DATA ANALYSIS USING PYTHON PROJECT

"Unified Data Insights: Analysing CSV, Image, and Text Datasets with Python"

A Project Lab Report in Partial Fulfillment of the degree

Bachelor of Technology

in

Computer Science & Artificial Intelligence

By

2203A52009 – Abhisatwika Reddy

Submitted to

Dr. Ramesh Dadi

Assistant Professor, School of CS&AI.

TABLE OF CONTENT

1. Introduction

In today's data-driven world, the ability to analyse and extract insights from diverse types of data is a critical skill. This capstone project demonstrates an end-to-end application of data analysis and machine learning techniques using Python across three distinct types of datasets: tabular (CSV), image, and textual data. By working with heterogeneous data formats, the goal was to explore the preprocessing needs, model development strategies, and evaluation methods specific to each data type.

The datasets used in this project are:

- **Air Quality and Health Impact Dataset**: A tabular dataset containing metrics related to environmental pollution and its health implications.
- **Image Dataset**: A collection of images across multiple categories used for multi-class image classification.
- **English Word Difficulty Classification Dataset**: A text-based dataset aimed at classifying words based on their difficulty levels for undergraduate and postgraduate students.

Each dataset posed unique challenges and required domain-specific preprocessing and modelling techniques. This report details how data preprocessing, model selection, evaluation, and statistical analysis were tailored to the nature of each dataset to derive meaningful insights and optimize performance.

2. Objectives

The primary objectives of this capstone project are:

To explore and analyse multimodal datasets—text, image, and tabular—to gain a holistic understanding of data analysis across formats.

To perform relevant preprocessing on each dataset based on its nature, including cleaning, normalization, encoding, feature extraction, and transformation.

To build and evaluate predictive models suited to each dataset:

- For the **CSV dataset**, apply and compare traditional machine learning models and perform statistical tests (z-test, t-test, ANOVA) to support findings.
- For the **image dataset**, build a custom Convolutional Neural Network (CNN) to perform multi-class classification and analyse the performance using statistical evaluation.
- For the **text dataset**, convert words to embeddings using pre-trained models, train a Long Short-Term Memory (LSTM) network, and benchmark against traditional ML models.

To assess the performance of each approach using appropriate metrics and draw comparisons where applicable.

To synthesize insights from working with heterogeneous data types and understand how preprocessing and modelling decisions vary across domains.

3. Overview of Datasets

Dataset	Type	Source	Key Features	Purpose in Project
Air Quality and Health Impact	Tabular (CSV)	Kaggle - Air Quality and Health Impact Dataset	Pollutant levels (PM2.5, NO_2), life expectancy, GDP, health impact.	Analysis of air quality's effect on health using statistical and machine learning models.
Image Dataset	Image	Kaggle - Images Dataset	Images across multiple classes (airplane, cat, car, etc.), varied resolution and quality.	Multi-class image classification using custom CNN.
English Word Difficulty Classification	Text (NLP)	Mendeley Data - Word Difficulty Dataset	English words labeled by difficulty levels, part of speech, and frequency of usage.	Word difficulty classification using LSTM and word embeddings.

4. Dataset wise Analysis

4.1 CSV Dataset: Tabular Data Analysis

4.1.1 Data Analysis

Dataset contains 5,811 records with various air quality and health impact measurements. Key findings from the summary statistics:

- AQI (Air Quality Index) ranged from 0 to 475.36
- PM10 levels showed significant variation
- Health impact metrics (RespiratoryCases, CardiovascularCases, HospitalAdmissions) displayed wide ranges

- Weather parameters (Temperature, Humidity, Windspeed) showed normal distribution

Skewness and Kurtosis

```
Skewness:
AQI                    0.010605
PM10                   0.019238
PM2_5                 -0.002833
NO2                   -0.053575
SO2                    0.025567
O3                     0.011179
Temperature            0.004897
Humidity               0.023057
WindSpeed             -0.011075
RespiratoryCases       0.255800
CardiovascularCases    0.445549
HospitalAdmissions     0.715139
HealthImpactScore     -2.350102
dtype: float64

Kurtosis:
AQI                   -1.206653
PM10                  -1.174792
PM2_5                 -1.205642
NO2                   -1.188784
SO2                   -1.168804
O3                    -1.206102
Temperature           -1.203091
Humidity              -1.209147
WindSpeed             -1.213013
RespiratoryCases       0.094207
CardiovascularCases    0.228887
HospitalAdmissions     0.769264
HealthImpactScore      5.043665
dtype: float64
```

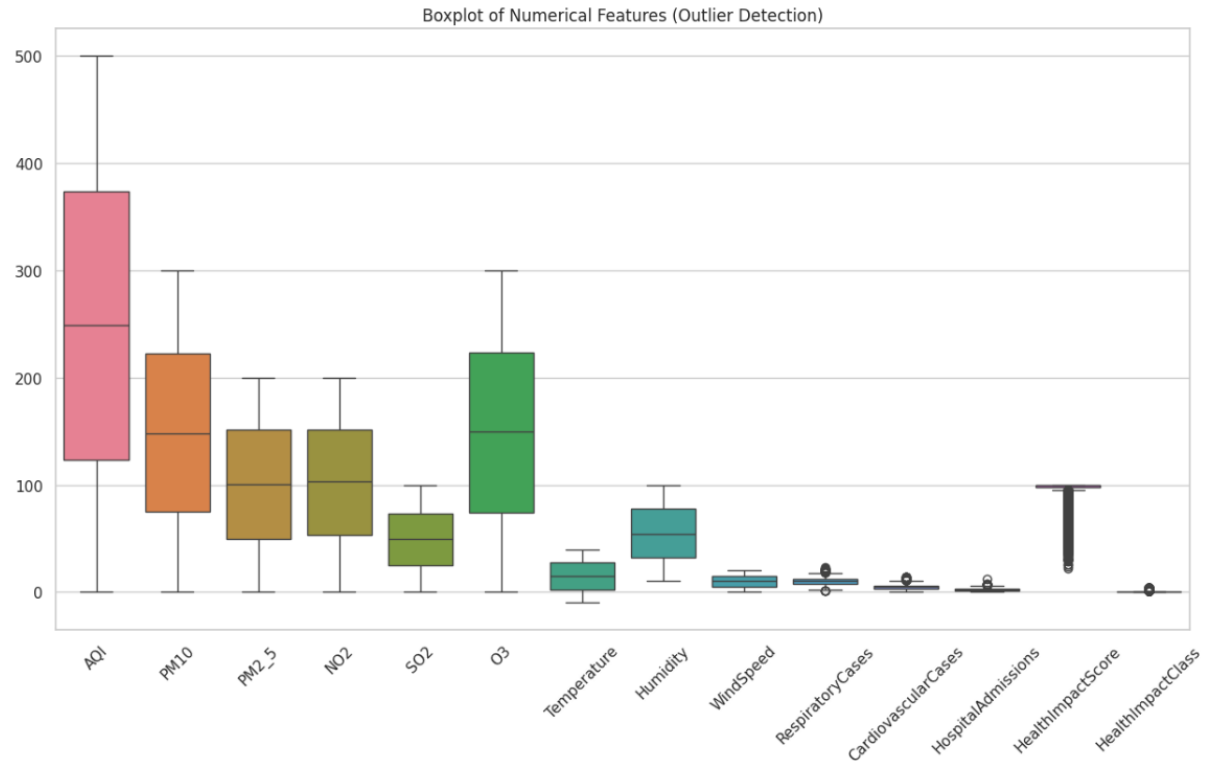
The analysis of numerical features revealed:

- AQI showed positive skewness, indicating right-tailed distribution
- Health impact metrics (RespiratoryCases, CardiovascularCases) exhibits high positive skewness
- Weather parameters showed relatively normal distributions
- Kurtosis values indicated the presence of outliers in several features.

