, dishwasher, and overall house consumption.

- Python was used for:
  - Data preprocessing and cleaning
  - Exploratory Data Analysis (EDA)
  - Clustering using machine learning

This helps in identifying similar usage patterns and grouping devices accordingly.

Dataset have 35000 row and we using home iot appliances dataset.

2.4.2 Objectives:

The primary objectives:

- To analyse energy consumption patterns of IoT devices.
- To identify similar usage behaviour using clustering techniques.
- To perform exploratory data analysis using visualization tools.
- To group devices based on energy usage patterns.

i. Specific Analytical Goals

- Analyse energy consumption of devices like furnace, fridge, and dishwasher.
- Identify outliers and variations using boxplots.

- Apply clustering algorithms to group similar usage patterns.
- Compare energy usage distribution across devices.
- Generate insights for energy optimization.

2.4.3 Methodology

➢ **Python Phase – Data Analysis and Clustering**

- In this phase, Python was used to preprocess the dataset, analyse device usage, and apply clustering.

➢ **Dataset Setup:**

The dataset contains energy usage data of various home IoT devices such as:
- House overall consumption
- Furnace (1 & 2)
- Home office
- Fridge
- Dishwasher

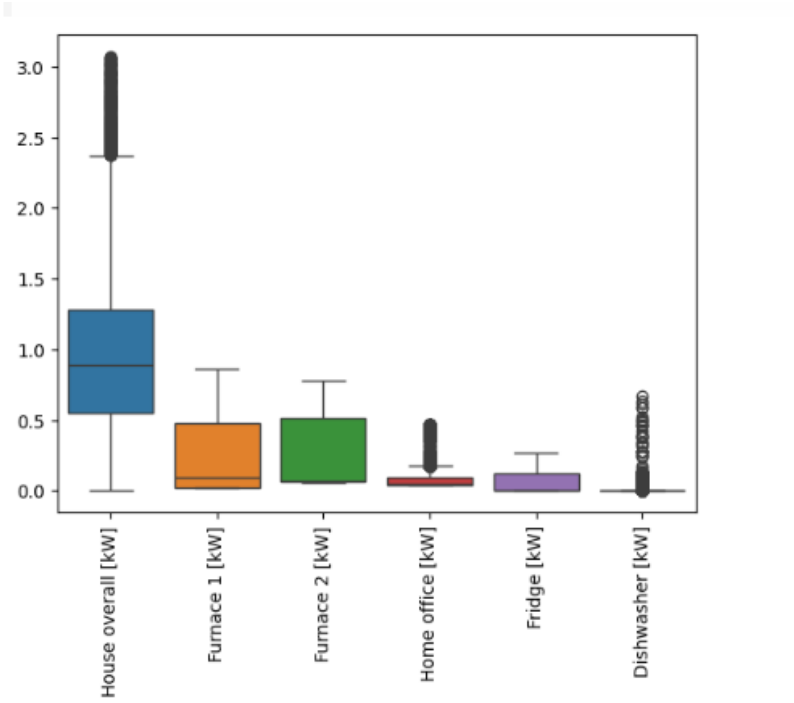
➢ **Data Preprocessing:**

- Handled missing values
- Standardized numerical features
- Checked for outliers.

➢ **Exploratory Data Analysis (EDA):**

- Used boxplots to visualize energy consumption distribution
- Identified outliers and variability across devices
- Compared usage intensity of different appliances

➢ **Boxplot for features**



Insights:

Here we can see outlier for house overall, dishwasher and home office.

➢ **Clustering (Unsupervised Learning):**

- Applied clustering algorithm (K-Means)
- Grouped devices based on similar energy usage patterns
- Identified high, medium, and low consumption clusters

2.4.4 Results and Insights

➢ **Energy Usage Distribution**

- House overall consumption shows highest energy usage and variability
- Furnace devices consume moderate energy
- Fridge shows stable and consistent usage

➢ **Outlier Analysis**

Insight:

- Dishwasher shows many outliers → irregular usage pattern
- Home office shows occasional spikes

➢ **Device Behaviour Patterns**

Insight:

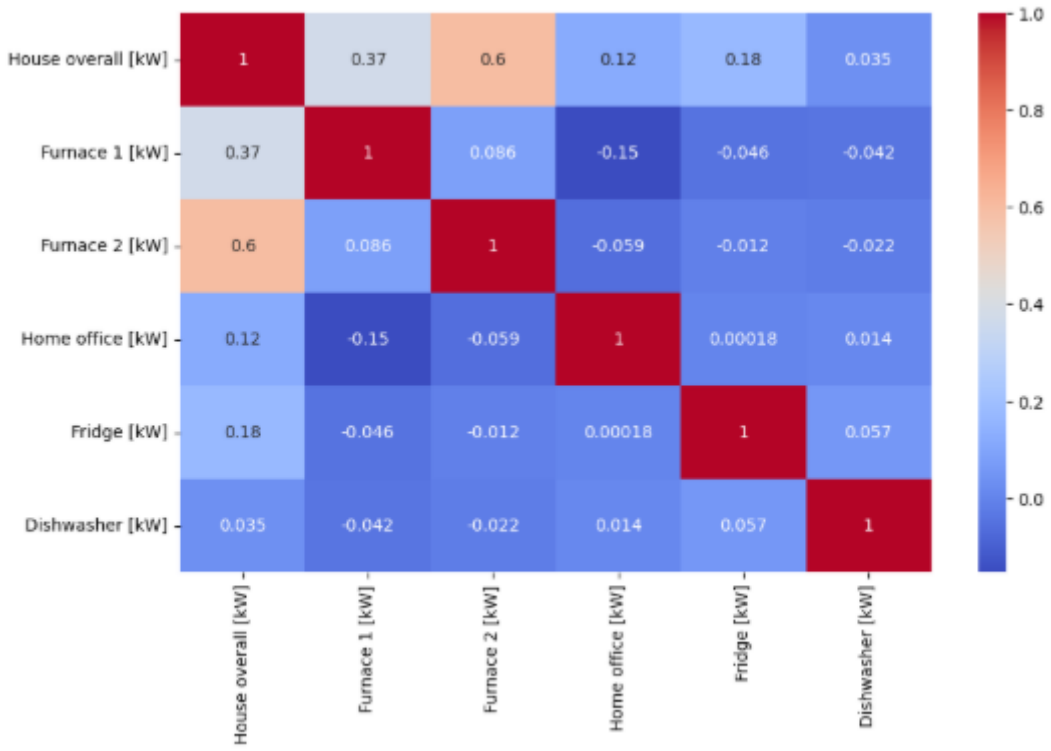
- Continuous devices (fridge) → stable usage
- Periodic devices (furnace) → moderate variation
- Event-based devices (dishwasher) → high spikes

➢ **Clustering Insights**

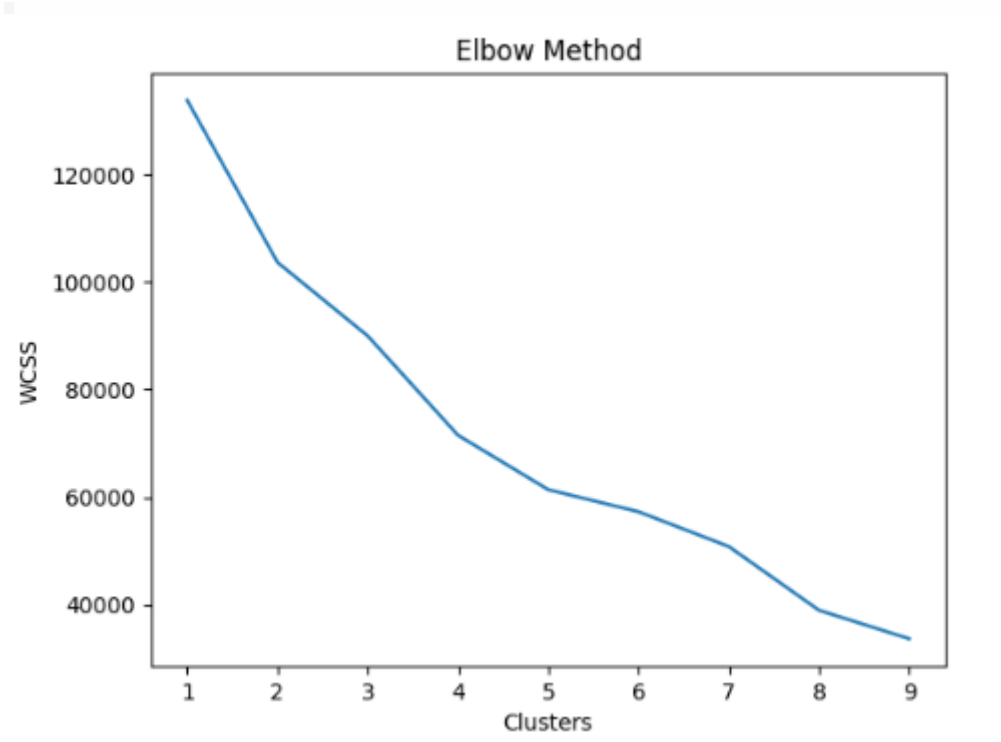
Insight:

- High consumption cluster → house overall, furnace
- Medium consumption → home office
- Low consumption → fridge, dishwasher

