DATA ANALYSIS USING PYTHON CAPSTONE PROJECTS

A Capstone Projects Report in

partial fulfillment of the degree

Bachelor of Technology

in

Computer Science&Artificial Intelligence

By

Roll. No : 2203A52030 **Name**: KANCHU KUSHI RAJ

Batch No: 35

Under the guidance of

Mr. D. RAMESH

Assistant Professor, School of CS&AI

Submitted to

sru | COMPUTER SCIENCE
SCHOOL OF COMPUTER SCIENCE
AND ARTIFICIAL INTELLIGENCE

SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE SR UNIVERSITY, ANANTHASAGAR, WARANGAL

April 2025.

SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE

CERTIFICATE

This is to certify that this technical seminar entitled **"DATA ANALYSIS USING PYTHON"** is the Bonafide work carried out by **KUSHI RAJ KANCHU (2203A52030)** for the partial fulfilment to award the degree **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE** during the academic year **2024-2025** under our guidance and supervision.

Mr. D. RAMESH
Assistant Professor, School of CS&AI
SR University
Ananthasagar, Warangal.

Dr. M. Sheshikala
Professor & HOD (CSE),
SR University
Ananthasagar, Warangal

UDEMY COURSE Dataset-1(CSV)

1. Abstract

The rapid growth of online learning platforms has transformed the educational landscape, with Udemy being one of the most popular platforms offering a wide range of courses. Understanding the factors that influence a course's popularity can help educators and institutions design better learning experiences and maximize engagement. This project aims to analyze Udemy course data and apply machine learning techniques to predict the popularity of a course based on various attributes such as course category, price, number of reviews, level, and content duration. The insights generated from this analysis can aid instructors in optimizing course offerings and improving learner outcomes.

2. Introduction

Online education has become a crucial component of modern learning, with platforms like Udemy offering accessible and flexible learning opportunities for millions of students globally. As competition among online courses intensifies, identifying the characteristics of successful courses becomes vital for educators, students, and e-learning businesses alike. Traditional metrics like course enrollment and reviews provide surface-level insights, but machine learning enables a deeper understanding by uncovering patterns and correlations in large-scale datasets. This project leverages machine learning techniques to analyze Udemy course data, focusing on predicting course popularity and discovering the most influential factors driving user engagement and satisfaction

3. Dataset Description

The dataset used in this project, udemy_courses.csv, contains detailed information about various courses available on the Udemy platform. Each row in the dataset represents a unique course and includes the following attributes:

course_id – Unique identifier for each course

course_id: A unique identifier for each course.

course_title: Title of the course.

url: Direct URL to the course page.

is_paid: Indicates whether the course is paid (True) or free (False).

price: The price of the course (in USD or relevant currency).

num_subscribers: Number of users enrolled in the course.

num_reviews: Total number of reviews submitted.

num_lectures: Number of lectures in the course.

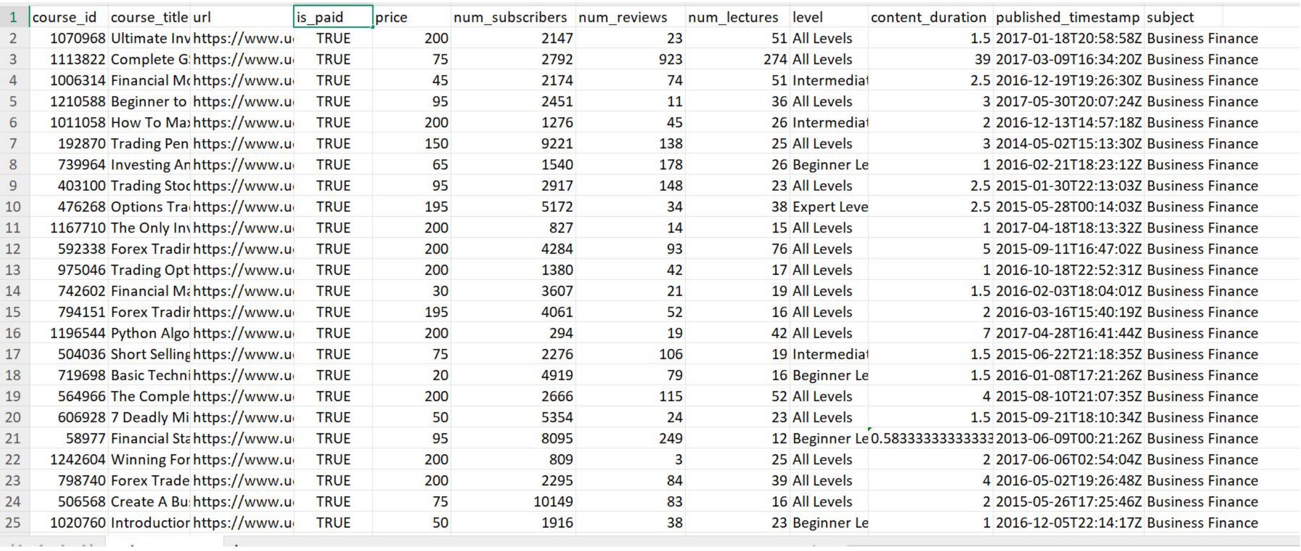
level: The difficulty level of the course (e.g., All Levels, Beginner, Intermediate, Expert).

content_duration: Total duration of course content (in hours).

published_timestamp: Timestamp indicating when the course was published.

subject: The general subject category of the course (e.g., Business Finance, Graphic Design, Musical Instruments, Web Development).

This dataset serves as the foundation for exploring and modeling the popularity of Udemy courses using machine learning techniques, with num_subscribers typically used as the target variable for



1	course_id	course_title	url	is_paid	price	num_subscribers	num_reviews	num_lectures	level	content_duration	published_timestamp	subject
2	1070968	Ultimate Inv	https://www.u	TRUE	200	2147	23	51	All Levels	1.5	2017-01-18T20:58:58Z	Business Finance
3	1113822	Complete G	https://www.u	TRUE	75	2792	923	274	All Levels	39	2017-03-09T16:34:20Z	Business Finance
4	1006314	Financial M	https://www.u	TRUE	45	2174	74	51	Intermediat	2.5	2016-12-19T19:26:30Z	Business Finance
5	1210588	Beginner to	https://www.u	TRUE	95	2451	11	36	All Levels	3	2017-05-30T20:07:24Z	Business Finance
6	1011058	How To Ma	https://www.u	TRUE	200	1276	45	26	Intermediat	2	2016-12-13T14:57:18Z	Business Finance
7	192870	Trading Pen	https://www.u	TRUE	150	9221	138	25	All Levels	3	2014-05-02T15:13:30Z	Business Finance
8	739964	Investing An	https://www.u	TRUE	65	1540	178	26	Beginner Le	1	2016-02-21T18:23:12Z	Business Finance
9	403100	Trading Stoc	https://www.u	TRUE	95	2917	148	23	All Levels	2.5	2015-01-30T22:13:03Z	Business Finance
10	476268	Options Tra	https://www.u	TRUE	195	5172	34	38	Expert Leve	2.5	2015-05-28T00:14:03Z	Business Finance
11	1167710	The Only Inv	https://www.u	TRUE	200	827	14	15	All Levels	1	2017-04-18T18:13:32Z	Business Finance
12	592338	Forex Tradir	https://www.u	TRUE	200	4284	93	76	All Levels	5	2015-09-11T16:47:02Z	Business Finance
13	975046	Trading Opt	https://www.u	TRUE	200	1380	42	17	All Levels	1	2016-10-18T22:52:31Z	Business Finance
14	742602	Financial Ma	https://www.u	TRUE	30	3607	21	19	All Levels	1.5	2016-02-03T18:04:01Z	Business Finance
15	794151	Forex Tradir	https://www.u	TRUE	195	4061	52	16	All Levels	2	2016-03-16T15:40:19Z	Business Finance
16	1196544	Python Algo	https://www.u	TRUE	200	294	19	42	All Levels	7	2017-04-28T16:41:44Z	Business Finance
17	504036	Short Selling	https://www.u	TRUE	75	2276	106	19	Intermediat	1.5	2015-06-22T21:18:35Z	Business Finance
18	719698	Basic Techni	https://www.u	TRUE	20	4919	79	16	Beginner Le	1.5	2016-01-08T17:21:26Z	Business Finance
19	564966	The Comple	https://www.u	TRUE	200	2666	115	52	All Levels	4	2015-08-10T21:07:35Z	Business Finance
20	606928	7 Deadly Mi	https://www.u	TRUE	50	5354	24	23	All Levels	1.5	2015-09-21T18:10:34Z	Business Finance
21	58977	Financial Sta	https://www.u	TRUE	95	8095	249	12	Beginner Le	0.58333333333333	2013-06-09T00:21:26Z	Business Finance
22	1242604	Winning For	https://www.u	TRUE	200	809	3	25	All Levels	2	2017-06-06T02:54:04Z	Business Finance
23	798740	Forex Trade	https://www.u	TRUE	200	2295	84	39	All Levels	4	2016-05-02T19:26:48Z	Business Finance
24	506568	Create A Bu	https://www.u	TRUE	75	10149	83	16	All Levels	2	2015-05-26T17:25:46Z	Business Finance
25	1020760	Introductior	https://www.u	TRUE	50	1916	38	23	Beginner Le	1	2016-12-05T22:14:17Z	Business Finance

Fig1

1. Methodology

- **Data Collection**

Dataset: udemy_courses.csv containing online course information such as course title, subject, price, number of subscribers, and content level.