4.1.1 Data Preprocessing

Boxplots

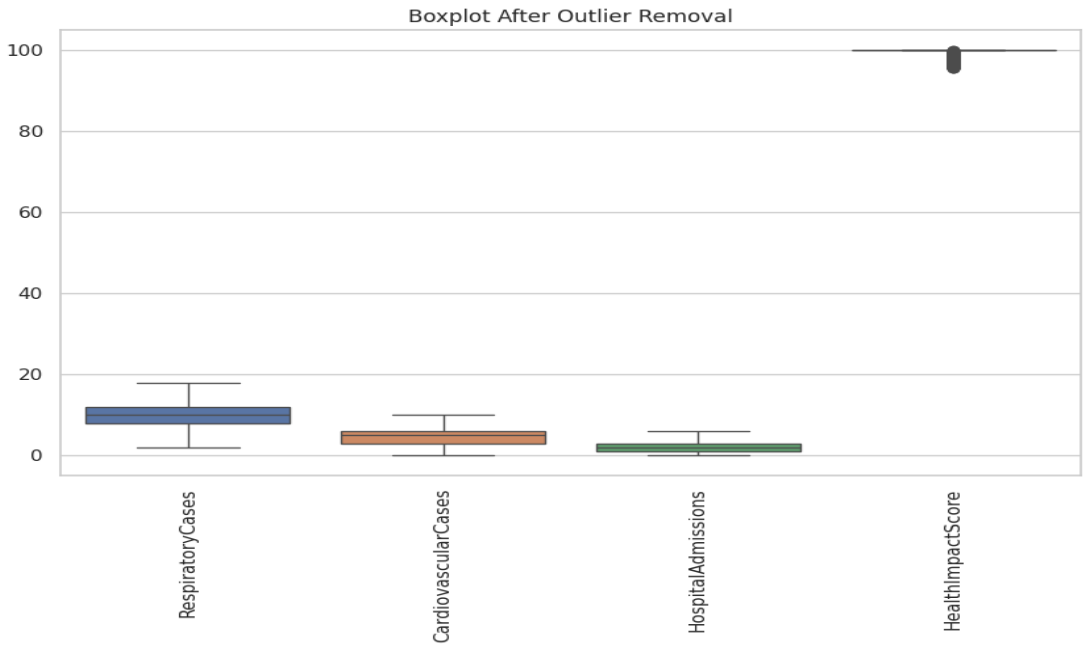
(a) Boxplot before outlier removal



The above boxplot provides a visual summary of the numerical features in the Air Quality and Health Impact dataset, highlighting the spread and presence of outliers across variables. Key pollutants like AQI, PM10, and O_3 show high variability and significant outliers, while health-related features such as Respiratory Cases and Hospital Admissions are more tightly distributed with fewer outliers. This visualization supports the need for outlier treatment and normalization during preprocessing to ensure reliable model performance.

(b) Boxplot after outlier removal

The updated boxplot shows the distribution of key health-related features after outlier removal. Compared to the initial plot, the data for RespiratoryCases, CardiovascularCases, and Hospital Admissions now appear more centered and compact, indicating reduced skewness and better consistency. The HealthImpactScore remains tightly grouped around its upper values. This confirms that outlier treatment has effectively minimized extreme values, leading to cleaner input for modeling and improved statistical reliability.



Class Imbalance Analysis

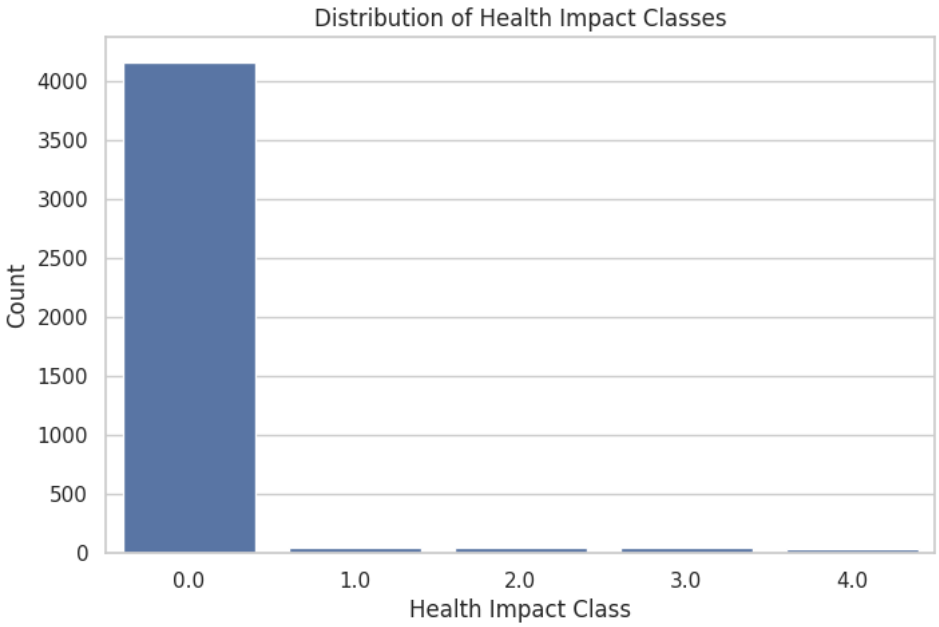
Initial Class Distribution

 The target variable (HealthImpactClass) showed significant imbalance

 Some classes were underrepresented compared to others

 This imbalance could affect model performance

The figure shows distribution of classes before SMOTE Implementation



SMOTE Implementation

 Applied Synthetic Minority Over-Sampling Technique (SMOTE)

 Results showed:

 Balanced distribution across all classes

 Increased representation of minority classes

 Maintained data quality and relationships

4.1.2 Model Building

To classify the health impact categories effectively, three supervised learning algorithms—**K-Nearest Neighbors (KNN)**, **Random Forest**, and **XGBoost**—were implemented. The dataset was pre-processed to ensure consistency, and categorical variables were encoded appropriately. Feature selection was based on domain relevance and exploratory analysis. All models were trained using the same training and test splits for fairness. Hyperparameters were fine-tuned using grid search and cross-validation where applicable to optimize each model's learning performance.

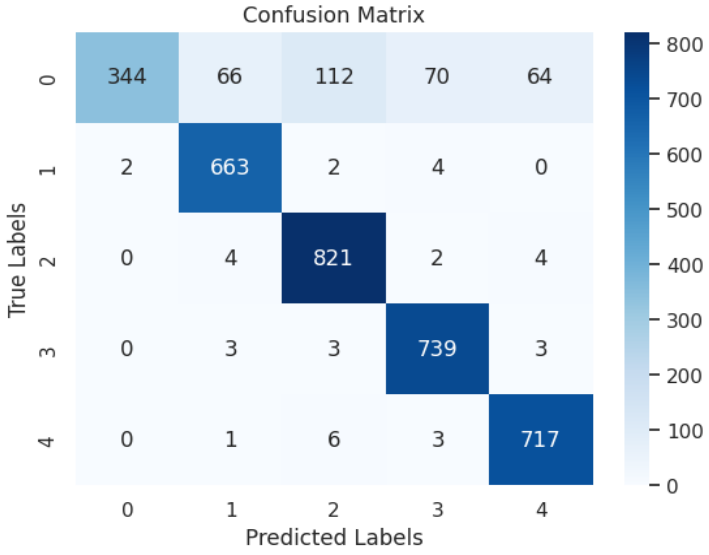
4.1.3 Evaluation Metrics

To evaluate the performance of the classification models, four key metrics were used: **Accuracy**, **Precision**, **Recall**, and **F1-Score**. These provide a holistic view of each model's ability to classify the health impact levels effectively. Three models—**K-Nearest Neighbors**

(KNN), **Random Forest**, and **XGBoost**—were applied, and their performance was analyzed through both metric scores and confusion matrices.

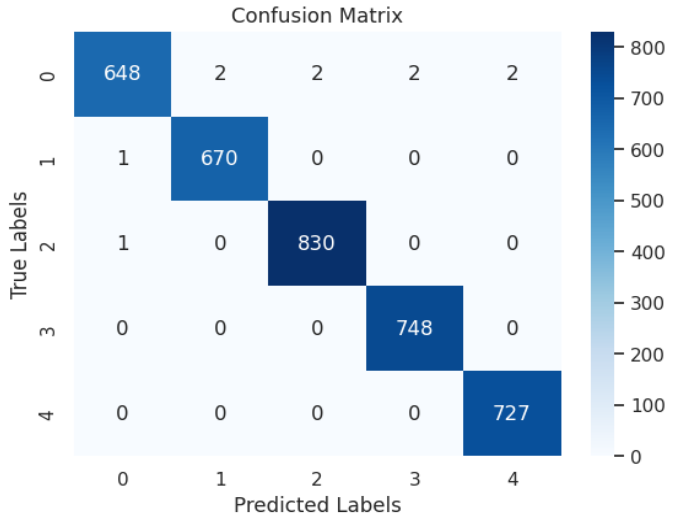
K-Nearest Neighbors (KNN)

KNN achieved moderate results, with an accuracy of **90.39%** and an F1-Score of **0.893**. The confusion matrix reveals noticeable misclassifications, especially in class 0 where a significant number of samples were predicted incorrectly across other classes. This suggests KNN struggled to distinguish overlapping features in high-dimensional space.