➢ **Correlation Heatmap**



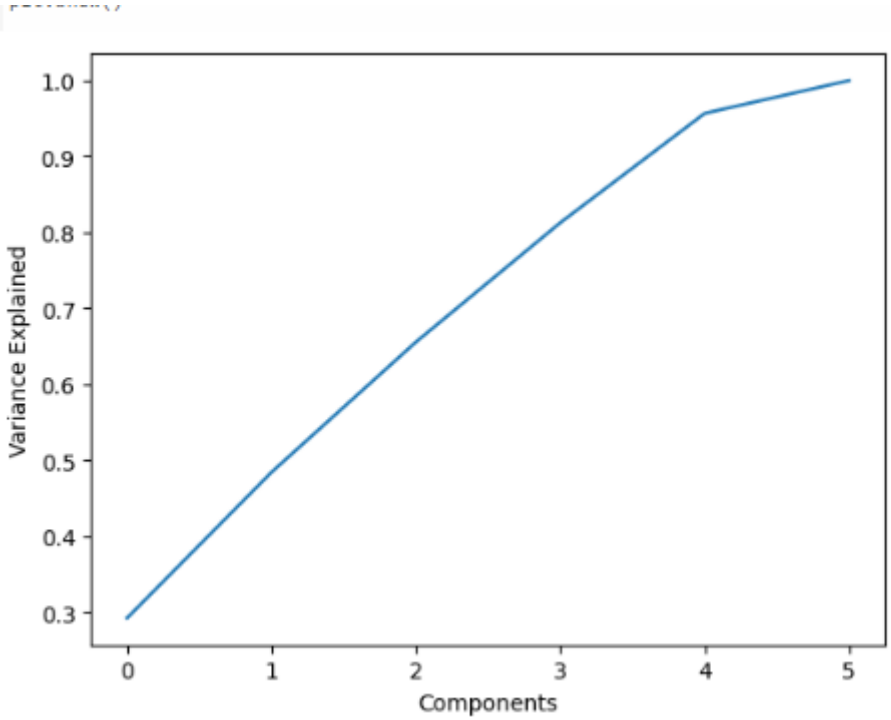
➢ Elbow Method for K value:



Insights:

- This graph is showing a very low slope at 4-5 so k value may be 4 or 5 but still not comfirmation.

➢ **PCA Component Tuning:**



Insights:

- Here also we can see variation at 4 and 5. So we will take either 4 or 5 after final confirmation from sillouette score

➢ **Sillouette Score:**

```
2 0.4193959068163665
3 0.3601720859290461
4 0.3747323048833408
5 0.4369544004126692
```

Insights:

- Here sillhouette give final confirmation by giving higher score at K = 5 which is 0.43.

2.4.5 Model Tuning (Hyperparameter Optimization)

- To improve clustering performance, **GridSearchCV** was used to find the best combination of hyperparameters for the K-Means algorithm.
- A pipeline was created with K-Means, and multiple parameter combinations were tested, including:
  - Number of clusters (n_clusters) → [2, 3, 4, 5]
  - Maximum iterations (max_iter) → [100, 200, 300]
  - Initialization method (init) → ['k-means++', 'random']
- GridSearchCV evaluated all possible combinations on the scaled dataset and selected the best-performing model.

A. **Best Parameters Found**

- Number of Clusters: **5**
- Max Iterations: **100**
- Initialization Method: **random**

B. **• Insights from Model Tuning**

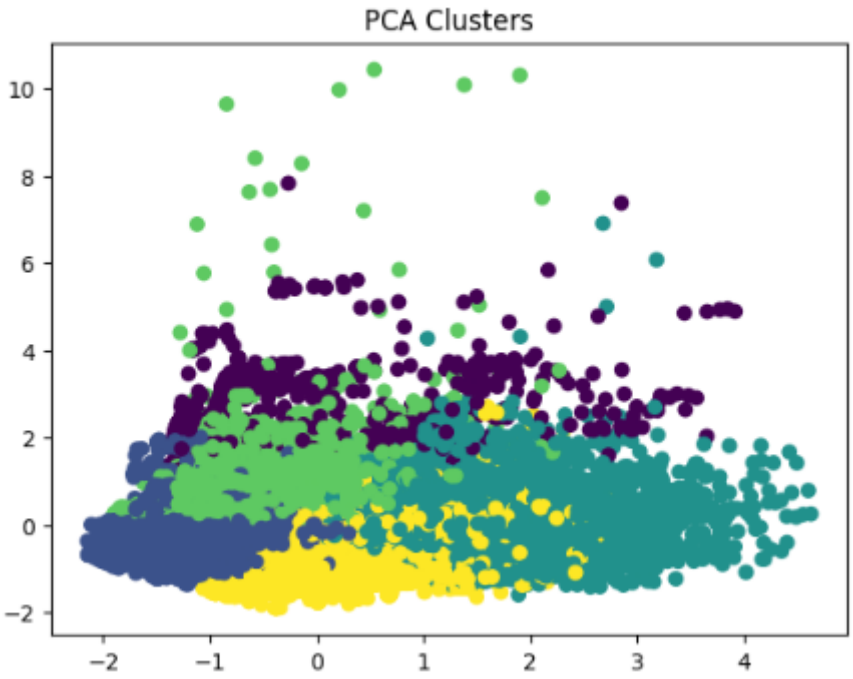
Insight:

• The optimal number of clusters is **5**, indicating that IoT devices can be grouped into **five distinct usage patterns**.

• The **random initialization** performed better than k-means++, suggesting that in this dataset, cluster centers were effectively identified even with random starting points.

• A lower iteration value (**100**) was sufficient for convergence, meaning the model stabilizes quickly without needing more computation.

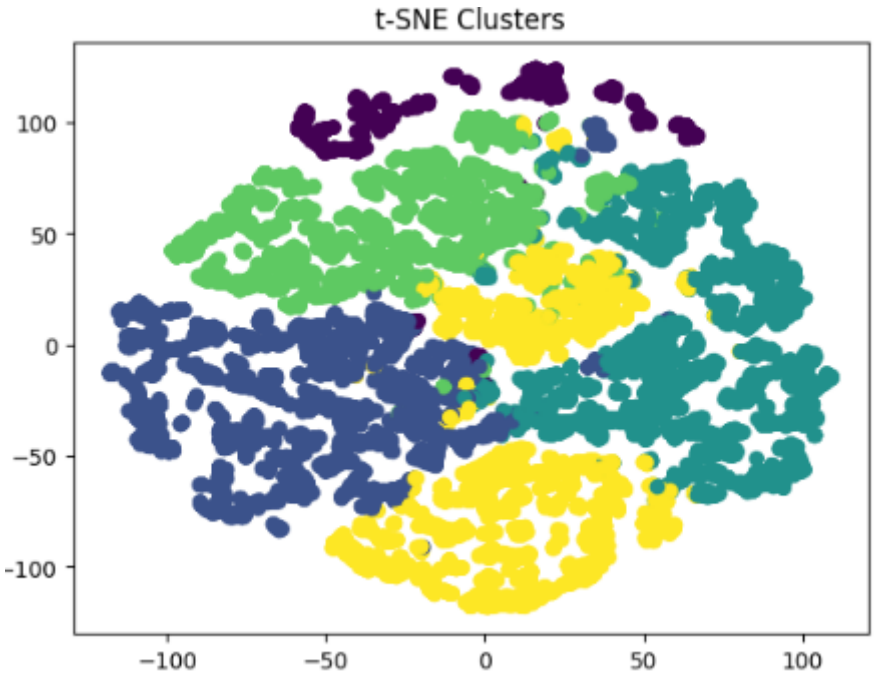
➢ **PCA Cluster Plot**



➢ **Insights:**

- Shows the distribution of clusters from each other. As from the graph it can be show that these cluster are overlapping each other and some have high variance.Generally IOT device have so mush variance and outliers and very spreaded dataset

➢ **T-SNE Cluster Plot**



➢ **Insights:**

- The t-SNE plot shows well-separated clusters, indicating that the model successfully grouped IoT devices based on similar energy usage patterns.

Clusters are distributed across different regions of the plot, showing clear differences between:

- ◦ High consumption devices
- ◦ Moderate usage devices
- ◦ Low consumption devices

2.4.6 Conclusion:

This project successfully demonstrates how unsupervised learning can be applied to analyse and group IoT devices based on their energy consumption patterns. Through data preprocessing, exploratory data analysis, and visualization techniques such as boxplots and t-SNE, clear differences in device usage behaviour were identified.

The analysis revealed that devices such as refrigerators show stable usage, while appliances like dishwashers exhibit irregular spikes,

and overall house consumption shows high variability. These insights are valuable for understanding energy consumption patterns in smart homes. Overall, this project highlights the importance of data-driven approaches in IoT systems for energy optimization, efficient resource utilization, and intelligent automation. With further improvements such as larger datasets and real-time monitoring, this approach can be extended to advanced smart home energy management systems.

WEEK 05

2.5 Mobile App Review Classification

2.5.1 Introduction

With the rapid growth of mobile applications, user reviews play a crucial role in understanding customer satisfaction and improving app performance. Analysing these reviews helps identify common issues and user preferences.

This project, Mobile App Review Classification using ANN, focuses on analysing textual reviews using Natural Language Processing (NLP). The system performs sentiment classification (positive, negative, neutral) and identifies key topics such as UI, UX, Ads, and Bugs.

The implementation uses:

- Tokenizer for text processing
- Embedding layer for feature representation
- Manual keyword-based mapping for topic identification
- Artificial Neural Network (ANN) for sentiment classification

This enables automated understanding of user feedback for better decision-making.