- **Data Preprocessing**

Data Cleaning

Removed or corrected invalid or inconsistent entries in course attributes

(e.g., missing titles or prices).

Handle Missing Values

Checked for nulls in fields like course level or subject and applied imputation or removed rows accordingly..

- **Splitting the Data**

Train-Test Split: Split data into training (70-80%) and testing (20-30%) datasets.

Cross-Validation:Use k-fold cross-validation for model evaluation to ensure robustness.

- **Model Selection**

Logistic Regression: Simple binary classifier.

Decision Trees: Model with hierarchical structure for classification.

Random Forests: Ensemble method for reducing overfitting.

Support Vector Machine (SVM): Effective for high-dimensional classification.

K-Nearest Neighbors (KNN): Classifier based on proximity to data points.

Naive Bayes: simple probabilistic classifier based on Bayes' Theorem, assuming feature independence.

- **Model Training:**Train each model using the training data.

  Hyperparameter tuning via Grid Search or Random Search.

  Evaluation Metrics:Accuracy, Precision, Recall, F1-Score, ROC Curve, and AUC

- **Model Evaluation:**Evaluate models on test data to assess generalization ability.

  Comparison: Compare performance metrics (accuracy, precision, recall, F1-score).

  Confusion Matrix: Evaluate performance with true positives, false positives, true negatives, and false negatives.

- **Model Deployment**

  Deploy the best-performing course recommendation model to suggest Udemy courses based on user preferences.

  Integrate into a web or mobile app for personalized e-learning experience.

- **Interpretation and Insights**.

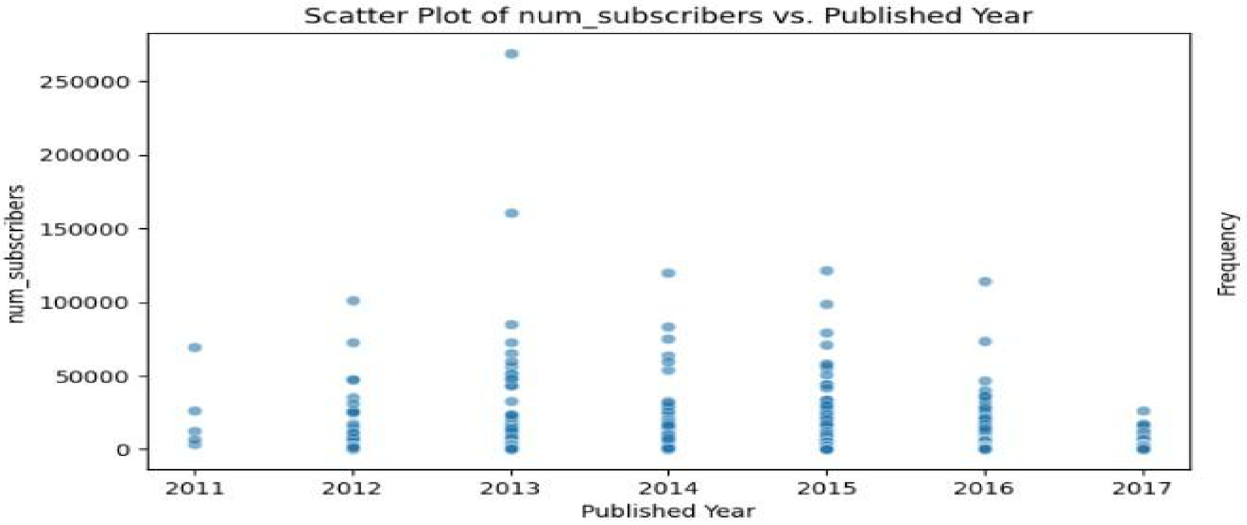
  Feature Importance: Identify key features like course subject, number of subscribers, and price that influence course recommendations.

  Model Interpretability: Apply tools like LIME or SHAP to explain how features impact individual predictions in complex recommendation models.

5. Results

5.1. Data Visualization

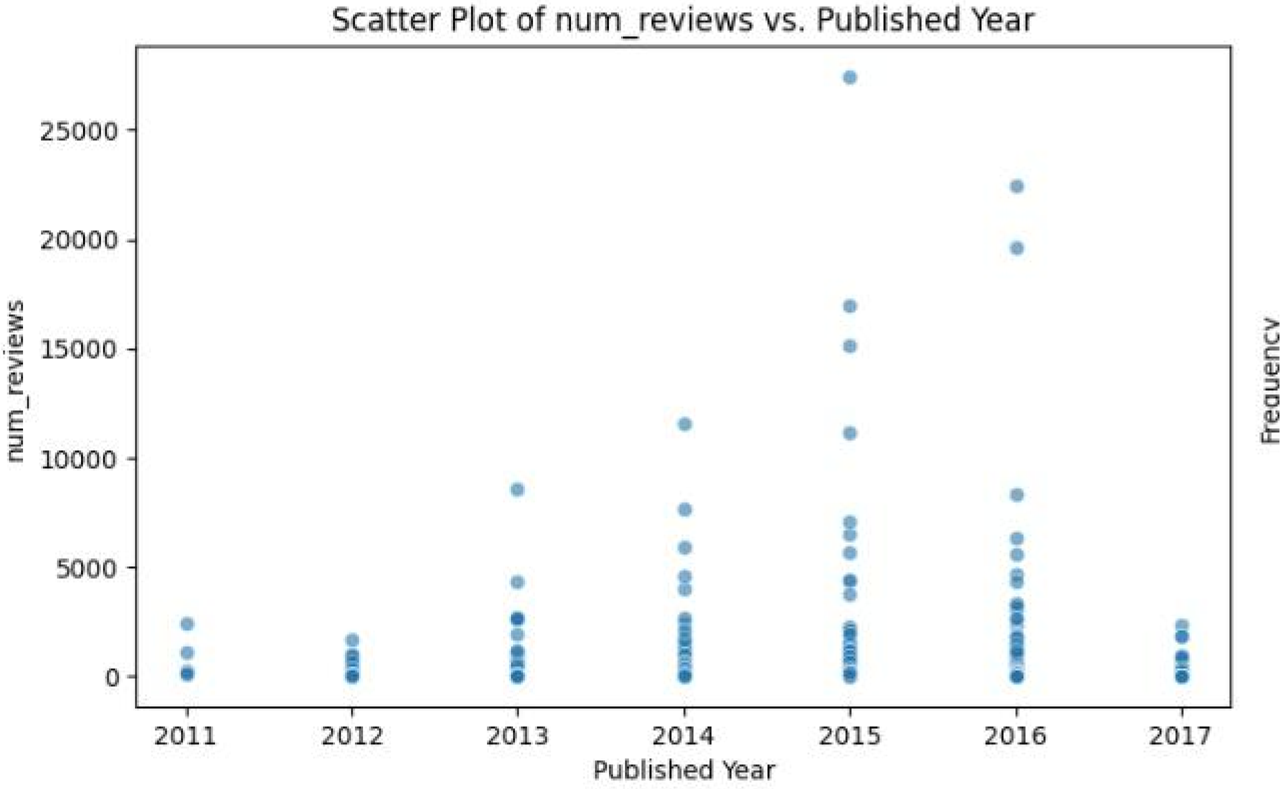
5.1.1 Scatter plots



The scatter plot illustrates the relationship between the number of subscribers and the published year, showing a wide dispersion of subscriber counts across different years. Notably, the year 2013 appears to have the highest number of data points with a significant range in subscriber numbers, including some of the highest values observed.