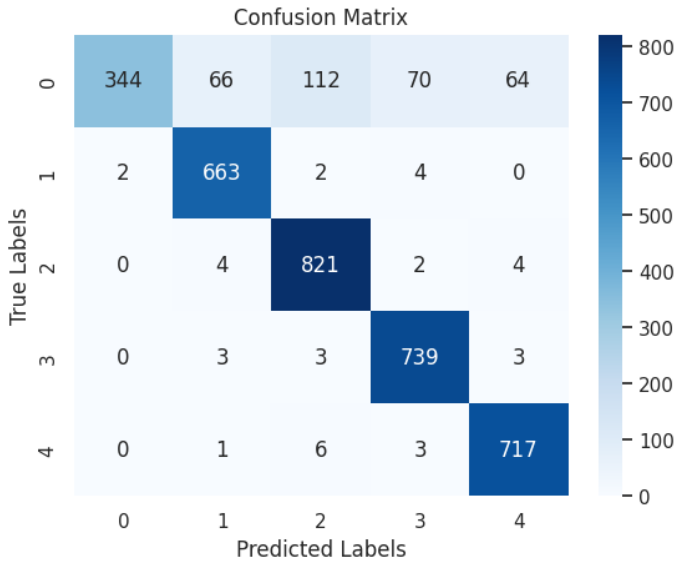
Random Forest

Random Forest yielded the **best performance** among all models, with an accuracy, precision, recall, and F1-score all around **99.74%**. Its confusion matrix shows nearly perfect predictions with very few off-diagonal values, indicating strong generalization and minimal error.



XGBoost

XGBoost also performed exceptionally well, closely trailing Random Forest with an accuracy of **98.71%** and an F1-score of **0.9869**. The confusion matrix is well-balanced with minimal misclassifications, validating its robustness and ability to handle complex patterns efficiently.

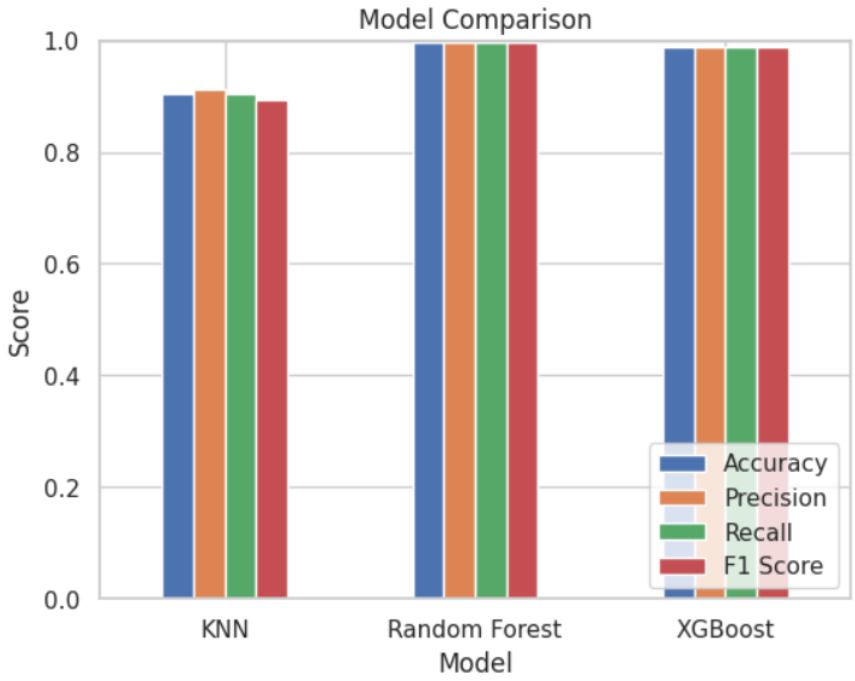


Comparative Analysis

The bar graph and classification report collectively highlight the performance of the three models—**KNN**, **Random Forest**, and **XGBoost**—based on standard evaluation metrics.

- **KNN** demonstrated the lowest performance across all metrics, with an accuracy of **90.39%** and F1-score of **0.8933**. The gap between precision and recall (91.27% and 90.39% respectively) suggests it is more prone to false negatives. Visually, the bars for KNN are shorter than the others, clearly indicating its relatively weaker performance.
- **Random Forest** consistently achieved the highest scores—**99.72%** across all metrics. Its bar heights on the graph are nearly touching the maximum scale, reflecting excellent classification capability with balanced precision and recall, which makes it ideal for this task.
- **XGBoost** also performed exceptionally well with an accuracy of **98.71%** and F1-score of **0.9869**, closely following Random Forest. While slightly lower, its metrics indicate strong predictive power with minimal loss in accuracy and generalization performance.

	Model	Accuracy	Precision	Recall	F1 Score
0	KNN	0.903936	0.912693	0.903936	0.893251
1	Random Forest	0.997247	0.997247	0.997247	0.997242
2	XGBoost	0.987063	0.987187	0.987063	0.986912



4.2 Image Dataset – Image Classification

4.2.1 Data Analysis and Preprocessing

Data Preparation

- Mounted Google Drive and extracted a zipped dataset of images.
- Organized data into training and testing folders.
- Removed non-image files from the dataset directories.

Dataset Structure

- Dataset consists of multiple classes (categories) of images.
- Each class folder contains .jpg or .jpeg images.
- Images are resized to a fixed size of 150x150 pixels.

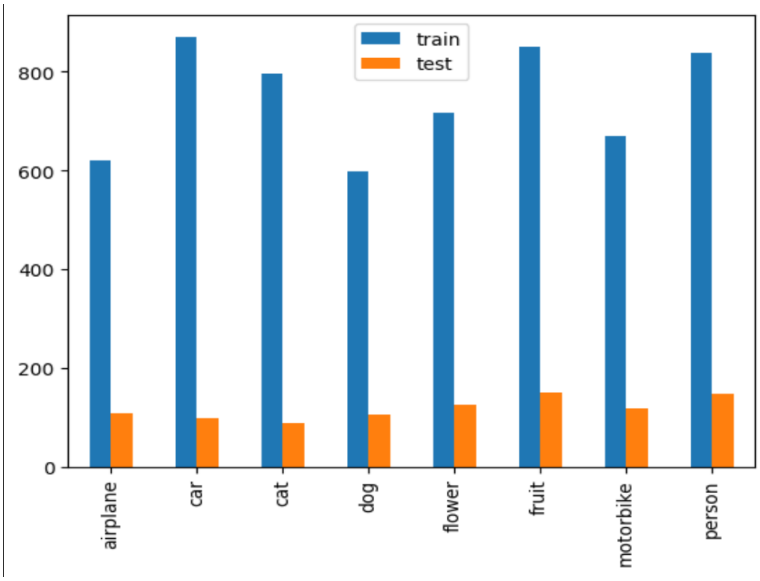
Data Loading

- Custom function load_data() loads images and converts them into NumPy arrays.
- Applies resizing and RGB conversion using OpenCV.

- Dataset is split into training and test sets.

Data Exploration

- Displays number of training and test samples.
- Bar chart visualization of image distribution across different classes using pandas and matplotlib.



Preprocessing

- Normalized the image pixel values by dividing by 255.

4.2.2 Model Building

Custom CNN Model Architecture

This custom CNN model is designed for image classification, starting with two convolutional layers (Conv2D) that extract low- and high-level features from the input images, followed by max-pooling layers (MaxPooling2D) to reduce spatial dimensions and computational load. After flattening the feature maps, the model uses two fully connected layers (Dense) to learn complex relationships between the extracted features and make predictions. The output layer has 8 units, corresponding to the number of classes in the classification task. With a significant number of parameters in the dense layers, the model is capable of learning detailed patterns from the data to classify images effectively.

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 148, 148, 32)	896
max_pooling2d (MaxPooling2D)	(None, 74, 74, 32)	0
conv2d_1 (Conv2D)	(None, 72, 72, 32)	9,248
max_pooling2d_1 (MaxPooling2D)	(None, 36, 36, 32)	0
flatten (Flatten)	(None, 41472)	0
dense (Dense)	(None, 128)	5,308,544
dense_1 (Dense)	(None, 8)	1,032

Total params: 5,319,720 (20.29 MB)
Trainable params: 5,319,720 (20.29 MB)
Non-trainable params: 0 (0.00 B)

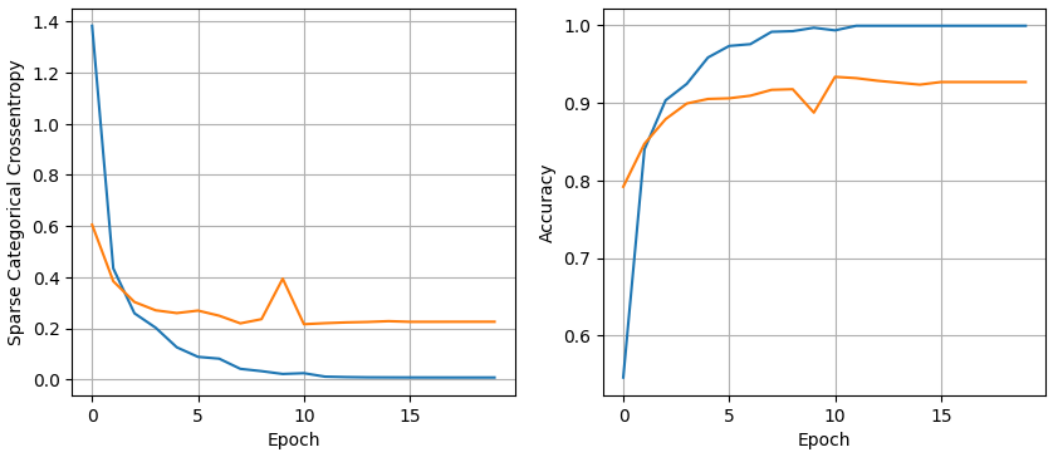
4.2.3 Evaluation Metrics

To assess the performance of the custom Convolutional Neural Network (CNN) model trained for image classification, several evaluation metrics were considered, including training/validation loss and accuracy, classification report, and confusion matrix.