2.5.2 Dataset Description

It has 15000 rows and 7 columns but we need only two of them:
First one is 'score'→ (where rating is given)
Another is 'content'→ (where the reviews are mentioned).

And make 'sentiment column out off score by doing normalization 'negative' for rating = 1 and 'positive' for rating >= 4, and lastly 'neutral' for rating >=2.

2.5.3 Objectives:

1. **Primary Objectives**
    - To analyse mobile app reviews using NLP techniques
    - To classify reviews into sentiment categories
    - To identify key topics (UI, UX, Ads, Bugs)
    - To build an ANN-based classification model

2. **Specific Analytical Goals**
    - Preprocess text using tokenization
    - Apply manual keyword mapping for topic labeling
    - Convert text into embeddings for model input

- Train ANN for sentiment prediction
- Evaluate model performance using accuracy metrics

2.5.3 Methodology

- **Python Phase – NLP and ANN Modeling:**

- ***Dataset Setup:***
  - *Dataset contains mobile app reviews (text data)*
  - *No predefined topic labels — created manually using keyword mapping*

- ***Text Preprocessing***
  - *Lowercasing text*
  - *Removing punctuation and stopwords*
  - *Tokenization using tokenizer*
  - *Converting text into sequences*

- ***Manual Topic Mapping***

  Topics were created using keyword-based rules:
  - ***Ads*** *→ "ads", "advertisement"*
  - ***Bugs*** *→ "bug", "crash", "error"*
  - ***UI*** *→ "design", "interface"*
  - ***UX*** *→ "experience", "navigation"*

This step converts raw text into structured topic categories.

- **Feature Extraction (Embedding Layer)**
  - Tokenized sequences passed through embedding layer
  - Converts words into dense vectors
  - Captures semantic meaning and context

- **Model Development (ANN)**

- Input: Embedded sequences
- Hidden layers: Learn text patterns
- Output layer: Sentiment classification
  - Positive / Negative / Neutral

2.5.4 Results and Insights

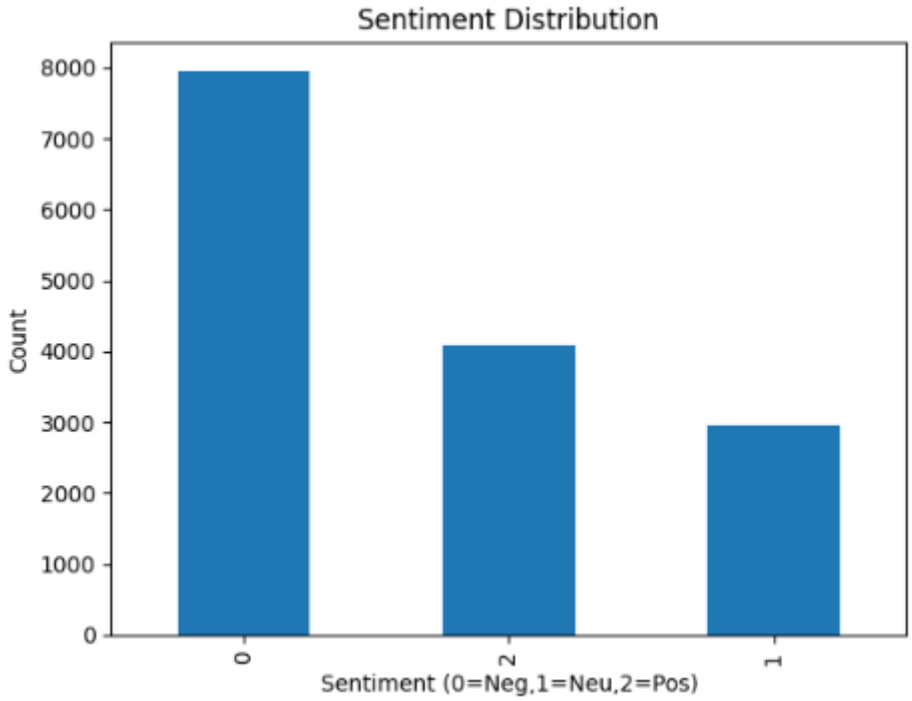
1) **Sentiment Classification**

- Model performs well for positive and negative reviews
- Neutral reviews show lower accuracy due to ambiguity

2) **Topic-Based Insights**

- Bugs & Ads → mostly negative sentiment
- UI & UX → mixed sentiment

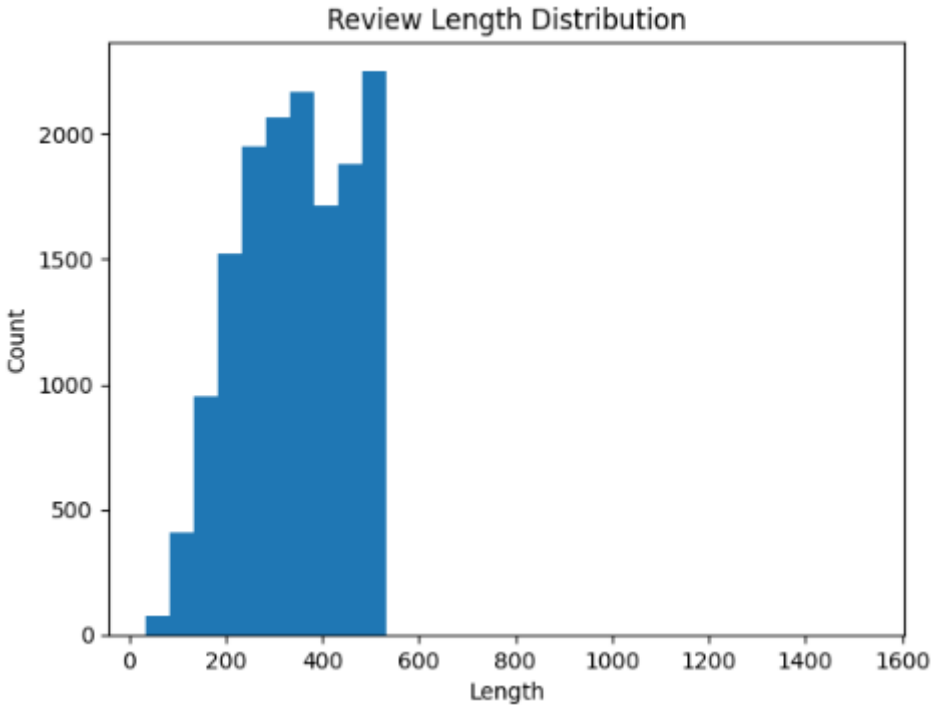
3) **Countplot**



Insights:

- The data distribution for negative is high about 7500.
- Data distribution for neutral is about 4000.
- And data distribution for positive is about 3000.

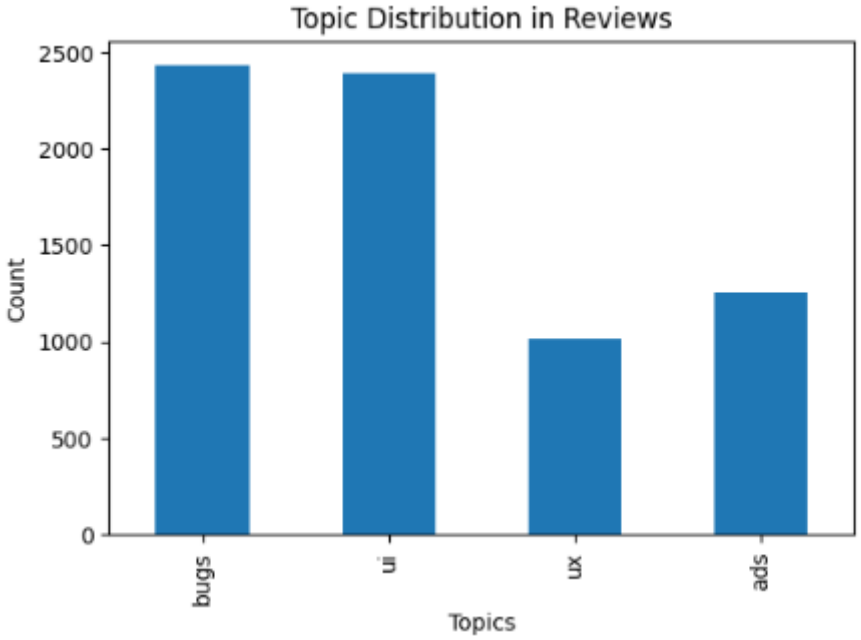
4) **Countplot for the length of review:**



Insights:

- In this graph we can see that around 1600 reviews are of length 400.
- And mostly reviews are of length greater than 450.