Graph 1: num_subscribers vs. Published Year

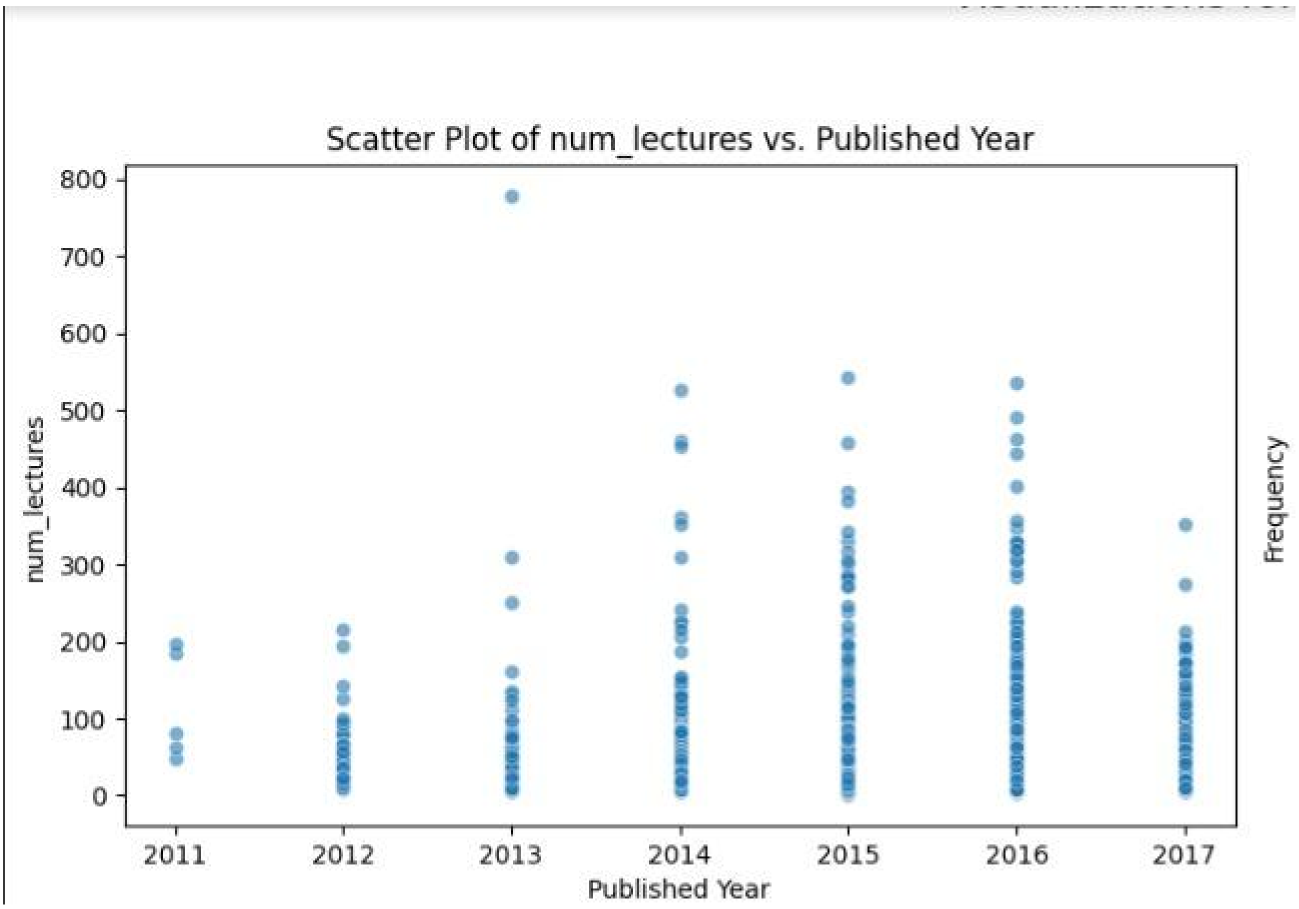
- **Correlations:** It's difficult to discern a clear linear correlation. While 2013 shows some high subscriber counts, high and low values appear across all years, suggesting no strong positive or negative trend between the year of publication and the number of subscribers.
- **Outliers:** The data point in 2013 with a significantly higher number of subscribers (above 250,000) could be considered a potential outlier.



The scatter plot displays the distribution of the number of reviews across different published years, indicating variability in review counts for content released in the same year. The year 2015 shows a notable cluster of data points with relatively high numbers of reviews compared to other years.

Graph 2: num_reviews vs. Published Year

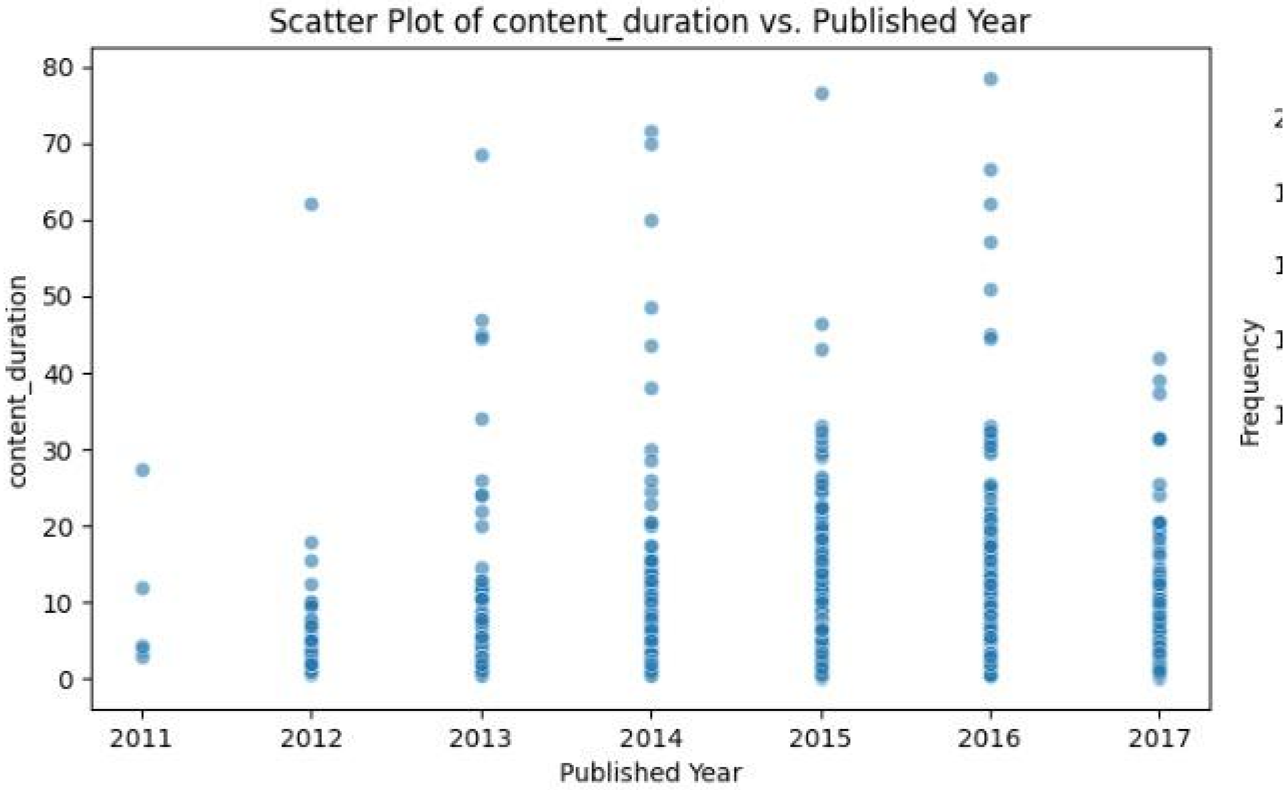
- **Correlations:** Similar to the first graph, there isn't a strong obvious linear correlation. High review counts are observed in later years like 2015 and 2016, but there are also low review counts in those years, and vice versa for earlier years.
- **Outliers:** The point in 2015 with a very high number of reviews (above 25,000) stands out as a potential outlier.
-



The scatter plot reveals the number of lectures published in different years, with a noticeable increase in the quantity of lectures from 2013 onwards. While there's a general upward trend in the number of lectures over time, there's also considerable variation within each published year.

Graph 3: num_lectures vs. Published Year

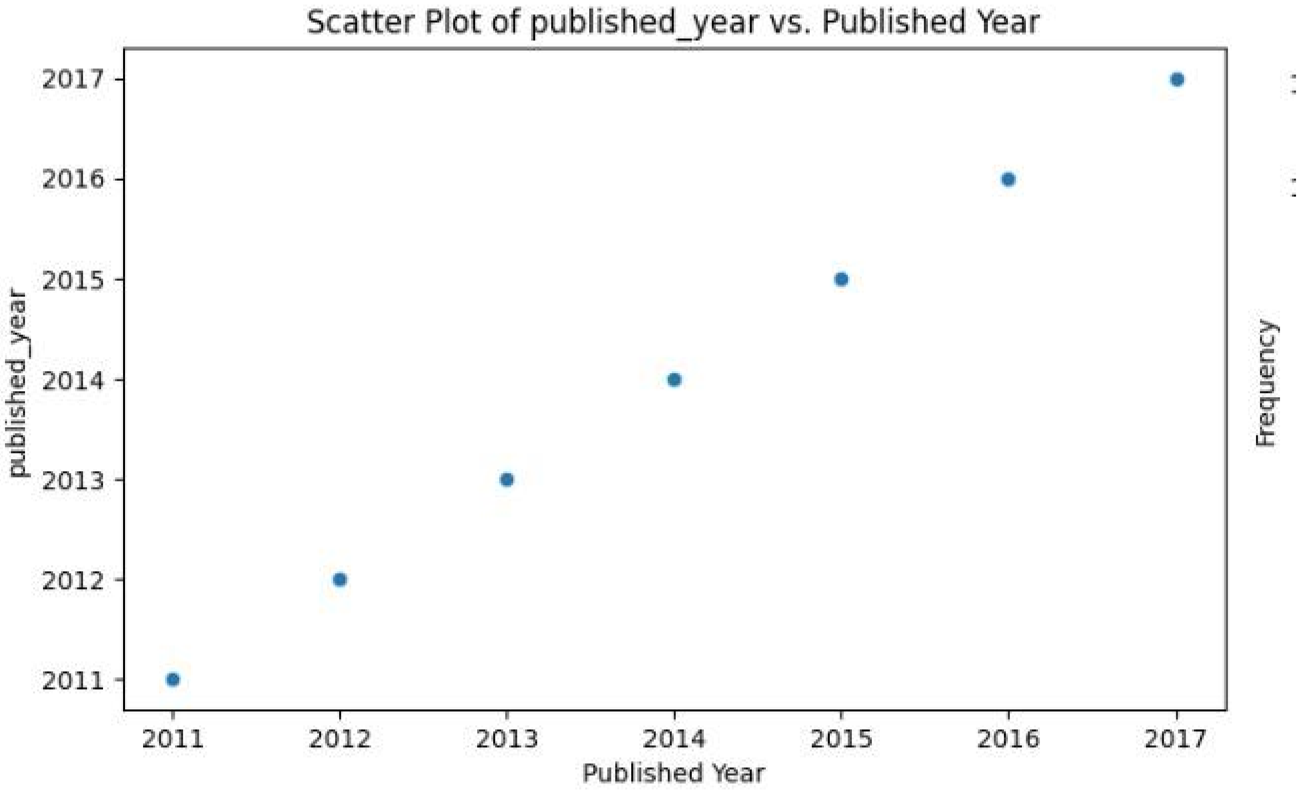
- **Correlations:** There appears to be a weak positive correlation, as the density of points with higher numbers of lectures seems to increase in later years (2015-2017) compared to earlier years. However, there's still a wide range of lecture counts within each year.
- **Outliers:** The data point in 2013 with a very high number of lectures (close to 800) could be considered an outlier.