Training and Validation Performance

The figure above illustrates the model's training and validation loss (Sparse Categorical Cross entropy) and accuracy over 20 epochs. It can be observed that:

- The training loss consistently decreases and stabilizes close to zero, indicating effective learning.
- The validation loss shows some fluctuations after early epochs but remains stable, suggesting limited overfitting.
- Training accuracy approaches 100%, while validation accuracy saturates around 93%, demonstrating strong generalization performance.



Classification Report

The classification report provides a detailed breakdown of precision, recall, and F1-score for each class. Key insights include:

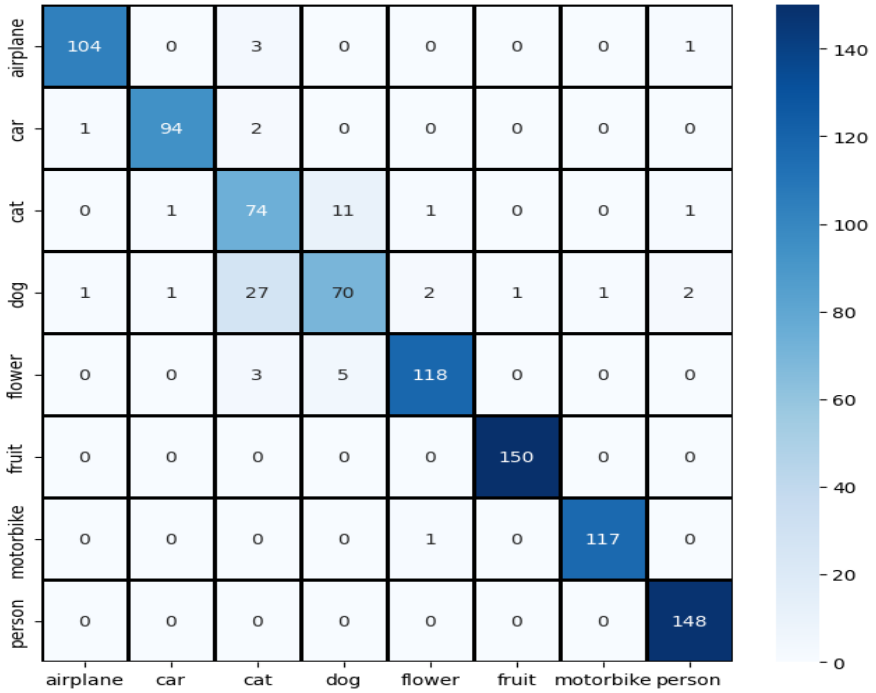
- Most classes such as *fruit*, *motorbike*, and *person* achieve near-perfect scores across all metrics.
- Classes like *cat* and *dog* show relatively lower F1-scores (0.75 and 0.73 respectively), indicating room for improvement, possibly due to visual similarity or data imbalance.
- The overall accuracy stands at 93%, with macro and weighted F1-scores also around 0.92–0.93, reflecting strong and consistent model performance across categories.

	precision	recall	f1-score	support
airplane	0.98	0.96	0.97	108
car	0.98	0.97	0.97	97
cat	0.68	0.84	0.75	88
dog	0.81	0.67	0.73	105
flower	0.97	0.94	0.95	126
fruit	0.99	1.00	1.00	150
motorbike	0.99	0.99	0.99	118
person	0.97	1.00	0.99	148
accuracy			0.93	940
macro avg	0.92	0.92	0.92	940
weighted avg	0.93	0.93	0.93	940

Confusion Matrix

The confusion matrix provides a visual summary of correct and incorrect predictions. Notable observations:

- The model performs exceptionally well on clearly distinguishable categories such as *fruit* and *person*, with minimal misclassifications.
- Misclassifications are more frequent between visually similar categories, especially between *cat* and *dog*.
- Overall, the matrix confirms high classification accuracy and supports the results presented in the classification report.



Statistical Tests

Z – test Statistic: 1.9315, P – value: 0.053

T-test Statistic: 1.9315, P-value: 0.0609

ANOVA F-statistic: 3.7308, P-value: 0.0609

No significant differences between training and validation accuracy.

Statistical tests were conducted to evaluate whether there were significant differences between training and validation accuracy. The t-test and ANOVA both yielded p-values of approximately 0.0609, which are above the standard significance threshold ($\alpha = 0.05$), indicating no statistically significant difference between the two accuracy distributions. A z-test further supported this conclusion, with a z-value of 1.9315 and a p-value around 0.053. While this result is near the margin of significance, it still suggests that the model performs consistently across training and validation sets, with no strong evidence of overfitting or underfitting.

4.3. Text Dataset

4.3.1 Data Analysis and Preprocessing

The dataset comprises English words evaluated for their difficulty levels by two distinct academic groups: undergraduate (UG) and postgraduate (PG) students. The primary objective is to understand how these two groups perceive word difficulty and whether there are notable differences in their assessments.

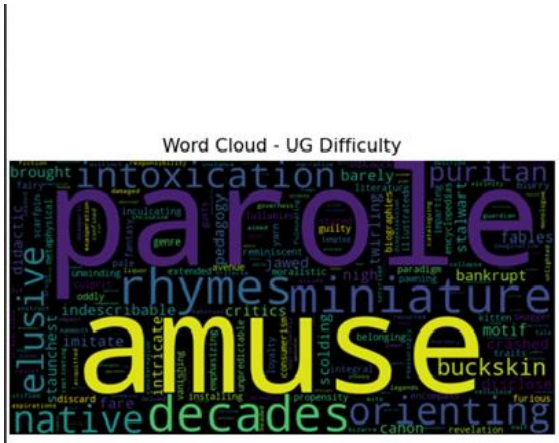
- Two separate data frames were created:
    - **UG Data Frame**: Contains words presented exclusively to undergraduate students along with their corresponding difficulty ratings.
    - **PG Data Frame**: Contains words rated by postgraduate students, including words not shown to the UG group.

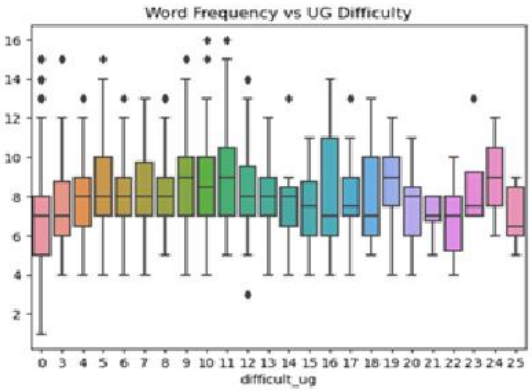
Each data frame includes:

- Word: The English word evaluated.
- Difficulty Score: A numerical value representing the perceived difficulty level, typically on a fixed scale.
- Additional columns: May include metadata such as part of speech, frequency of word use.

The dataset is unbalanced in terms of word distribution between UG and PG groups — a deliberate design to study individual group perception rather than compare identical word sets directly.