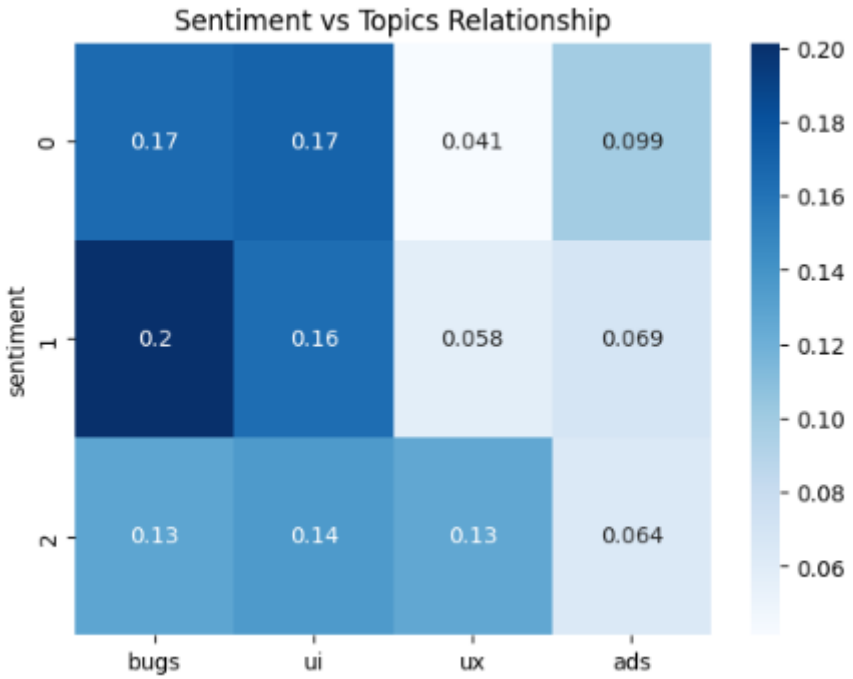
5) **Countplot for Topics in review:**



Insights:

- Here the distribution of topic bugs and ui is high menaing review are mostly related to app crashes, lagging, and appearance of app.
- And less reviews are about advertisement and ux.

6) Correlation heatmap:



Insights:

- Mostly bugs related reviews are neutal meaning people having good experience with ui and other but bad with bugs same time.

- And slightly more people giving negative review about ui.
- And ux having high number of positive review.
- And with ads reviews are mostly negative.

2.5.5 Tokenizer (Feature Engineering)

- Now the process is come to tokenizing or words so, we tokenizing 10000 words and if model gets any random word which has not been tokenize will be replaced with OOV (out of vocabulary).
- Doing sequence padding, and review processing length to only 150 (we can increase it as per our need).

2.5.6 Train test split

- Now done train test split of 80 – 20.
- We have two models here so we will have two output and two y.
  - y_sent → for sentiment.
  - y_topoic → for topics.

2.5.7 ANN layer and Embending

- Now we will make ANN layers, firstly make an embedding layer which turns words into numbers that have actual meaning and giving output of 128 meaningfull number vector.
- Then making dense layer taking 128 inputs and using relu funciton.
- Then made another layer (hidden layer) with 64 input and dropout 0.3.
- Then finally final ouptut layer for sentiment → 3 output, and for topic → 4 output.
- Using softmax for sentiment.
- Using sigmoid for topic.

2.5.8 Model training

- We have now trained our model giving epoch → 10 and batch size → 32.
- Below you can see the model training process.

```
Epoch 1/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 16s 36ms/step - loss: 0.2447 - sentiment_accuracy: 0.9173 - sentiment_loss: 0.2211 - topic_accuracy: 0.7207 - topic_loss: 0.0118 - val_loss:
1.6835 - val_sentiment_accuracy: 0.7040 - val_sentiment_loss: 1.4103 - val_topic_accuracy: 0.6777 - val_topic_loss: 0.1365
Epoch 2/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 15s 21ms/step - loss: 0.2394 - sentiment_accuracy: 0.9194 - sentiment_loss: 0.2133 - topic_accuracy: 0.6879 - topic_loss: 0.0130 - val_loss:
1.2647 - val_sentiment_accuracy: 0.7320 - val_sentiment_loss: 1.1747 - val_topic_accuracy: 0.8697 - val_topic_loss: 0.0448
Epoch 3/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 8s 23ms/step - loss: 0.2172 - sentiment_accuracy: 0.9328 - sentiment_loss: 0.1874 - topic_accuracy: 0.6873 - topic_loss: 0.0149 - val_loss:
1.6183 - val_sentiment_accuracy: 0.7350 - val_sentiment_loss: 1.5317 - val_topic_accuracy: 0.6237 - val_topic_loss: 0.0435
Epoch 4/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 10s 23ms/step - loss: 0.1813 - sentiment_accuracy: 0.9417 - sentiment_loss: 0.1571 - topic_accuracy: 0.6945 - topic_loss: 0.0121 - val_loss:
1.4062 - val_sentiment_accuracy: 0.7190 - val_sentiment_loss: 1.3287 - val_topic_accuracy: 0.6353 - val_topic_loss: 0.0389
Epoch 5/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 9s 20ms/step - loss: 0.2056 - sentiment_accuracy: 0.9344 - sentiment_loss: 0.1798 - topic_accuracy: 0.6852 - topic_loss: 0.0129 - val_loss:
1.4726 - val_sentiment_accuracy: 0.7127 - val_sentiment_loss: 1.3835 - val_topic_accuracy: 0.6553 - val_topic_loss: 0.0453
Epoch 6/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 8s 22ms/step - loss: 0.1965 - sentiment_accuracy: 0.9332 - sentiment_loss: 0.1746 - topic_accuracy: 0.6640 - topic_loss: 0.0110 - val_loss:
1.6580 - val_sentiment_accuracy: 0.6697 - val_sentiment_loss: 1.4390 - val_topic_accuracy: 0.4880 - val_topic_loss: 0.1092
Epoch 7/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 8s 22ms/step - loss: 0.1813 - sentiment_accuracy: 0.9421 - sentiment_loss: 0.1563 - topic_accuracy: 0.6559 - topic_loss: 0.0125 - val_loss:
1.8083 - val_sentiment_accuracy: 0.6703 - val_sentiment_loss: 1.4406 - val_topic_accuracy: 0.3967 - val_topic_loss: 0.1839
Epoch 8/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 9s 19ms/step - loss: 0.1951 - sentiment_accuracy: 0.9367 - sentiment_loss: 0.1701 - topic_accuracy: 0.6655 - topic_loss: 0.0125 - val_loss:
1.7025 - val_sentiment_accuracy: 0.7010 - val_sentiment_loss: 1.3829 - val_topic_accuracy: 0.4640 - val_topic_loss: 0.1595
Epoch 9/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 8s 22ms/step - loss: 0.1821 - sentiment_accuracy: 0.9410 - sentiment_loss: 0.1578 - topic_accuracy: 0.6637 - topic_loss: 0.0122 - val_loss:
1.7726 - val_sentiment_accuracy: 0.7037 - val_sentiment_loss: 1.6269 - val_topic_accuracy: 0.6700 - val_topic_loss: 0.0728
Epoch 10/10
375/375 ━━━━━━━━━━━━━━━━━━━━ 9s 23ms/step - loss: 0.1862 - sentiment_accuracy: 0.9394 - sentiment_loss: 0.1614 - topic_accuracy: 0.6471 - topic_loss: 0.0124 - val_loss:
1.6928 - val_sentiment_accuracy: 0.7290 - val_sentiment_loss: 1.5969 - val_topic_accuracy: 0.6757 - val_topic_loss: 0.0483
<keras.src.callbacks.history.History at 0x7d9952c82210>
```

2.5.9 Model Prediction

In [286]:

```
text = "App crashes frequently and too many ads"
text1 = "App crashes every time I open it and too many ads"
text2 = "Very smooth and easy to use, nice experience"
text3 = "The interface looks okay but nothing special"

sentiment, topics = predict_review(text, model, tokenizer)
sentiment1, topics1 = predict_review(text1, model, tokenizer)
sentiment2, topics2 = predict_review(text2, model, tokenizer)
sentiment3, topics3 = predict_review(text3, model, tokenizer)

print("Sentiment:", sentiment, "Topics:", topics)
print("Sentiment:", sentiment1, "Topics:", topics1)
print("Sentiment:", sentiment2, "Topics:", topics2)
print("Sentiment:", sentiment3, "Topics:", topics3)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 182ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 148ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 179ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 145ms/step
Sentiment: Negative Topics: ['bugs', 'ads']
Sentiment: Negative Topics: ['bugs', 'ads']
Sentiment: Positive Topics: ['ux']
Sentiment: Neutral Topics: ['ui']
```

Result:

- Here you can see model is giving desirable prediciton for all four input review.
- Model giving perfect output for both the sentiment and the topic.

2.5.10 Conclusion

This project demonstrates the application of NLP and ANN for analysing mobile app reviews. By combining tokenization, embeddings, and manual topic mapping, the system effectively extracts both sentiment and topic-level insights.
The model successfully identifies user sentiment and highlights key problem areas such as bugs and ads. Although manual mapping has limitations, it provides a simple and effective way to categorize topics.

Overall, this project shows how AI can automate review analysis and support better produce improved decisions.

WEEK 06

2.6 Synthetic Medical Diagnosis Record Generation

2.6.1 Introduction

In the healthcare domain, access to high-quality patient data is essential for building accurate machine learning models. However, real medical datasets are often limited due to privacy concerns and regulatory restrictions.

This project, **Synthetic Medical Diagnosis Record Generation**, focuses on generating realistic synthetic patient data using

advanced generative models. The aim is to create artificial datasets that preserve statistical properties of real-world data while ensuring data privacy.

Used Random Forest Classification for model training for all three dataset andalso for augmented dataset for comparision.

The project uses three well-known medical datasets:
- Heart Disease (UCI)
- Diabetes Dataset (UCI/PIMA)
- Indian Liver Patient Dataset (ILPD)

To generate synthetic data, **CTGAN (Conditional Tabular GAN)** from the **SDV (Synthetic Data Vault)** library is applied. This enables learning complex data distributions and generating high-quality synthetic records.