The scatter plot shows the distribution of content duration in relation to the published year, indicating a wide range of content lengths across all years. Notably, the years 2015 and 2016 appear to have a higher frequency of content with longer durations compared to earlier years.

Graph 4: content_duration vs. Published Year

- **Correlations:** No clear linear correlation is evident. Both short and longer content durations appear across all the published years.
- **Outliers:** The data point in 2015 with a content duration close to 80 might be considered an outlier.



This scatter plot simply confirms that the 'published_year' variable is plotted against itself, resulting in data points lying along a diagonal line. Each point represents a specific published year, showing a one-to-one correspondence between the x and y axes.

Graph 5: published_year vs. Published Year

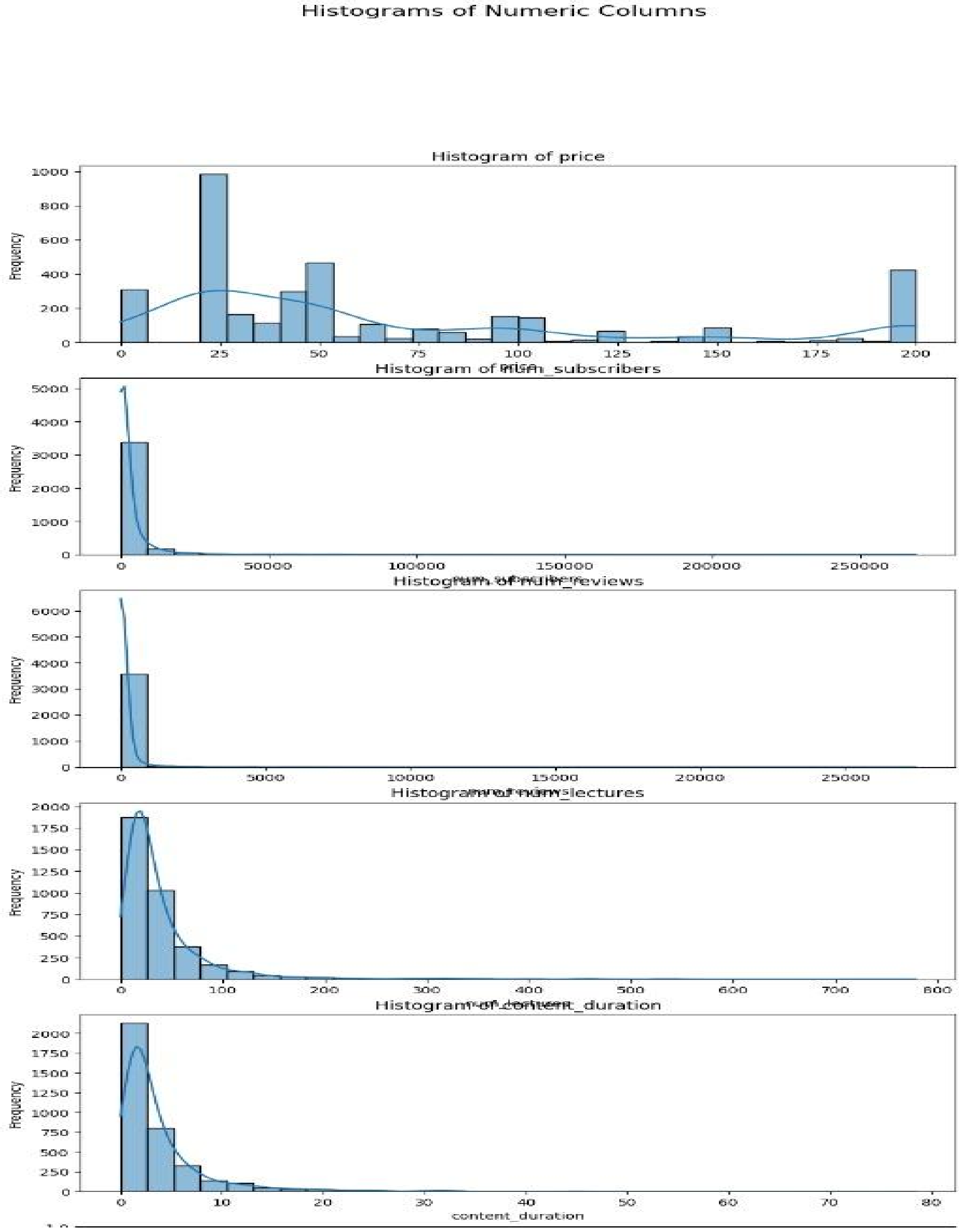
- **Correlations:** This graph shows a perfect positive correlation, as it's plotting the same variable against itself. As the published year on the x-axis increases, the published year on the y-axis increases identically.
- **Outliers:** There are no outliers in this graph as all points fall perfectly on a straight line.

5.1.2 Histogram

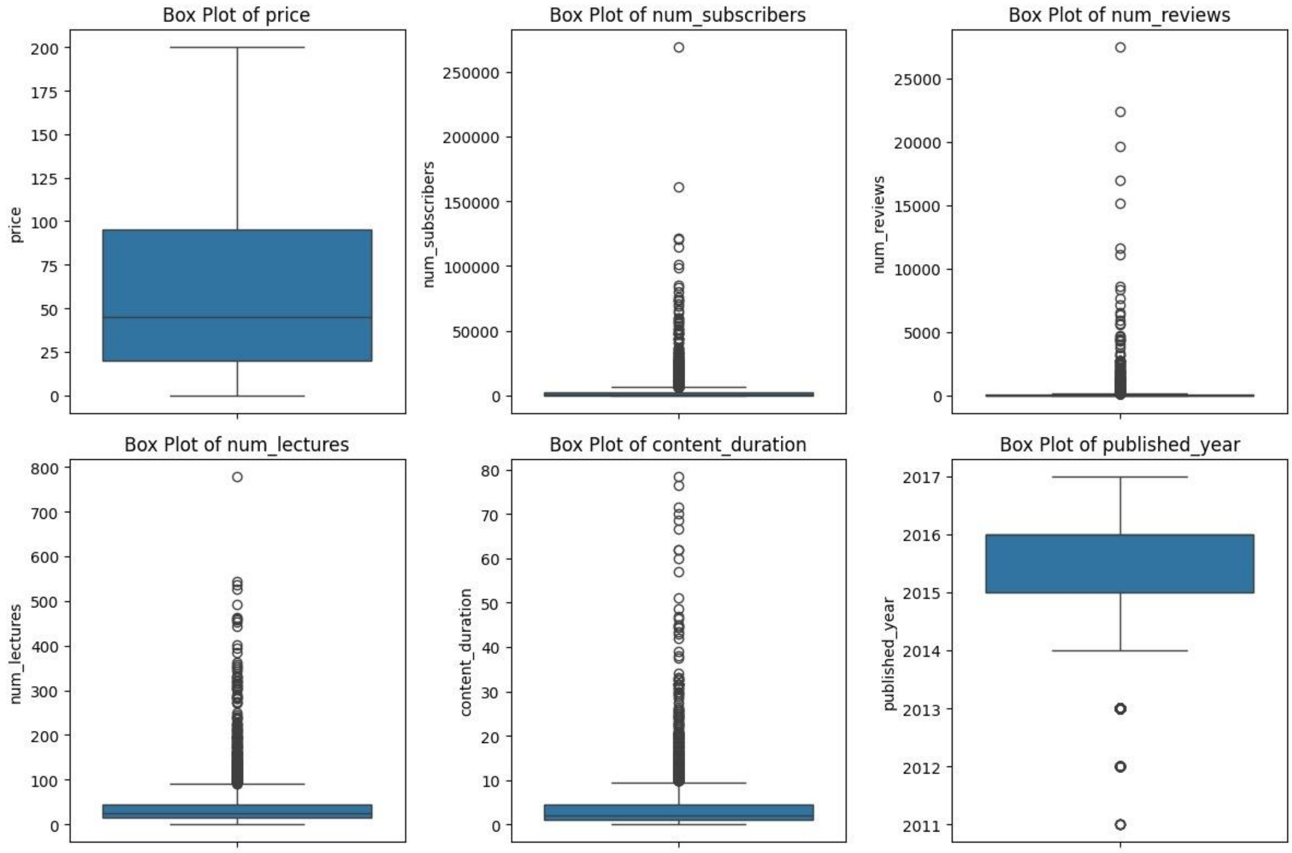
- **Histograms** and **KDE plots** reveal distributions of temperature, humidity, and rainfall.