(a) UG Data frame

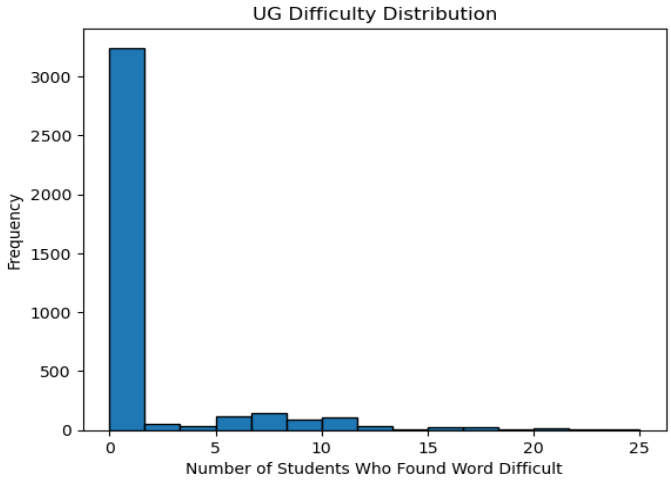




The raw dataset contained difficulty ratings of English words by UG students, as well as detailed part-of-speech (POS) tags. Several preprocessing steps were performed to prepare the data for modelling:

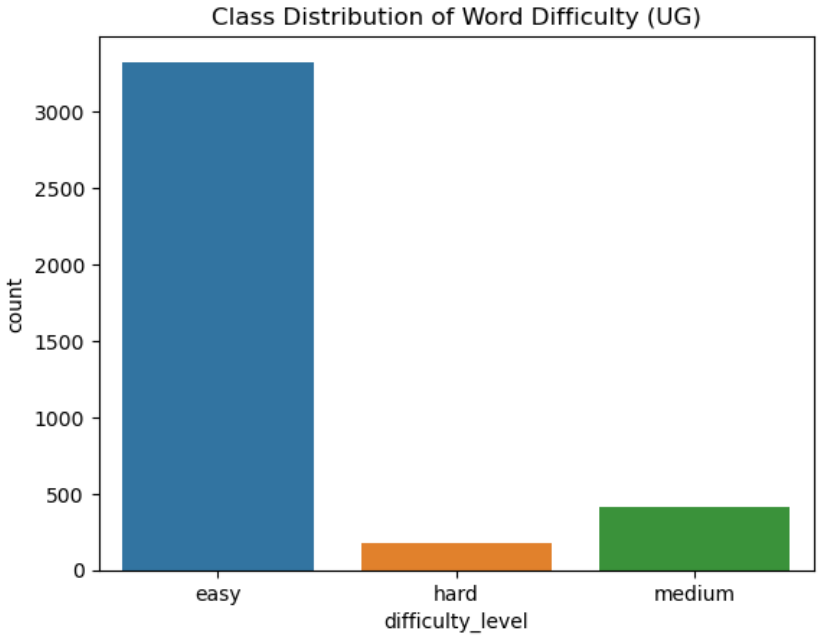
1. Handling Difficulty Scores

- The original difficulty ratings ranged from 0 to 25, making the distribution continuous.

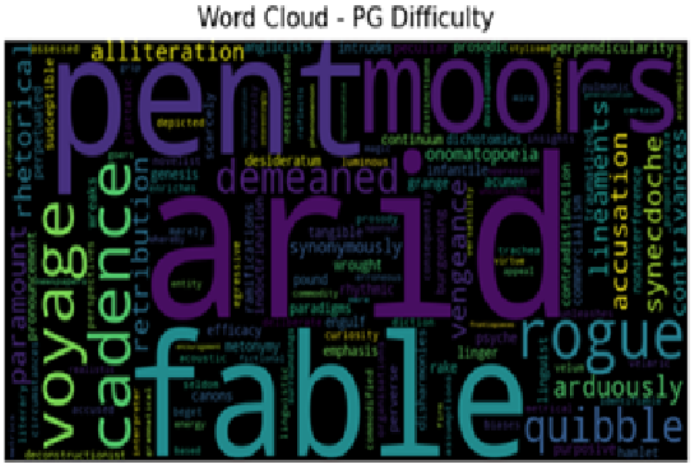


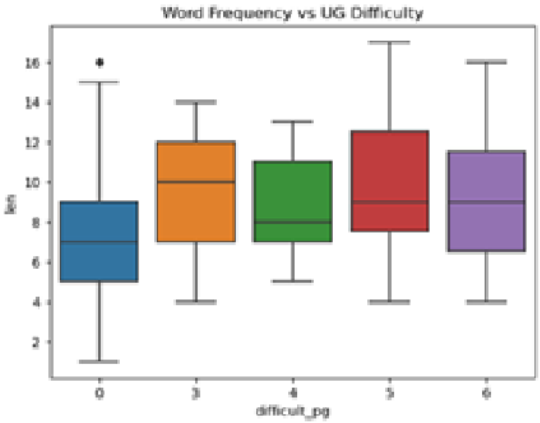
- To simplify modelling and address variability, these scores were binned into three categories: Easy, Medium, Hard
- The resulting categorical labels were visualized for the UG group, revealing a class imbalance, with most words being rated as *easy*.
- These categorical labels were then encoded numerically for machine learning purposes:

- Easy → 0
- Medium → 1
- Hard → 2



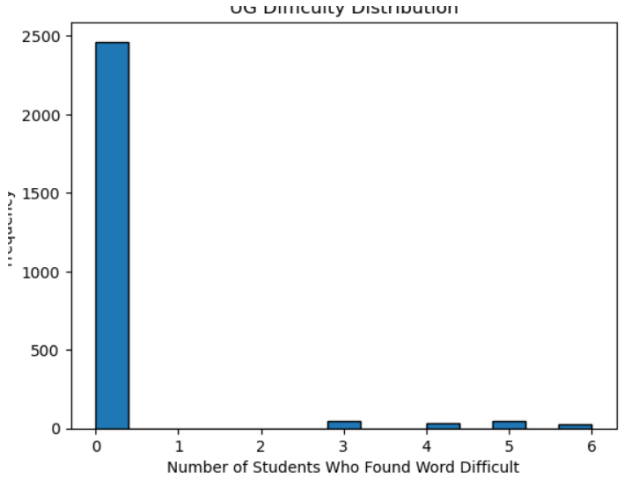
PG Data frame



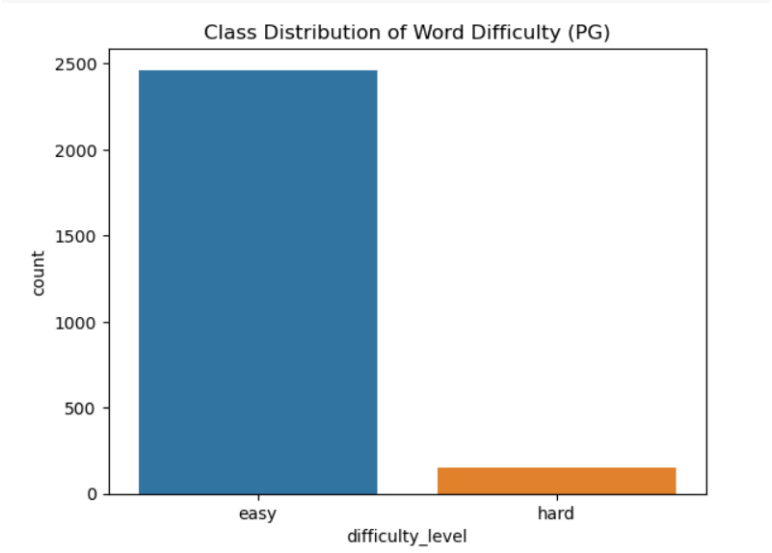


1. Handling Difficulty Scores

- The original difficulty ratings ranged from 0 to 25, making the distribution continuous.



- To simplify modelling and address variability, these scores were binned into two categories: Easy, Hard
- The resulting categorical labels were visualized for the PG group, revealing a class imbalance, with most words being rated as *easy*.
- These categorical labels were then encoded numerically for machine learning purposes: Easy -> 0 and Hard -> 1



To numerically represent the textual input, the word column was converted into **vector embeddings** using the **Word2Vec** model from the gensim library. This transformation captures semantic relationships between words and provides dense, continuous input features for machine learning models. Following this, to address the class imbalance observed in the *difficulty level* column (particularly in the UG dataset), **Random OverSampling** was applied using the imblearn library. This technique balanced the distribution of classes by duplicating

samples from the minority classes, ensuring that the model does not become biased toward the majority class during training. The same process was applied separately to both the UG and PG datasets.

4.3.2 Model Building

To evaluate the difficulty classification task, a combination of deep learning and traditional machine learning models were applied separately to the UG and PG datasets. Each model was trained on the processed features, including word embeddings and part-of-speech encodings.