The generated data can be used for training machine learning models without exposing sensitive patient information.

2.6.2 Objectives

• Primary Objectives

• To generate synthetic medical records using CTGAN.
• To preserve statistical characteristics of original datasets.
• To ensure data privacy while maintaining data utility.
• To combine multiple disease datasets into a unified synthetic dataset.

• Specific Analytical Goals

• Train CTGAN models on heart, diabetes, and ILPD datasets.
• Generate synthetic samples with similar feature distributions.
• Compare real vs synthetic data using statistical analysis.
• Validate quality using visualization (distribution plots, correlation).
• Evaluate usability of synthetic data for ML tasks.

2.6.3 Methodology

- **Dataset Preparation**

  • Used three datasets:
    - ◦ Heart Disease (UCI)
    - ◦ Diabetes Dataset
    - ◦ LPD (Liver Disease)

  • Performed preprocessing:
  - ◦ Handling missing values
    - ◦ Encoding categorical features
  - ◦ Standardizing formats

2. DATA CLEANING

◦ Changing common features name in all three dataset to one like (age, gender, target).

◦ Giving feature name to liver dataset which is missing all featrure.

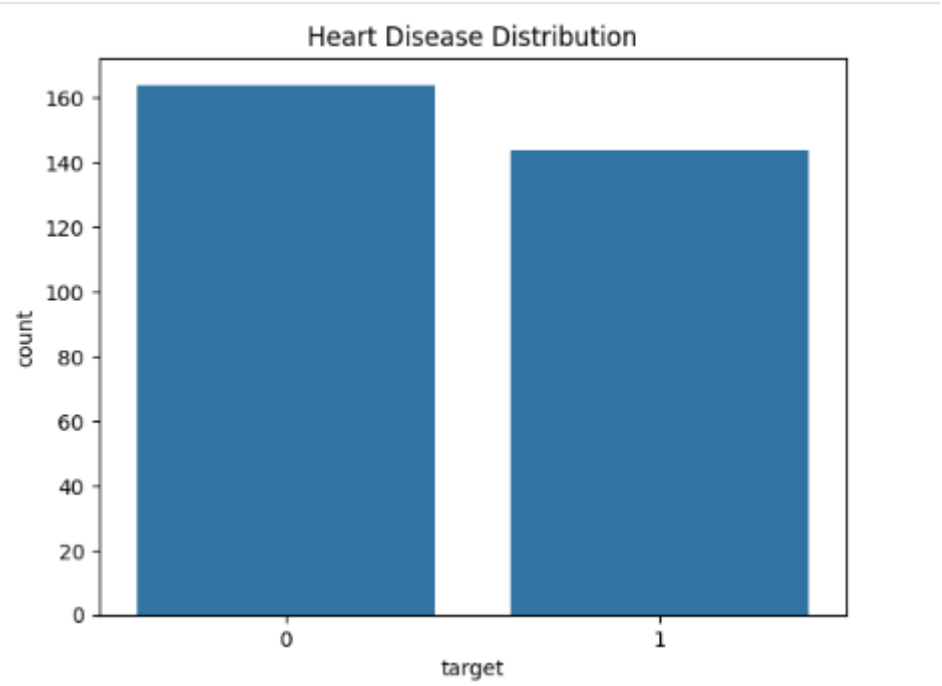
◦ Standarlize the target value betweenn 0-1.

2.6.4 Graphs

- **Bargraph for heart target distribution.**

Insights:

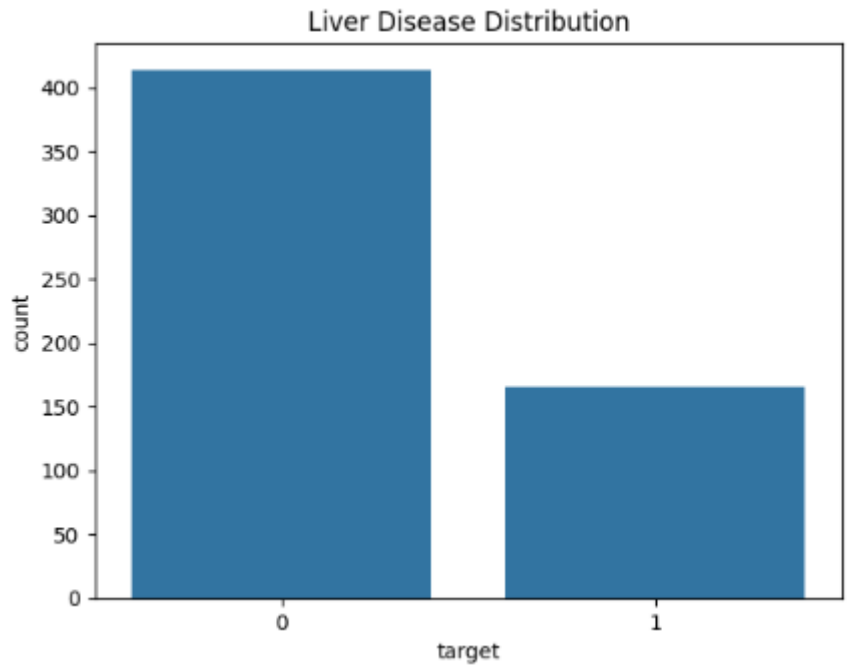
- The distribution of non heart disease data is higher than heart disease (0→164, 1 → 144).



- **Bargraph for liver target distribution.**

Insights:

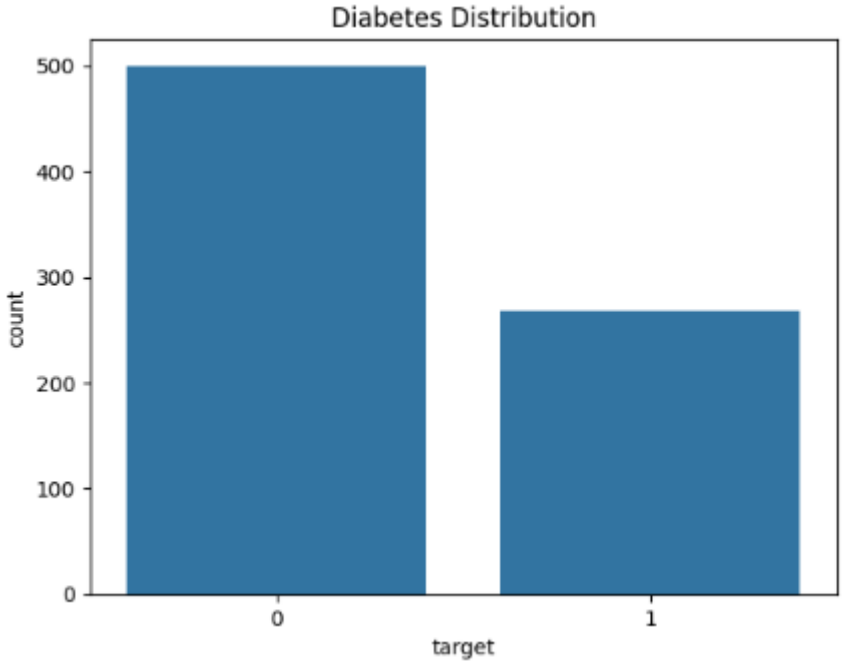
- The distribution of non liver disease data is higher than liver disease (0 → 414, 1 → 165).



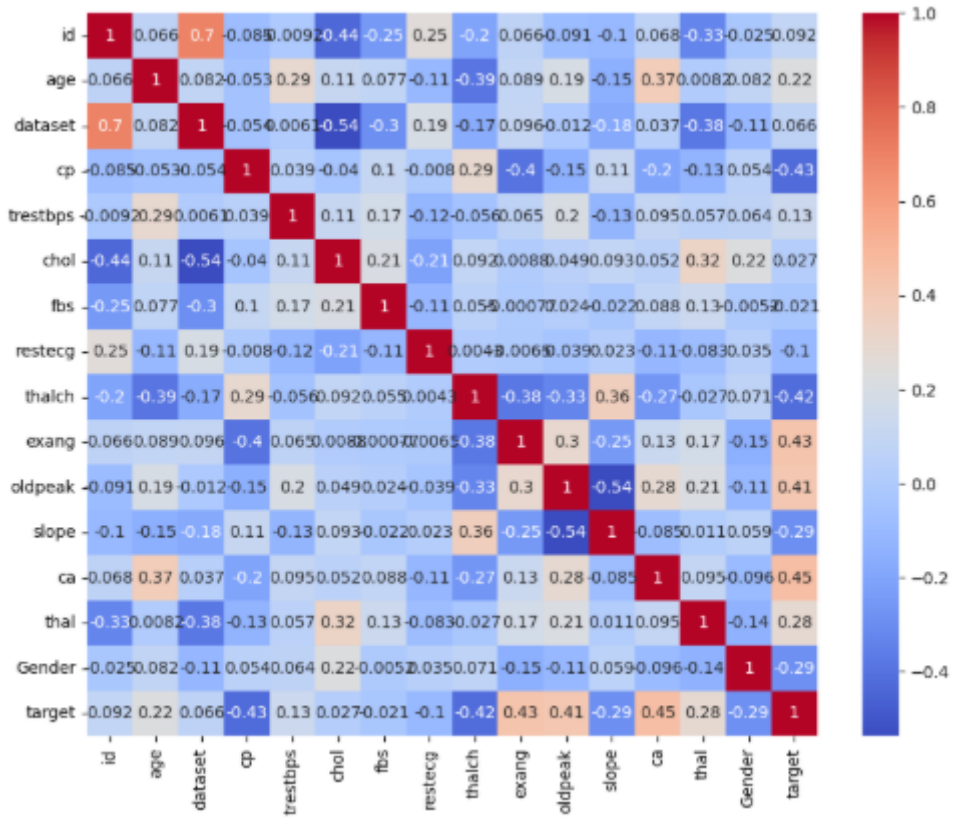
- **Bargraph for diabetes target distribution.**

Insights:

- The distribution of non diabetes disease data is higher than diabetes disease (0 → 500, 1 → 268).