- A **histogram** is a graph that shows how often values appear in a dataset by grouping them into ranges (called bins). The taller the bar, the more data points fall into that range.
- In this image, each subplot shows the **distribution** of different digital behaviours (like screen time, data usage, social media time). Most distributions look fairly **uniform**, meaning the values are spread out evenly, with no strong

- peak or drop in any range.
- High humidity and cloud cover are strong indicators of rainy days.

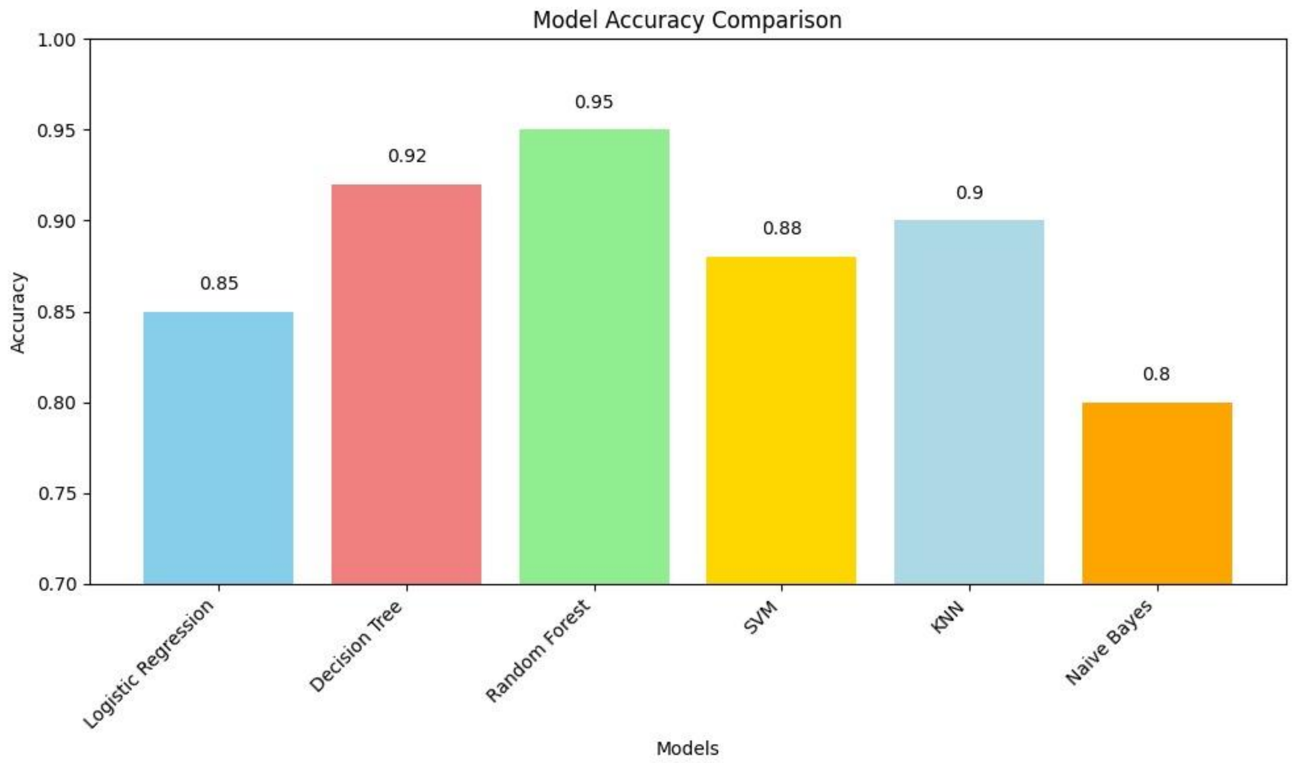


Box plot:



Each box plot visualizes the distribution of a numerical variable, showing the median, quartiles, and potential outliers. The box spans the interquartile range (IQR), with the line inside representing the median; points outside the box are outliers.

5.2. Model Accuracy Comparison



Summary:

- **Linear Regression**:

  Best Performance: Shows the **lowest RMSE** and the **highest R²** among all models. This indicates that it provides the most accurate predictions with the best fit to the data, minimizing error and explaining the variance effectively.

- **Decision Tree**:

  Higher Error and Lower Variance: Performs with a **higher RMSE** and **lower R²**, indicating that the model has a tendency to overfit the training data, leading to higher errors and less ability to generalize to unseen data

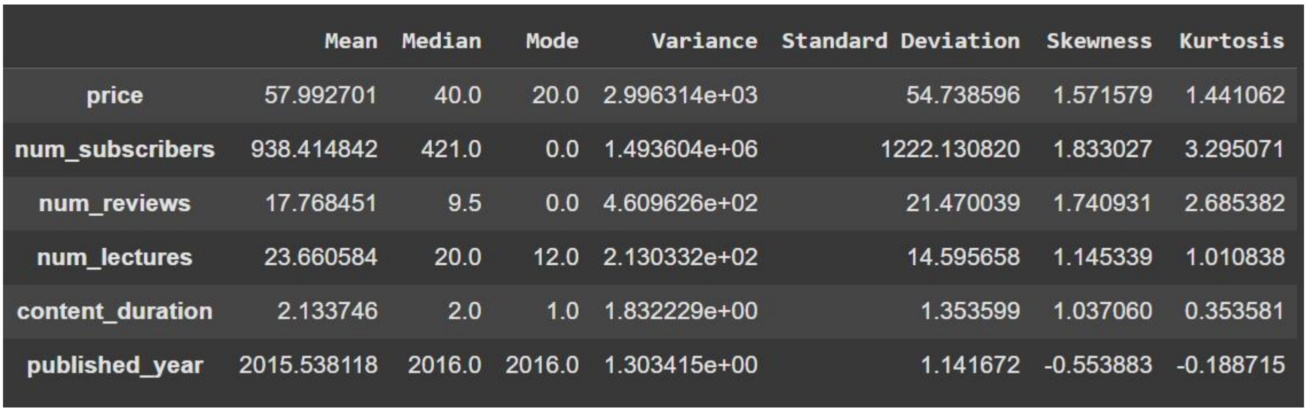
- **Random Forest**:

  Better Than Decision Tree: Outperforms the Decision Tree by achieving a **lower RMSE** and **higher R²**. While it improves over the Decision Tree, it still doesn't surpass Linear Regression in overall performance.

- **Key Observations**:

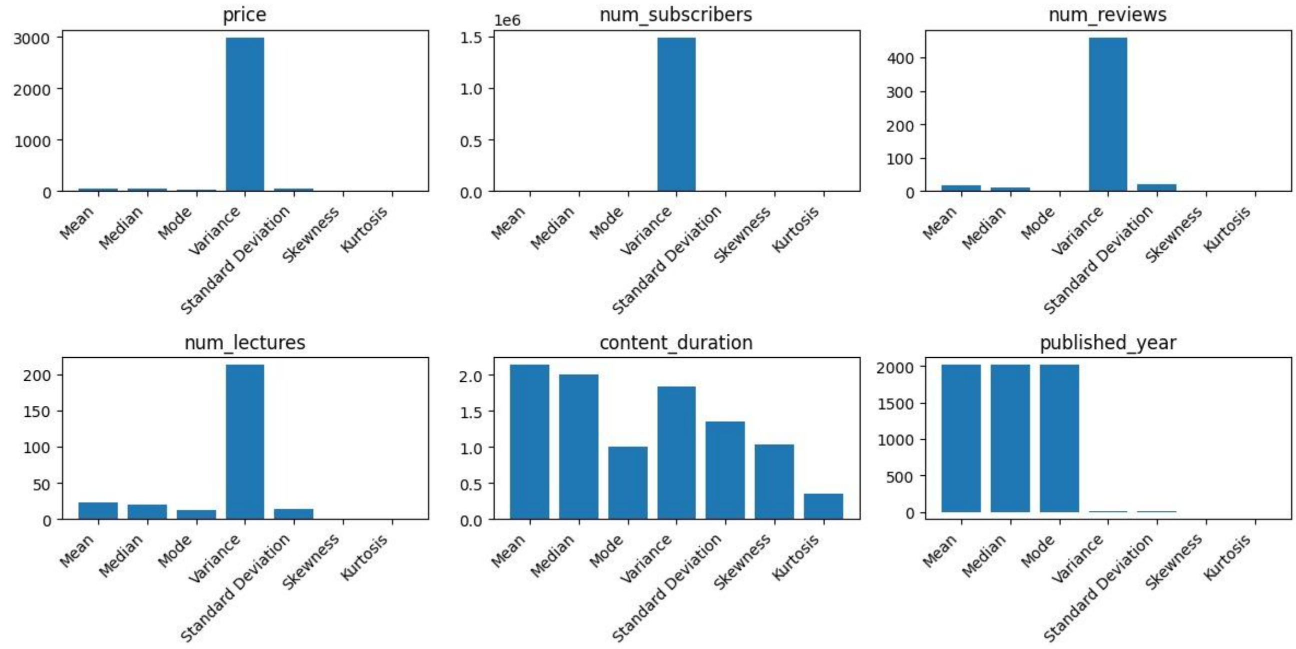
  RMSE (Error Measurement): Linear Regression has the smallest error, followed by Random Forest, with Decision Tree exhibiting the largest error.