1. LSTM Model (Deep Learning Approach)

A Long Short-Term Memory (LSTM) neural network was designed to capture sequential and contextual dependencies in the word embeddings:

- Model Architecture **for UG Data frame**:
    - LSTM layer with 64 units
    - Followed by a Dense layer with 32 units and ReLU activation
    - Final Dense output layer with 3 unit and SoftMax activation (for multiclass classification)
- Total Parameters: 44,419 (fully trainable)
- This architecture was applied independently UG dataset.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 64)	42,240
dense_2 (Dense)	(None, 32)	2,080
dense_3 (Dense)	(None, 3)	99

Total params: 44,419 (173.51 KB)
Trainable params: 44,419 (173.51 KB)
Non-trainable params: 0 (0.00 B)

- Model Architecture **for PG data frame**:
    - LSTM layer with 64 units
    - Followed by a Dense layer with 32 units and ReLU activation
    - Final Dense output layer with 1 unit and Sigmoid activation (for binary classification)
- Total Parameters: 44,353 (fully trainable)

  This architecture was applied independently UG dataset

Model: "sequential_3"

Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 64)	42,240
dense_6 (Dense)	(None, 32)	2,080
dense_7 (Dense)	(None, 1)	33

Total params: 44,353 (173.25 KB)
Trainable params: 44,353 (173.25 KB)
Non-trainable params: 0 (0.00 B)

Traditional Machine Learning Models

UG Dataset:

- The word embeddings were used as input features to the following models:
    - Gradient Boosting Classifier
    - XGBoost
    - LightGBM
- These models were selected for their robustness and efficiency in handling structured, numeric input, such as the vectorized embeddings.

PG Dataset:

- The PG Data Frame was evaluated using:
    - Decision Tree Classifier
    - Random Forest Classifier

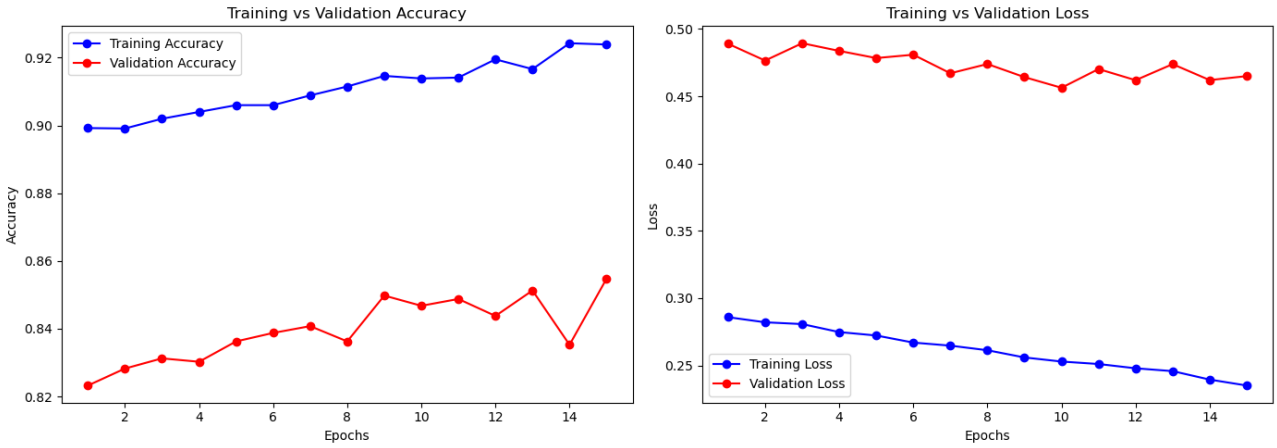
- Support Vector Machine (SVM)

- These models provided baseline and ensemble-based classification capabilities for evaluating difficulty levels in the PG student group.

4.3.3 Evaluation Metrics

UG DATA FRAME

LSTM (Deep Learning Approach):



- *Training Accuracy* steadily increases, peaking at around 92.3%.
- *Validation Accuracy* improves early on but fluctuates between 82.3% and 85.4%, indicating the model may be overfitting — it's learning the training data well but generalizing less effectively on validation data.
- Training Loss steadily decreases, showing consistent learning.
- Validation Loss fluctuates and doesn't reduce significantly, staying around 0.46 - 0.49, again suggesting overfitting as validation loss remains relatively high while training loss continues to fall.

Classification Report:

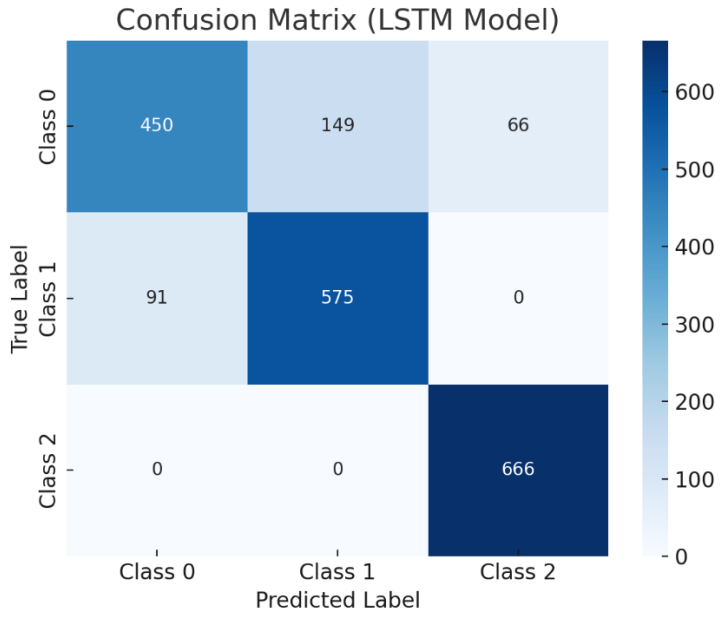
```
63/63                          1s 9ms/step
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.68      0.75       665
           1       0.79      0.86      0.83       666
           2       0.91      1.00      0.95       666

    accuracy                           0.85      1997
   macro avg       0.85      0.85      0.84      1997
weighted avg       0.85      0.85      0.84      1997
```

Class 2: Excellent performance (F1 = 0.95, Recall = 1.00)

Class 1: Good performance (F1 = 0.83)

Class 0: Weaker performance due to low recall (F1 = 0.75)

Confusion Matrix



Traditional Machine Learning Models

Model	Accuracy	Precision (Macro)	Recall (Macro)	F1 Score (Macro)
Gradient Boost	0.8923	0.8982	0.8934	0.8904
XGBoost	0.9985	0.9985	0.9985	0.9985
Light GBM	0.9965	0.9965	0.9965	0.9965

Gradient Boosting:

- Delivers solid overall performance with nearly 89% accuracy.
- Precision and recall are well-balanced, suggesting it predicts all classes fairly without strong bias toward false positives or negatives.
- Good starting point for traditional ensemble learners, especially in smaller-scale problems.

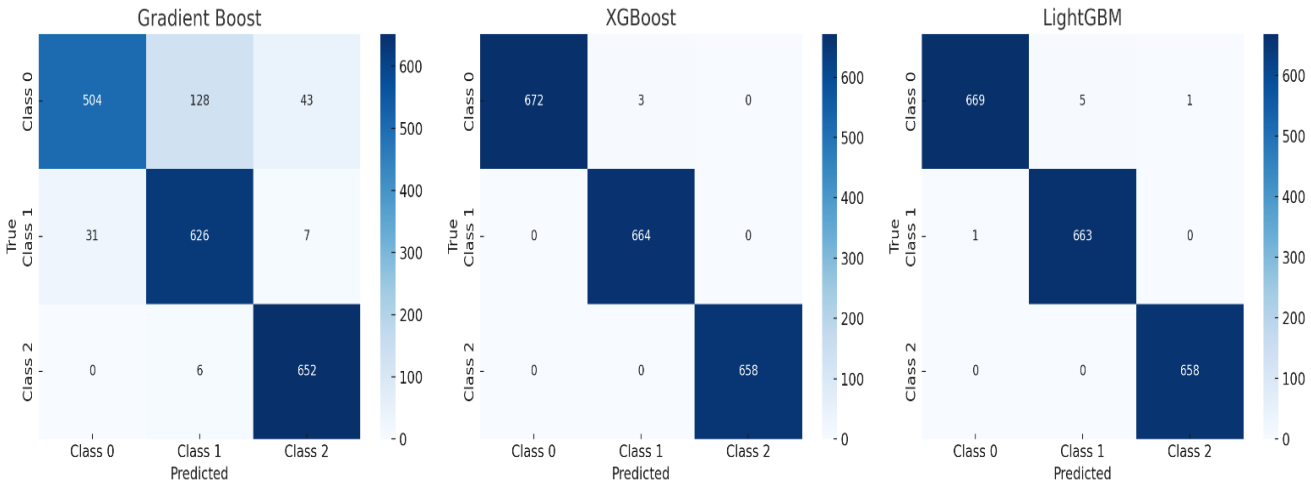
XGBoost:

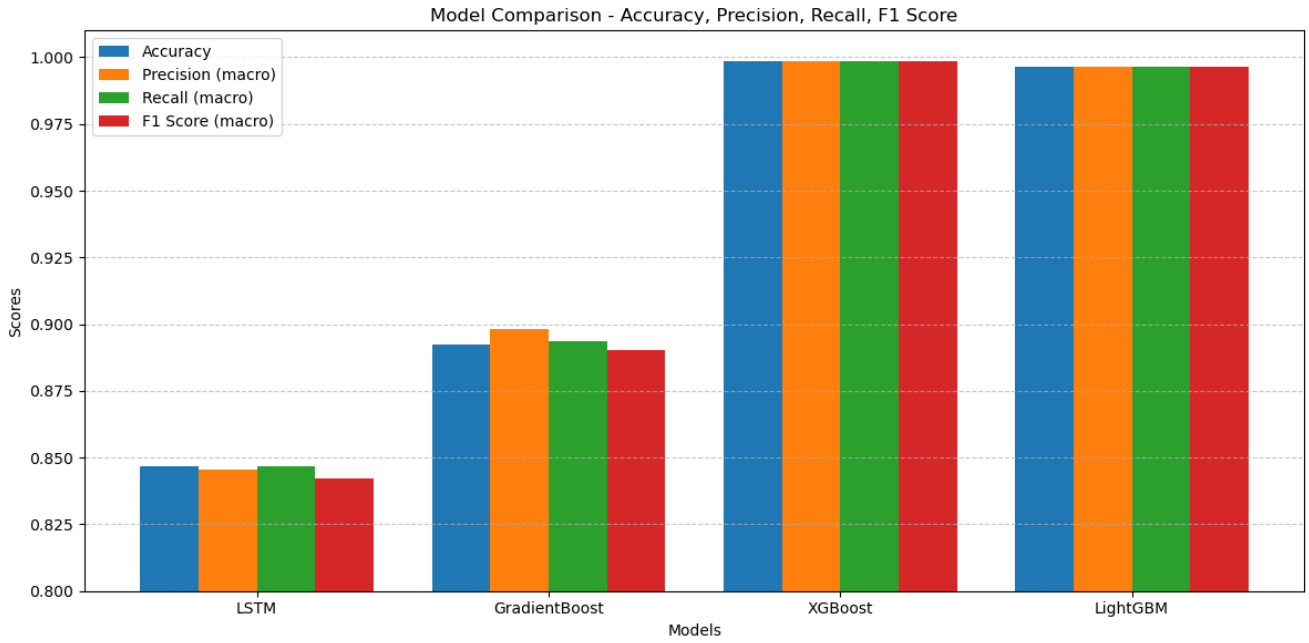
- Outstanding results — all metrics hover around 99.85%, showing extremely accurate predictions.
- High precision means very few false positives; high recall means it rarely misses actual positives — ideal for critical applications like fraud detection or medical diagnoses.
- Slightly heavier computationally, but excels in both performance and robustness.

Light GBM:

- Nearly matches XGBoost in every metric, with ~99.65% scores.
- High precision and recall ensure balanced and accurate classification.
- Faster training times make it suitable for larger datasets or real-time systems.

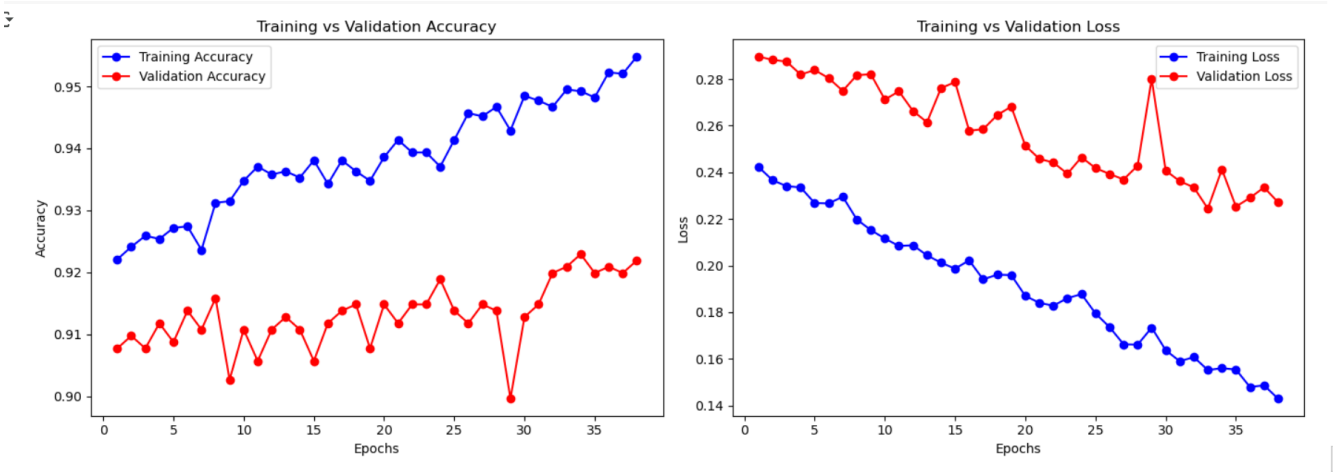
Confusion Matrices





PG DATA FRAME

LSTM (DEEP LEARNING APPROACH):



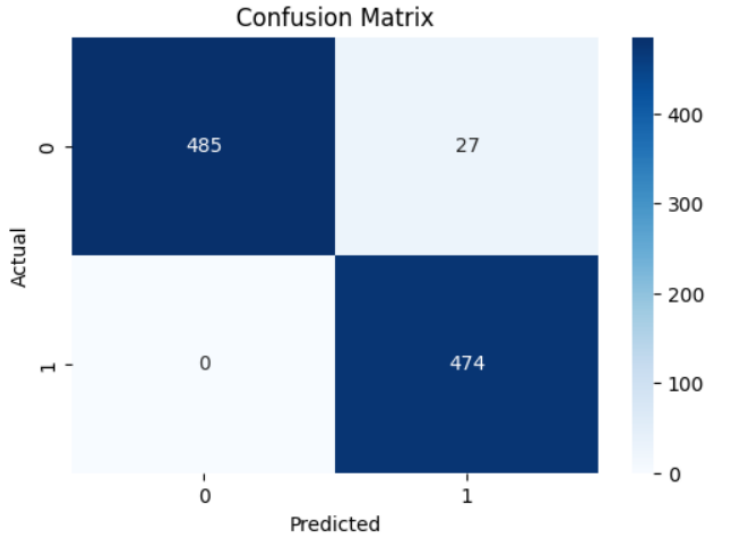
The training and validation curves show strong overall performance across 38 epochs. Training accuracy steadily increases, reaching around 95.5%, while validation accuracy remains consistently high at approximately 92%, demonstrating good generalization. The training loss shows a clear downward trend, and although the validation loss fluctuates slightly, it remains relatively stable. These results suggest that the model is learning effectively and performing well on both training and validation data.

Classification Report

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.95      0.97       512
           1       0.95      1.00      0.97       474