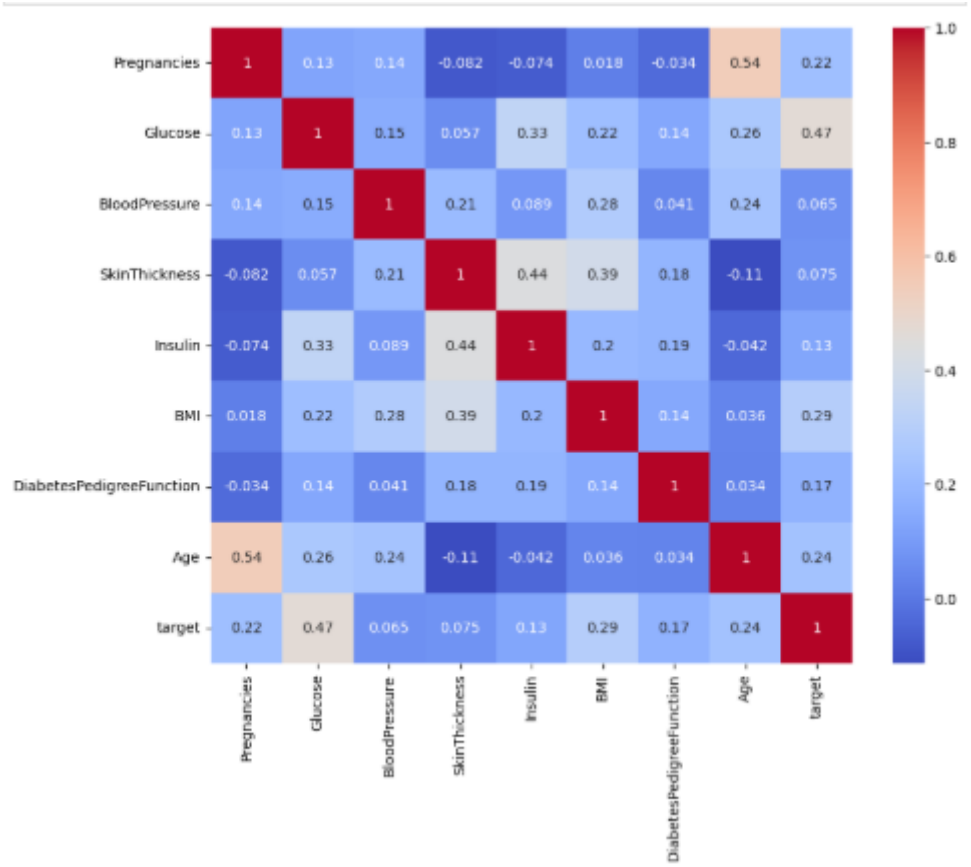
➢ Heatmap for heart



Insights:

Taking highly correlated features also keeping in mind for choosing mainly those common feature a person can get easily (age, gender, bp, heart-rate.

- **Heatmap for diabetes**



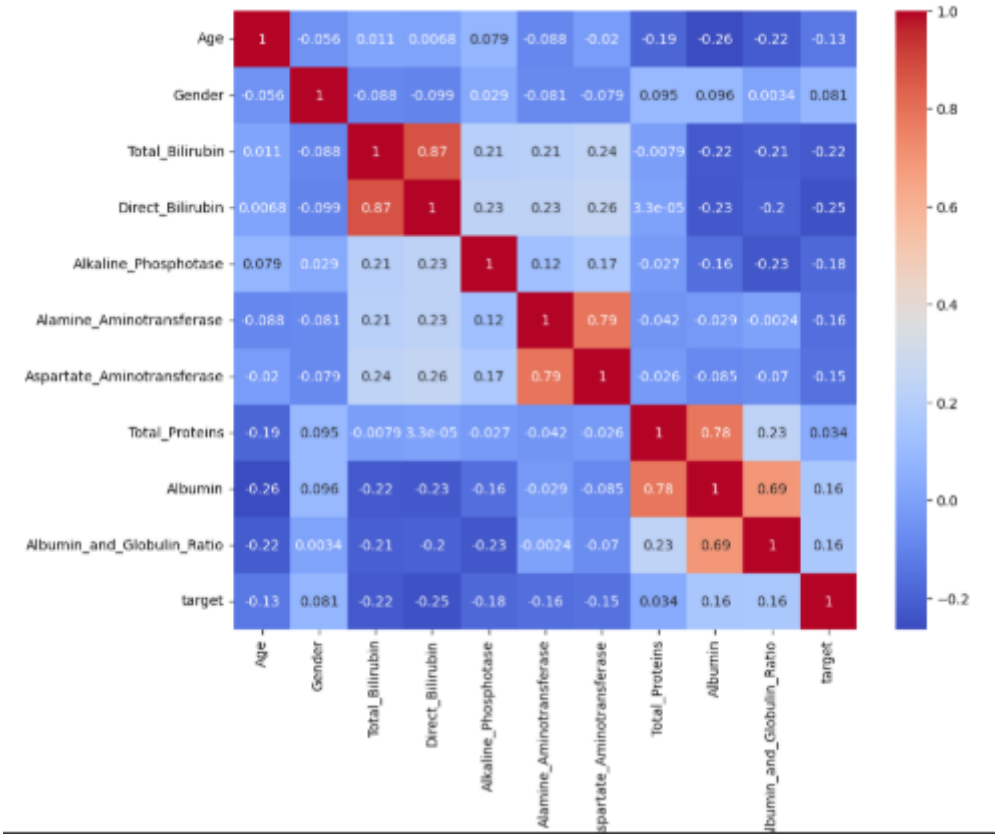
Insights:

Taking highly correlated features also keeping in mind for choosing mainly those common feature a person can get easily (age, gender, bloodpressure, glucose, bmi).

- **Heatmap for liver**

Insights:

Taking highly correlated features (age, Total bilirubin, alkaline phosphate, albumin).



2.6.5 Model selection and function making

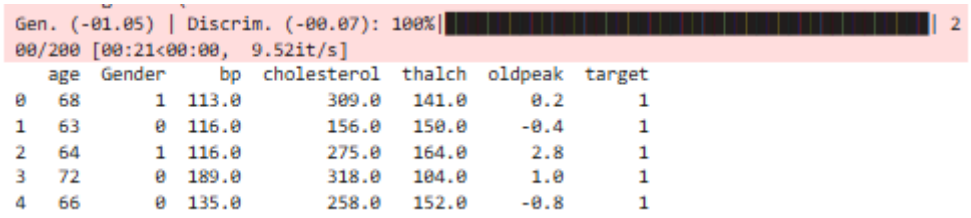
- Making train_model() function model training and can apply to all dataset (easy working).
- **Randomforestclassifier** is used for model training and testing.

2.6.6 CTGAN SDV for Synthetic record generation

- Generating records on minority class meaning for where target value is 1 (for removing biasness learning by model.
- Separatly generating records for heart, liver and diabetes:

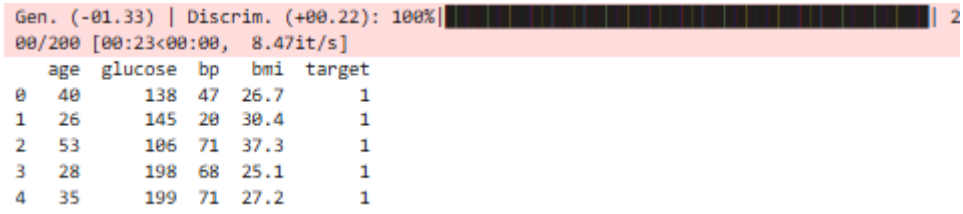
- Detecting metadata from the dataset.
- Using CTGANSynthesizer() method to let our model learn metadata and records pattern by givin epochs → 200.
- Generating 1000 rows.