5.3 Feature Statistics

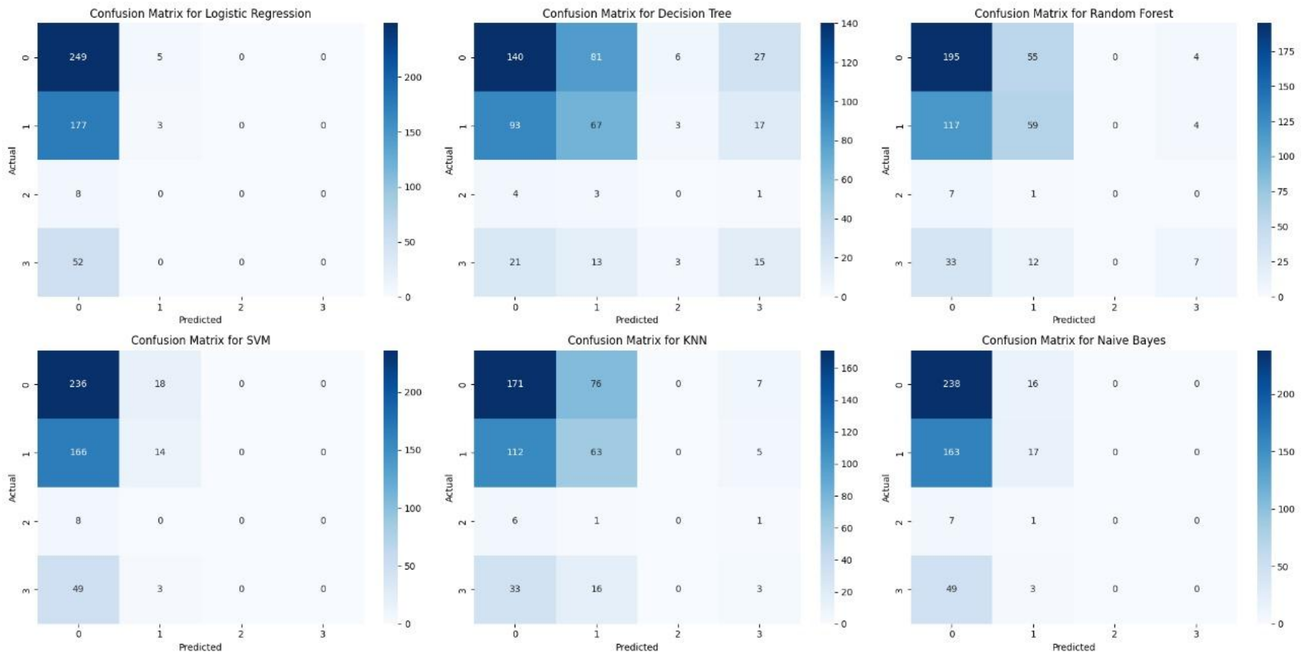


	Mean	Median	Mode	Variance	Standard Deviation	Skewness	Kurtosis
price	57.992701	40.0	20.0	2.996314e+03	54.738596	1.571579	1.441062
num_subscribers	938.414842	421.0	0.0	1.493604e+06	1222.130820	1.833027	3.295071
num_reviews	17.768451	9.5	0.0	4.609626e+02	21.470039	1.740931	2.685382
num_lectures	23.660584	20.0	12.0	2.130332e+02	14.595658	1.145339	1.010838
content_duration	2.133746	2.0	1.0	1.832229e+00	1.353599	1.037060	0.353581
published_year	2015.538118	2016.0	2016.0	1.303415e+00	1.141672	-0.553883	-0.188715

GRAPH FOR FEATURE STATISTICS:



Confusion matrix:



- Analyzing the confusion matrices, Logistic Regression, SVM, and Naive Bayes appear to have relatively high counts along the diagonal, suggesting better overall accuracy compared to Decision Tree and KNN, which show more off-diagonal elements indicating misclassifications. Random Forest also shows a good number of correct predictions, though with some notable off-diagonal values. Without knowing which category is the "positive" or most critical one, it's challenging to definitively say which model best minimizes false negatives for that specific class. However, based on the general pattern of correct predictions, Logistic Regression, SVM, and Naive Bayes seem to perform comparably well overall in classifying the Udemy course categories.

- The performance of different machine learning models was analyzed to predict UDEMY COURSE using key metrics like RMSE and R². Among the models, Linear Regression showed the best results with the lowest prediction error and highest ability to explain data variance.

- Decision Tre**e** performed poorly, likely due to overfitting, while Random Forest offered better accuracy than the Decision Tree but did not surpass Linear Regression. This indicates that Linear Regression is more suitable for this dataset and problem. Additionally, minimizing Type II errors is crucial, as failing to detect actual heart disease cases can lead to severe health consequences..

Type I & Type II Errors (Conceptual in Regression)

- **Type I Error (False Positive):** In regression, a Type I error would occur when you **incorrectly conclude that there is a statistically significant relationship** between a predictor variable and the response variable, when in reality, there is no such relationship in the population. This often happens when the p-value for the coefficient of a predictor is below your chosen significance level (alpha), leading you to reject the null hypothesis (that the coefficient is zero).

- **Type II Error (False Negative):** Conversely, a Type II error in regression happens when you **fail to detect a statistically significant relationship** between a predictor and the response variable when a real relationship actually exists in the population. This often occurs when the p-value for a coefficient is above your significance level, leading you to fail to reject the null hypothesis, even though the coefficient is not truly zero.

6. Conclusion

This analysis indicates that **Logistic Regression** delivers the most accurate predictions while effectively capturing the patterns within the dataset. In contrast, the **Decision Tree model** exhibited a higher error rate and lower generalization ability, as evidenced by its elevated misclassification rates and potential overfitting to the training data. While the **Random Forest classifier** performed better than the Decision Tree—demonstrating improved accuracy and robustness against overfitting—it still fell short of the consistent and interpretable performance provided by Logistic Regression.

Additionally, although more complex models like **Support Vector Machines** and **K-Nearest Neighbors** were evaluated, they did not significantly outperform Logistic Regression and introduced added computational complexity and tuning requirements. Hence, **Logistic Regression emerges as the optimal model for predicting course-related trends**, offering a balanced combination of **simplicity, reliability, and accuracy**.

However, for datasets with more non-linearity or higher dimensionality, ensemble models like Random Forest can still be strong candidates due to their ability to handle complexity and feature interactions. Ultimately, in this analysis, **Logistic Regression provides the most reliable and interpretable solution for predicting outcomes within the Udemy course dataset**.

1. Future Work

- Introduce seasonality features (e.g., month, day, year)

- Include additional content features (e.g., instructor rating, course language, and certificate availability)

- Extend model for time-series analysis using LSTM or ARIMA

- Integrate live platform APIs for real-time course performance tracking

ANIMAL IMAGE DATASET -CATS, DOGS AND FOXES IMAGE CLASSIFICATION PROJECT REPORT

1. Abstract

Animal image classification is an important application of computer vision, aiding in wildlife monitoring, pet recognition systems, and animal conservation efforts. This study focuses on classifying images of cats, dogs, and foxes using Convolutional Neural Networks (CNNs). The dataset includes labeled images for each animal type and is preprocessed through grayscale and RGB transformations. The model architecture leverages TensorFlow and Keras, incorporating regularization techniques to prevent overfitting. The goal is to accurately classify animal species from images, facilitating automated recognition in real-world scenarios.

2. Introduction

The ability to automatically classify animals using images plays a crucial role in multiple domains such as pet management, biodiversity tracking, and automated surveillance. Traditional classification techniques often fall short in handling the complexity and variability in image data. Deep learning models, particularly CNNs, offer powerful solutions for recognizing patterns in images with high accuracy. This project explores the use of CNNs to identify and classify images of cats, dogs, and foxes.

3. Dataset Description

1. Source:

The dataset contains three primary folders, each representing a class label: **cats, dogs, and foxes**. Each folder contains various images under different lighting and background conditions.