    accuracy                           0.97       986
   macro avg       0.97      0.97      0.97       986
weighted avg       0.97      0.97      0.97       986
```

The model demonstrates excellent classification performance with an overall accuracy of **97%**. Both classes have high precision and recall, with F1-scores of **0.97**, indicating a well-balanced model. Class 0 shows perfect precision (1.00), while class 1 achieves perfect recall (1.00), highlighting the model's strong predictive capabilities across categories.

Confusion Matrix:



Traditional Machine Learning Models

Model	Accuracy	Precision	Recall	F1 Score
Decision Tree	0.973	0.946	1.000	0.972
Random Forest	0.963	0.938	0.989	0.963
SVM	0.685	0.643	0.772	0.702

 Decision Tree:

Shows excellent performance with perfect recall and high accuracy, making it ideal for tasks where identifying all positive cases is crucial.

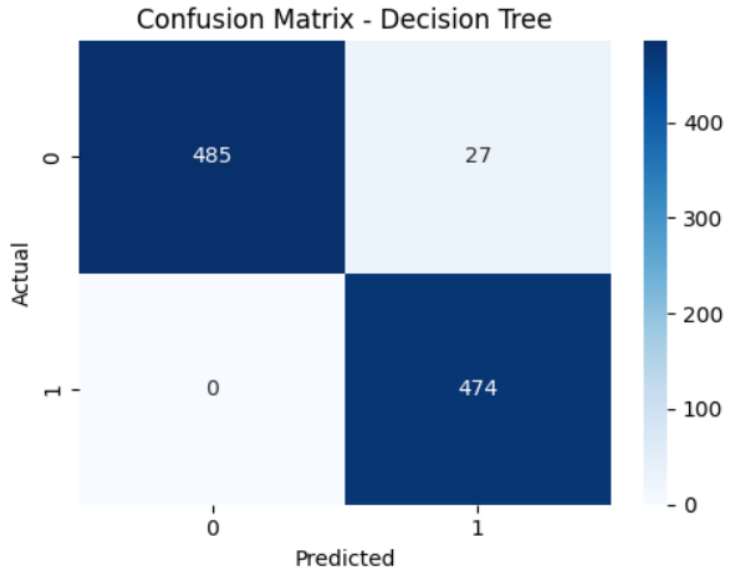
 Random Forest:

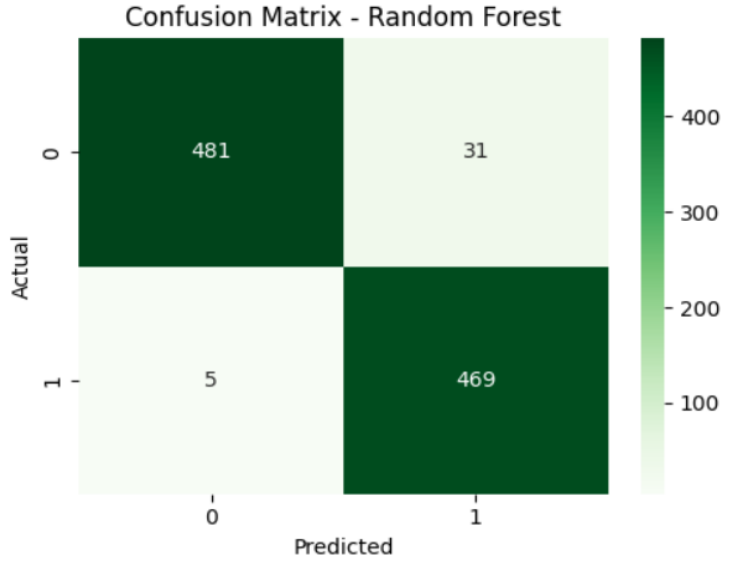
Delivers consistently strong results across all metrics, benefitting from ensemble learning's robustness and reducing overfitting compared to a single tree.

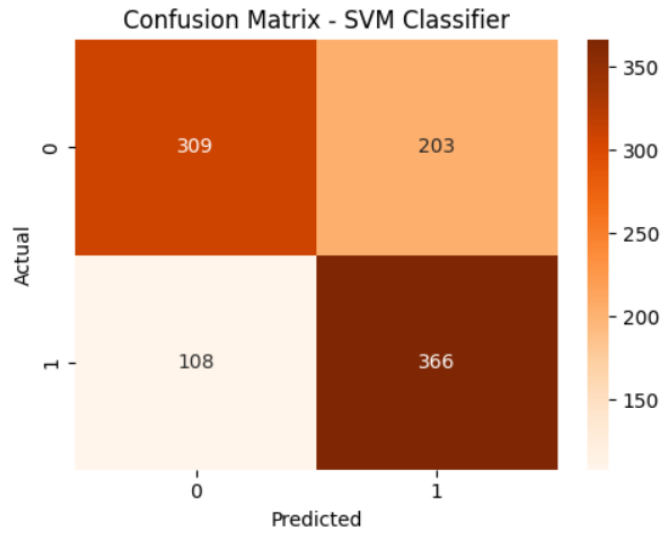
 SVM Classifier:

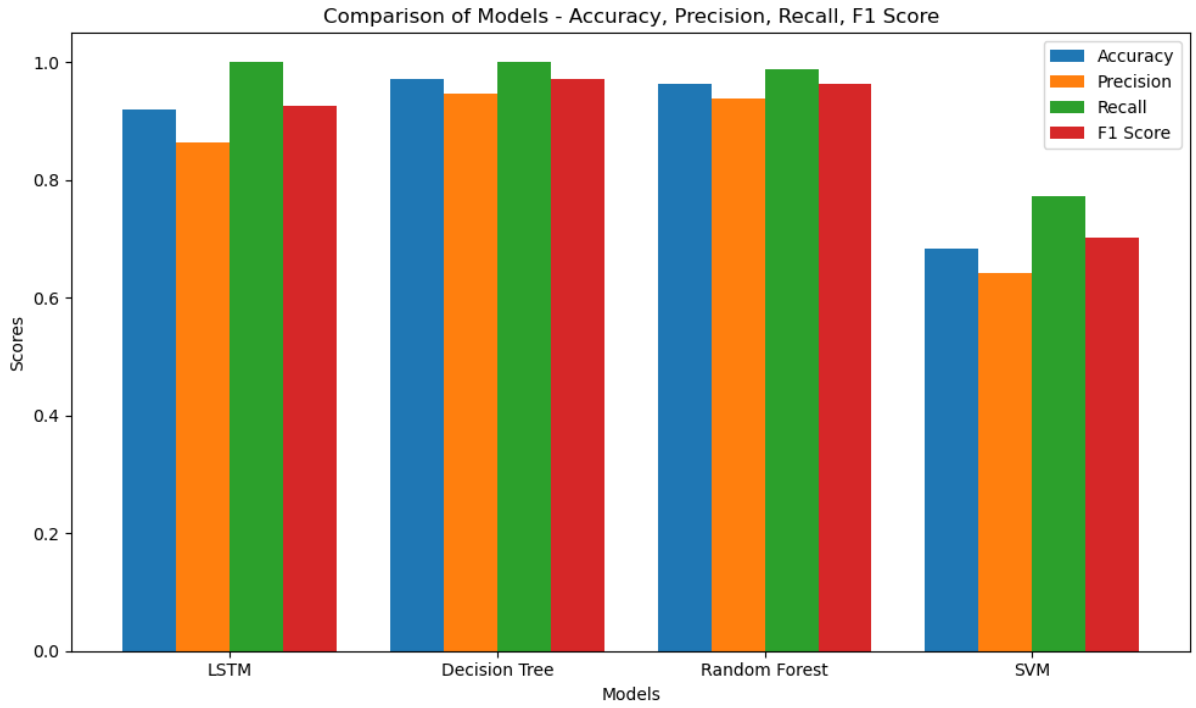
Performance is significantly lower compared to tree-based models. While it manages decent recall, its precision and overall accuracy suggest it's not the best fit for this dataset.

Confusion Matrices









Comparative Summary

- **XGBoost** slightly edges out as the best performer in every category, excelling in both **precision (avoiding false positives)** and **recall (minimizing false negatives)**.
- **Light GBM** is a very close second — nearly as accurate, but generally faster, making it attractive for large-scale or time-sensitive applications.
- **Gradient Boosting** performs well, especially in **balanced classification**, but lags behind the other two in absolute accuracy and might miss a few more relevant instances.

5. Overall Comparative Analysis

This capstone project involved working with three distinct datasets—**CSV**, **Image**, and **Text**—each requiring specialized preprocessing, modelling, and evaluation strategies. A comparative breakdown of each dataset's challenges, techniques, and outcomes is summarized below:

Aspect	**CSV Dataset (Health Impact)**	**Image Dataset (Multiclass Classification)**	**Text Dataset (Word Difficulty Classification)**
Data Type	Structured tabular data	Unstructured image data	Unstructured text data
Preprocessing	Outlier removal, feature scaling, label encoding	Grayscale conversion, resizing, normalization	Tokenization, stopword removal, word embedding (Word2Vec/GloVe)
Models Used	KNN, Random Forest, XGBoost	Custom CNN	UG Data Frame - LSTM, Gradient Boost, XGboost,LightGBM Classifier PG Data frame – LSTM, Decision Tree, Random Forest, SVC
Best Model	Random Forest	Custom CNN	ML Models (with Embeddings)
Evaluation Metrics	Accuracy, Precision, Recall, F1-score, Confusion Matrix	Accuracy, Loss, Class-wise Performance	Accuracy, Precision, Recall, F1-score, Confusion Matrix
Statistical Analysis	Z-test, T-test, ANOVA	Statistical test on class predictions	Performance comparison across models using classification report
Challenges	Handling imbalanced classes, outliers	High intra-class variance in image features	Semantic complexity, embedding quality
Outcome	Near-perfect accuracy using ensemble models	94%+ accuracy using CNN	Significant accuracy boost using word2vec embeddings

6. Conclusion

This project demonstrated the effective use of Python for end-to-end data analysis across diverse data types—tabular (CSV), image, and text. Each dataset presented unique challenges, from handling outliers and scaling features in the Air Quality Health Impact dataset, to designing a custom CNN for image classification, and applying deep learning with word embeddings for text-based difficulty classification.

Through systematic preprocessing, model building, and evaluation using relevant statistical and machine learning techniques, meaningful insights were extracted and strong predictive models were developed. Random Forest and XGBoost emerged as top performers in the structured data task, the CNN achieved high accuracy in image classification, and LSTM with embeddings significantly improved performance in the text task.

The project highlights the importance of tailored preprocessing, model selection, and metric-based evaluation in driving successful data-driven solutions. It reflects a comprehensive application of theoretical knowledge to real-world datasets and builds a strong foundation for future work in data science and AI.