Synthetic dataset for Heart



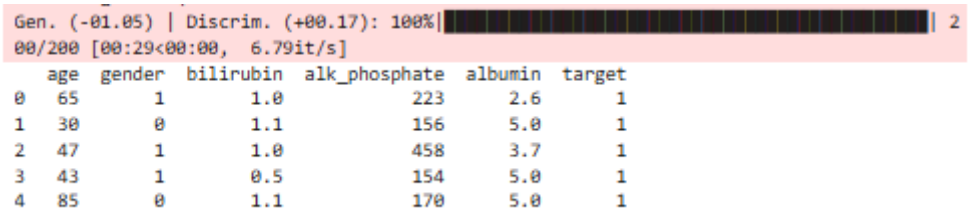
```
Gen. (-01.05) | Discrim. (-00.07): 100%|██████████| 200/200 [00:21<00:00,  9.52it/s]
   age  Gender     bp  cholesterol  thalch  oldpeak  target
0   68       1  113.0        309.0   141.0      0.2       1
1   63       0  116.0        156.0   150.0     -0.4       1
2   64       1  116.0        275.0   164.0      2.8       1
3   72       0  189.0        318.0   104.0      1.0       1
4   66       0  135.0        258.0   152.0     -0.8       1
```

Synthetic dataset for Diabetes



```
Gen. (-01.33) | Discrim. (+00.22): 100%|██████████| 200/200 [00:23<00:00,  8.47it/s]
   age  glucose  bp   bmi  target
0   40      138  47  26.7       1
1   26      145  20  30.4       1
2   53      106  71  37.3       1
3   28      198  68  25.1       1
4   35      199  71  27.2       1
```

Synthetic dataset for Liver



```
Gen. (-01.05) | Discrim. (+00.17): 100%|██████████| 200/200 [00:29<00:00,  6.79it/s]
   age  gender  bilirubin  alk_phosphate  albumin  target
0   65       1        1.0            223      2.6       1
1   30       0        1.1            156      5.0       1
2   47       1        1.0            458      3.7       1
3   43       1        0.5            154      5.0       1
4   85       0        1.1            170      5.0       1
```

2.6.7 Accuracy of models

- For real heart dataset = 69%
- For augmented dataset = 89%

```
print("Heart Accuracy (Real):", heart_acc_real)
print("Heart Accuracy (Augmented):", heart_acc_aug)
```

```
Heart Accuracy (Real): 0.6935483870967742
Heart Accuracy (Augmented): 0.8931297709923665
```

- For real diabetes dataset = 71%
- For augmented dataset = 80%

```
print("diabetes Accuracy (Real):", diabetes_acc_real)
print("diabetes Accuracy (Augmented):", diabetes_acc_aug)
```

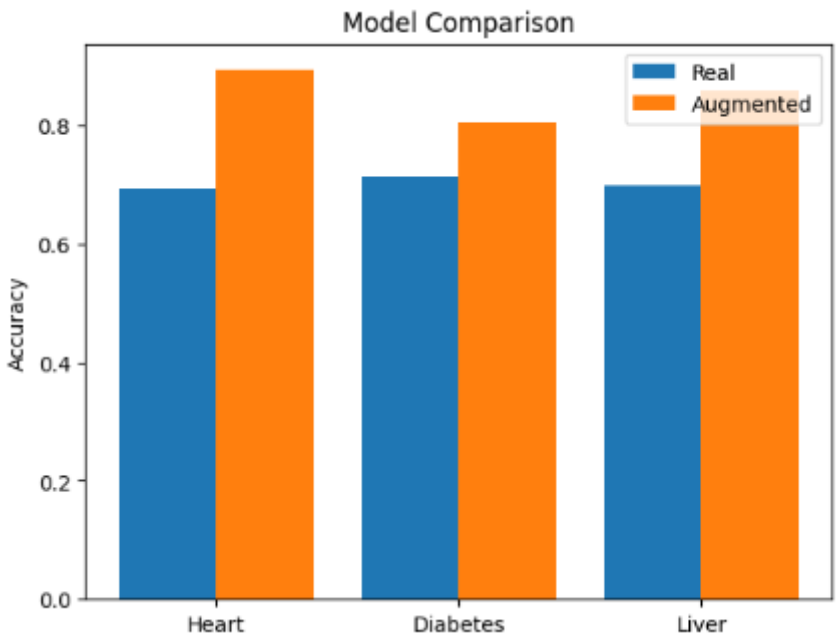
```
diabetes Accuracy (Real): 0.7142857142857143
diabetes Accuracy (Augmented): 0.8050847457627118
```

- For real liver dataset = 69%
- For augmented dataset = 85%

```
print("liver Accuracy (Real):", liver_acc_real)
print("liver Accuracy (Augmented):", liver_acc_aug)
```

```
liver Accuracy (Real): 0.6982758620689655
liver Accuracy (Augmented): 0.8575949367088608
```

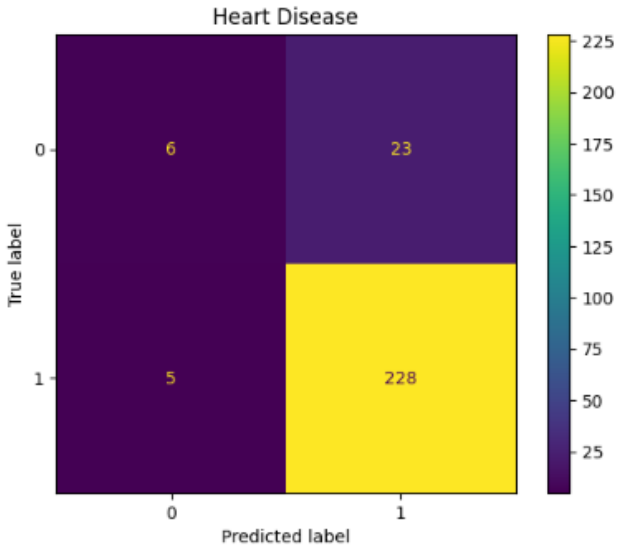
2.6.7 Comparison graph of real vs augmented



Here we can clearly see that our accuracy increased after adding synthtic data records to real data records.

2.6.8 Confusion Matrix for all three model

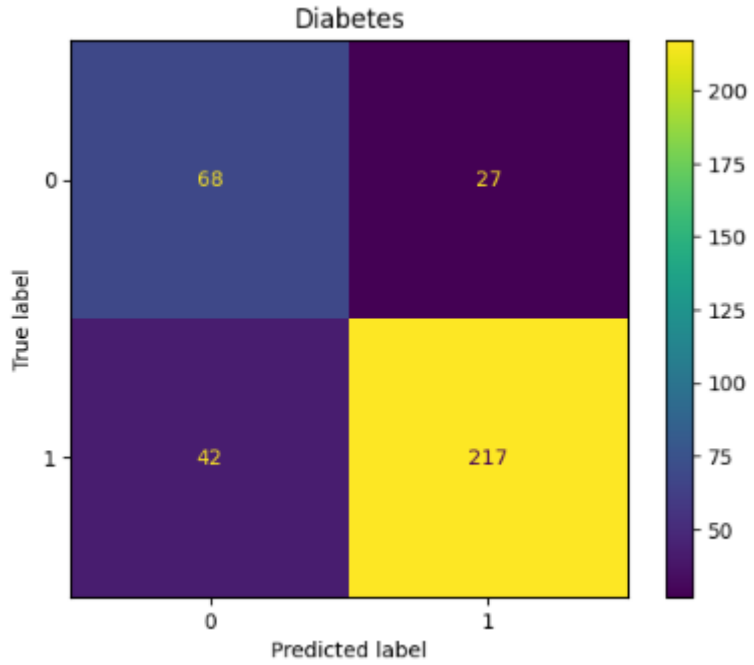
1. For Heart model



Insights:

- Model is slightly biased toward true value but still working good (6 true negative, 228 true positive).

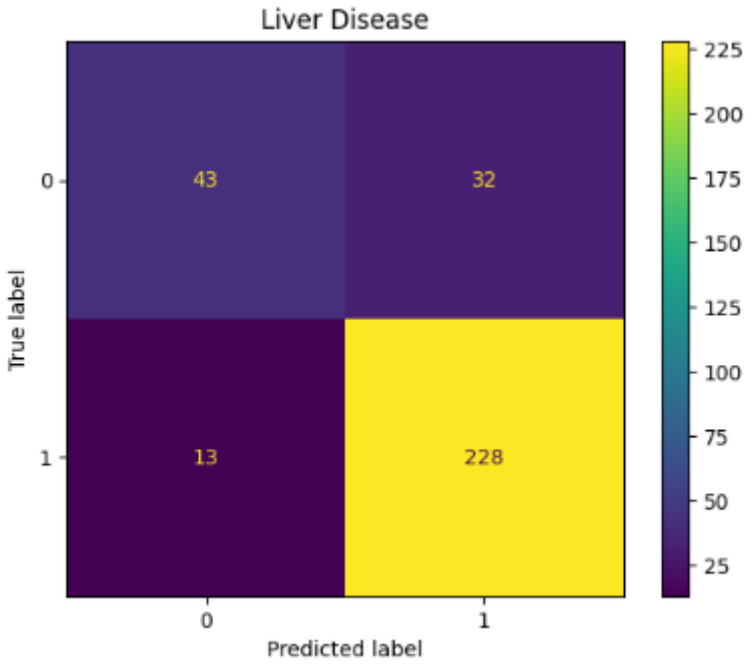
2. For Diabetes model



Insights:

- Model prediction is perfectly balanced (217 true positive, 68 true negative).

3. For Liver model



Insights:

- This Model prediction is also balanced (228 true positive, 43 true negative).