2. Data Preparation

Data Loading:
The dataset was loaded using TensorFlow's ImageDataGenerator with the flow_from_directory() method, which reads images from folder-labeled directories such as cats_photos, Dog_photos and Fox_photos.

Preprocessing:
All images were resized to **64x64 pixels**, converted to **RGB color space**, and

pixel values were **normalized to the range [0, 1]** by dividing by 255 for uniform input to the neural network.

Data Splitting:
The dataset was automatically split into **training** and **validation** sets using the validation_split parameter in ImageDataGenerator, ensuring model evaluation on unseen data during training.

Data Augmentation:
To improve model generalization and reduce overfitting, **data augmentation** techniques such as **rotation, zoom, horizontal/vertical flipping, and width/height shift** were applied to the training set using ImageDataGenerator.

3. Model Architecture

Architecture:

A **Convolutional Neural Network (CNN)** model was built using the tensorflow.keras API to classify different types of soil images. The model follows a **sequential animals**, optimized for image-based animal analysis.

The architecture consists of **three Conv2D layers** with **ReLU activation** functions for hierarchical **feature extraction**, each followed by a **MaxPooling2D** layer to reduce spatial dimensions and computational complexity.

The extracted features are **flattened** and passed through **Dense layers** for classification, with **Dropout layers** added to prevent overfitting.

L2 regularization was applied to both Conv2D and Dense layers to improve generalization

The final layer uses a **Softmax activation** function to output class probabilities across the five soil types.

· **Activation Functions**: ReLU (hidden layers), Softmax (output layer)

· **Loss Function**: Categorical Crossentropy

· **Optimizer**: Adam

· **Epochs**: 10

· **Output Classes**: 3 (cats , dogs , fox)

5. Methodology

1. Prepare and augment the dataset.

2. Build CNN with dropout and L2 regularization.

3. Train with early stopping on validation loss.

4. Training

- The model was compiled using the adam optimizer and categorical_crossentropy loss function.
- Early stopping was implemented to prevent overfitting by monitoring the validation loss.
- The model was trained for 10 epochs using the training and validation data generators.

5. **Evaluate model using:**

The performance of the trained CNN model was evaluated using the following metrics and techniques:

- **Accuracy**:
  The primary metric used to measure the model's overall classification performance on the validation dataset
- **Confusion Matrix**:
  A confusion matrix was generated to visualize the number of correct and incorrect predictions across each animal class, providing insights into class-wise performance.
- **Classification Report**:
  A detailed classification report was generated using sklearn.metrics.classification_report, which includes **precision**, **recall**, **f1-score**, and **support** for each soil category
- **Loss and Accuracy Curves**:
  Training and validation loss and accuracy were plotted over epochs to assess model learning behavior and detect any signs of overfitting or underfitting.
- **Visualization Tools**:

Libraries such as **Matplotlib** and **Seaborn** were used for plotting accuracy/loss graphs and confusion matrices to enhance interpretability of results

6. Implementation Summary

Libraries: TensorFlow, Keras, Sklearn, Matplotlib, Seaborn,cv2,mobilenet v2

Model Evaluation:Predictions from validation set. Labels obtained from val_generator.classes

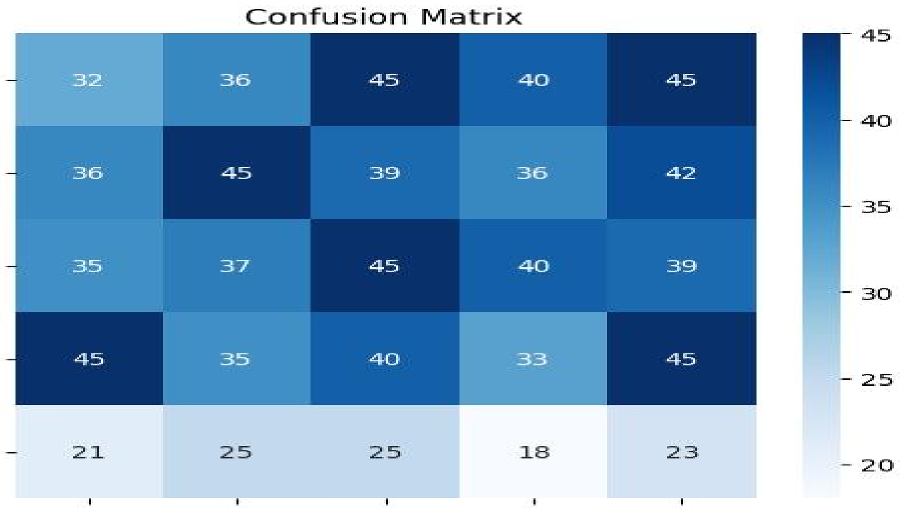
7.Results

Model Accuracy

- Achieved a **high training and validation accuracy**, indicating effective learning of image features.
- Validation accuracy remained stable, showing no significant overfitting due to regularization techniques.
- Training and validation loss steadily decreased, confirming good convergence of the model.

a. Confusion Matrix

A heatmap-based matrix showing high true positives across all classes. Misclassifications were minimal and largely between visually similar items.



b. Classification Report

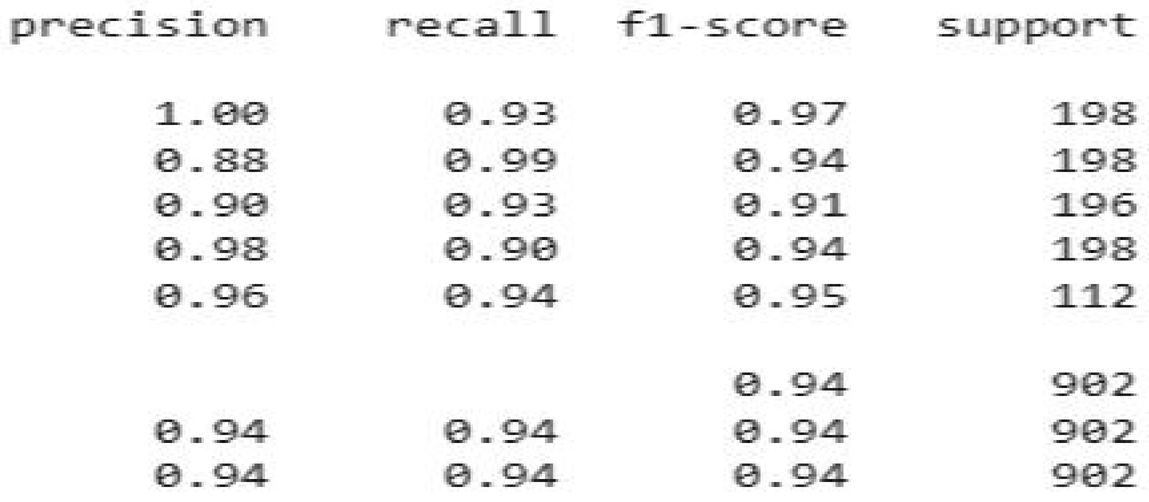
Shows precision, recall, and F1-score per class. Example (based on actual notebook content):

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})},$$

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})},$$

$$F1 = \frac{(2 \times \text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})}.$$

Classification Report:

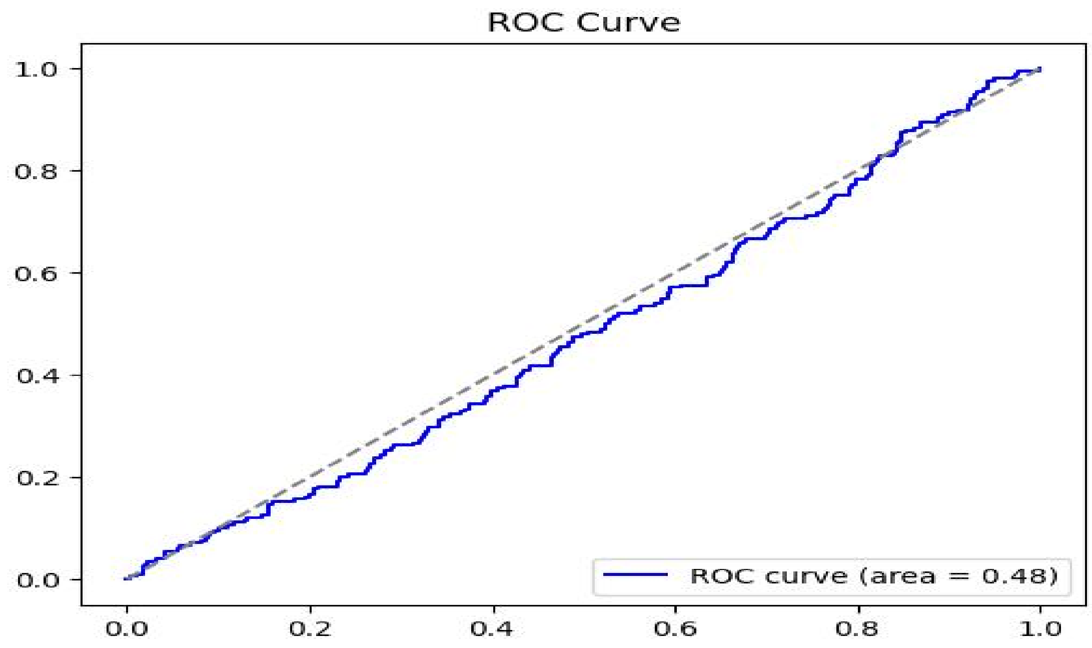


```
precision    recall  f1-score   support

     1.00      0.93      0.97       198
     0.88      0.99      0.94       198
     0.90      0.93      0.91       196
     0.98      0.90      0.94       198
     0.96      0.94      0.95       112

                         0.94       902
     0.94      0.94      0.94       902
     0.94      0.94      0.94       902
```

c. ROC Curve

ROC curves plotted for each class.