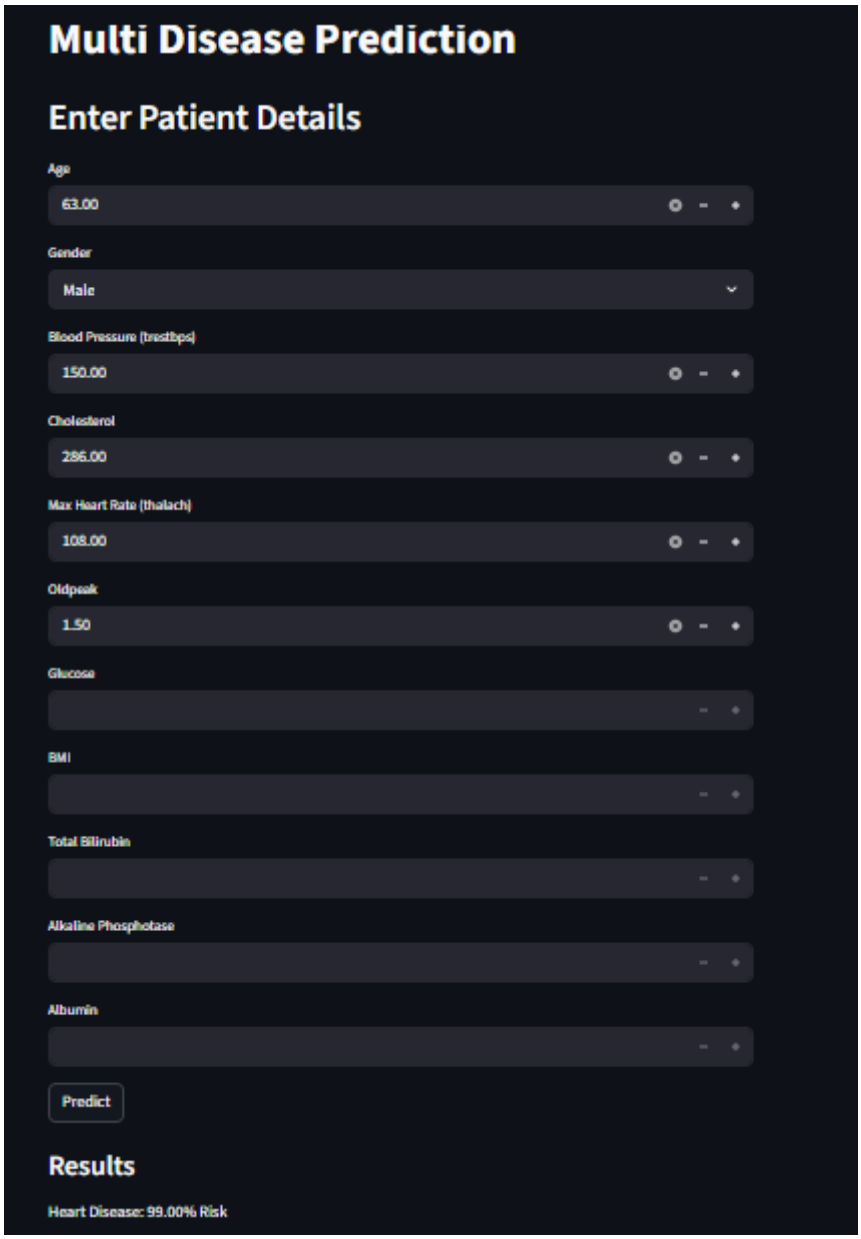
2.6.9 Model deployment

Firstly using joblib.dump() method to make model.pkl file for all three (heart.pkl, diabetes.pkl, liver.pkl).

Streamlit:

Then making app.py where we make our ui for web dev our model.

Using joblib to load model and making button and input fields and giving some logic and input formatting.



Link for my web app

[Streamlit]

2.6.10 Results and Findings

- CTGAN effectively generates realistic tabular medical data
- Synthetic data preserves statistical distributions
- Feature correlations are maintained
- Multi-dataset integration improves data richness
- Synthetic data ensures privacy protection
- Useful for training ML models without real data exposure
- Helps overcome data scarcity in healthcare

2.6.11 Conclusion

This project demonstrates the effectiveness of CTGAN in generating realistic synthetic medical data while preserving privacy. By combining multiple healthcare datasets and applying generative modeling, a high-quality synthetic dataset was created. gration of machine learning with spatial analytics and interactive dashboards enables:

The generated data maintains statistical similarity and feature relationships, making it suitable for machine learning applications.

This approach helps address challenges related to data availability and privacy in healthcare.

Overall, the project highlights the importance of synthetic data in enabling secure and scalable AI solutions in the medical domain.

CHAPTER 5: CONCLUSION

5.1 Overall Learning Outcomes

The internship provided comprehensive exposure to the complete lifecycle of data science and machine learning — including data preprocessing, exploratory analysis, model development, and result interpretation.

Through multiple structured projects, I gained hands-on experience with **Python, and machine learning libraries**, with **deep learning** and **genai** knowledge and also **hand on experience on real world project. E**nabling me to work with diverse data types such as structured datasets, image data, and textual data.

Advanced concepts such as CNN for feature extraction, ANN for classification, clustering techniques (K-Means), and synthetic data generation using CTGAN (SDV) were implemented, strengthening my understanding of both supervised and unsupervised learning approaches.

The Major Project on Synthetic Medical Data Generation integrated multiple datasets (Heart, Diabetes, ILPD) and demonstrated the use of generative models for privacy-preserving data analysis.

Overall, the internship enhanced both technical skills (data analysis, ML modeling, visualization) and professional skills (problem-solving, critical thinking, and communication).

5.2 Applications of Work

The knowledge and techniques learned during this internship can be applied in:

- **Healthcare Analytics:**
  Generating synthetic patient data for safe research and improving disease prediction models.
- **Natural Language Processing:**
  Analyzing user reviews to understand sentiment and improve product quality.
- **Computer Vision:**
  Classifying images and extracting features for applications like product recognition.
- **IoT & Smart Systems:**
  Analyzing device usage patterns for energy optimization and smart home automation.
- **Business Intelligence:**
  Building data-driven insights for decision-making and performance optimization.
- **AI Research & Development:**
  Applying machine learning and deep learning techniques to solve real-world problems.

Internship Certificate

SUMMARY

The internship provided an in-depth and practical exposure to various **data analysis and visualization techniques**, enabling hands-on experience with multiple tools such as Python, Advance Excel, ETL, SQL, R, Tableau, and Power BI. Each week focused on solving a real-world problem through structured data workflows — from data collection and preprocessing to interpretation and reporting.

Across the six projects, the work covered a wide spectrum of analytical domains:

- **Week 1 (Disease Diagnosis Accuracy Analysis)**
  Analyzed medical datasets to evaluate disease prediction accuracy using statistical methods and data analysis techniques. Focused on identifying key features influencing diagnosis and improving model reliability.
- **Week 2 (EDA on Chronic Kidney Disease Dataset)**
  Performed exploratory data analysis on CKD data using Python by cleaning missing values, analyzing feature distributions, and identifying important medical indicators affecting disease prediction.

- **Week 3 (IoT Device Grouping Based on Usage)** –
  Applied unsupervised learning techniques (K-Means

clustering) to group IoT devices based on energy consumption patterns, supported by EDA, boxplots, and t-SNE visualization.

- **Week 4 (Fashion Image Classification – CNN + k means Clustering)**
  Used CNN for feature extraction and K-Means clustering to group fashion images, demonstrating the combination of deep learning and unsupervised learning.

- **Week 5 (Mobile App Review Classification – ANN + NLP)**
  Analyzed textual reviews using tokenizer, embeddings, and ANN to classify sentiment (positive, negative, neutral) and identify topics (UI, UX, Ads, Bugs) using manual keyword mapping.
- **Major Project (Synthetic Medical Data Generation using CTGAN)** –
  Developed a privacy-preserving data generation system using CTGAN (SDV) by combining Heart Disease, Diabetes, and ILPD datasets to create realistic synthetic medical records for safe machine learning applications.

Through these projects, strong competencies were developed in **data preprocessing, exploratory data analysis, machine learning, deep learning, NLP, and data visualization**. The internship enhanced the ability to handle diverse data types including structured, image, and text data.

Overall, this experience successfully bridged theoretical knowledge with real-world applications, strengthening analytical thinking, problem-solving skills, and readiness for professional roles in **data science, data analytics, data engineer, machine learning, and Artificial intelligence**.

REFERENCES

1. **UCI Machine Learning Repository** –
   Heart Disease Dataset, Diabetes Dataset (PIMA), Chronic Kidney Disease Dataset
2. **Kaggle Datasets** –
   IoT Device Energy Consumption Dataset, Fashion MNIST Dataset, Mobile App Review Dataset
3. **Indian Liver Patient Dataset (ILPD)** –
   Liver Disease Dataset for medical analysis and synthetic data generation
4. **Python Documentation (python.org)** –
   Libraries used: Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn, TensorFlow/Keras
5. **Scikit-learn Documentation** –
   K-Means Clustering, GridSearchCV, Preprocessing Techniques
6. **TensorFlow / Keras Documentation** –
   Artificial Neural Networks (ANN), CNN, Embedding Layers
7. **Research Papers & Articles** –
   • "CTGAN: Generating Synthetic Tabular Data using GANs"
   • "Applications of Machine Learning in Healthcare"
   • "Natural Language Processing for Sentiment Analysis"
8. **W3Schools / Online Resources** –
   Python Programming, Data Preprocessing, and Machine Learning Concepts
9. **Google Colab / Jupyter Notebook** –
   Development environment used for implementation and experimentation .

10. **CTGAN: Modeling Tabular Data using Conditional GAN** –
    Introduces CTGAN architecture for generating realistic tabular data and handling mixed data types.
11. **Deep Learning** by Ian Goodfellow, Yoshua Bengio, Aaron Courville –
    Comprehensive resource covering ANN, deep learning, and generative models like GANs.
12. **A Neural Probabilistic Language Model** by Yoshua Bengio –
    A foundational paper explaining neural networks for language modeling and embeddings.
13. **Link for my web app**

    Streamlit