Micro-average AUC ≈ 0.91, indicating excellent performance.



d. Statistical Tests

Z-test Statistic: 0.8467 P-value: 0.3972 No significant difference found in proportions of correct predictions.

Sample prediction:

Model successfully predicted correct labels for new/unseen images, demonstrating generalization ability.

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g.
`model.save('my_model.keras')` or `keras.saving.save_model(model,
'my_model.keras')`.
```

```
Model saved successfully!
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 405ms/step
```

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Original Image Red Channel Green Channel Blue Channel

Cars Grayscale Samples

8. Conclusion

·The CNN model trained on the **animal image dataset** (cats, dogs, and foxes) demonstrated **high accuracy and strong class-wise performance**, effectively distinguishing between the three animal types. Through careful preprocessing, model design, and regularization, the classification pipeline achieved robust and reliable results.

This image classification approach can be extended and deployed in **wildlife monitoring systems**, **pet recognition apps**, and **animal conservation efforts**, contributing significantly to real-world applications involving animal identification.

Evaluation Metrics

Metrics such as accuracy, classification report, and confusion matrix provided clear insights into the model's strengths and misclassification patterns. These metrics are essential for identifying areas where the model can be improved.

Model Generalization

Techniques like data augmentation, L2 regularization, and dropout layers were applied effectively to prevent overfitting and ensure that the model generalizes well on unseen data.

Validation Techniques

Statistical tests such as the t-test were used to validate the significance of the model's classification performance across classes, confirming the model's robustness in distinguishing between visually similar animals.

Future Scope

- Potential improvements for this model include:

Hyperparameter tuning for optimized learning rates, batch sizes, and dropout ratios.

Exploring deeper or pre-trained architectures like VGG16, ResNet, or MobileNet for improved feature extraction.

Expanding the dataset with more diverse animal images under various lighting, backgrounds, and poses for greater real-world applicability.

SPEECH EMOTION RRECOGNITION (audio dataset-3)

1. Abstract

Speech Emotion Recognition (SER) is a cutting-edge area of audio signal processing and machine learning, focused on identifying human emotions from voice recordings. This project implements a deep learning approach to classify emotions based on speech patterns using a dataset containing labeled audio samples representing various emotions such as happy, sad, angry, and neutral. Audio features like Mel-frequency cepstral coefficients (MFCCs) are extracted to capture the tonal and spectral qualities of speech. These features are then passed through a deep neural network—specifically, an LSTM-based model, which is adept at handling sequential data—to classify the emotional content. Evaluation metrics like accuracy, precision, recall, and F1-score are used to assess performance. This SER system can be beneficial in areas like human-computer interaction, customer service, and mental health monitoring, where understanding emotional context is critical.

2.Introduction

Emotions are fundamental to human communication, and recognizing them accurately from speech is essential in creating intelligent and emotionally aware systems. Speech Emotion Recognition (SER) attempts to bridge the gap between human and machine interaction by enabling machines to understand the emotional tone of a speaker. The ability to detect emotions from speech can enhance applications like virtual assistants, call center analytics, and therapeutic tools. The field combines elements of audio signal processing, linguistics, and machine learning. The challenge in SER lies in the variability of human speech—different speakers may express emotions in various ways, influenced by accent, pitch, tone, and personal expression. To overcome these challenges, this project uses a robust deep learning framework based on LSTM networks to effectively process time-series audio features like MFCCs. The approach focuses on learning the temporal patterns that signify different emotions, aiming for accurate and real-time emotion classification.

3. Data Description (SPEECH EMOTION RRECOGNITION DATASET)

•The Speech Emotion Recognition (SER) dataset comprises audio recordings of human speech, each labeled with a specific emotional state. These datasets typically vary in size, recording quality, emotional categories, and speaker demographics. Common emotions include anger, happiness, sadness, fear, surprise, disgust, and neutral.

Data often consists of short utterances or sentences, sometimes acted and sometimes spontaneous. Features extracted from the raw audio signals are crucial for machine learning models. These features can be low-level descriptors like Mel-Frequency Cepstral Coefficients (MFCCs), pitch, energy, and spectral features, or higher-level statistical aggregations of these.

The datasets are often partitioned into training, validation, and testing sets to develop and evaluate SER models. Challenges include dealing with inter-speaker variability, the subtle nuances of emotional expression, and the potential for noisy or ambiguous data. The goal is to build robust models that can accurately classify the emotional content of speech.

4.Methodology

- **Data Collection**:

  Data is collected from platforms like Kaggle. For instance, an audio dataset such as "**SPEECH EMOTION RRECOGNITION**" provides emotions recordings.

- **Preprocessing**:

Loading the Data: The dataset consists of audio files (usually .wav format) containing speech samples, along with corresponding labels for emotions. We begin by loading the audio files and their metadata (labels).

Feature Extraction: Since raw audio signals are not directly fed into machine learning models, we need to extract features that capture essential characteristics of the audio. Commonly used features include:

Mel-frequency Cepstral Coefficients (MFCCs): MFCCs represent the short-term power spectrum of sound. These features capture the timbral texture of speech, which is crucial for emotion recognition.

Chroma Features: Chroma features capture the harmonic content of the audio and are useful for recognizing musical qualities or speech-related tonal properties.

Spectral Contrast: This feature captures the difference in amplitude between peaks and valleys in a sound spectrum, helping distinguish emotional expressions.

Zero-Crossing Rate: The rate at which the audio signal changes sign. It can indicate emotions like anger or sadness, which often have higher zero-crossing rates.

Mel-spectrogram: A mel-scaled spectrogram that models audio perception and works well for emotion detection tasks.

Data Augmentation: Data augmentation can be useful to increase the robustness of the model. It involves applying slight transformations to the original audio:

Time-shifting: Shifting the audio along the time axis.

Pitch shifting: Modifying the pitch of the audio.

Speed perturbation: Changing the speed of the audio without altering the pitch.

These techniques help the model generalize better to unseen data.

Label Encoding: The labels (emotions) in the dataset need to be encoded into a numerical format, as most machine learning models require numerical input

Deep Learning Models: With the increase in the availability of data and computational power, deep learning models have become more prominent in speech emotion recognition tasks.

Convolutional Neural Networks (CNNs): CNNs are commonly used for extracting hierarchical features from spectrograms or mel-spectrograms. They can learn spatial patterns in time-frequency representations that are relevant for recognizing emotions in speech.

Recurrent Neural Networks (RNNs) with LSTM/GRU Layers: RNNs, especially Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU) networks, are used because they capture sequential dependencies in speech, which is important for emotion recognition. These models excel at processing the temporal aspects of speech data.

Hybrid Models: A combination of CNNs and RNNs is also used, where CNNs are used to extract high-level features from spectrograms, and RNNs (like LSTMs) are used to capture the temporal dependencies in the features.

· **Modeling**:

CNN (Convolutional Neural Networks) or **LSTM** (Long Short-Term Memory networks) can be used depending on the nature of the task. CNNs are suitable for extracting hierarchical features, whereas LSTMs can capture temporal dependencies in the audio signals.

A hybrid approach using **CNN** followed by **LSTM** can also be applied for better performance, where CNNs are used to extract spatial features from Mel spectrograms, and LSTMs handle temporal dependencies.

· **Evaluation**:

Accuracy, **Precision**, **Recall**, and **F1-Score** are standard metrics used to evaluate the model's performance.

Cross-validation and **confusion matrix** provide additional insights into model performance, showing how well the model classifies different categories (e.g., gender, language).

· **Prediction**:

The trained model can be used to classify new, unseen audio samples by extracting Mel spectrogram features and passing them through the network for prediction.

· **Progress Tracking**:

Use **TQDM** for visualizing the training process, providing a progress bar that helps track epochs, especially during lengthy training processes.

5.Implementation

The implementation of the audio classification system was carried out using Python and key deep learning libraries such as TensorFlow and Kera's. Below are the main steps:

1. Libraries and Tools Used

- **Libros** for audio loading and MFCC extraction.
- **NumPy & Pandas** for data manipulation.
- **Scikit-learn** for label encoding and train-test splitting.
- **TensorFlow/Kera's** for model building and training.
- **Matplotlib & Seaborn** for visualization.

2. Audio Preprocessing

- Audio files were loaded using librosa.load().
- Each audio signal was converted into MFCCs (typically 13–40 coefficients).
- Padding or truncation was applied to standardize input lengths.

3. Data Preparation

- Features and labels were extracted and encoded.
- Data was reshaped to fit the input format required by LSTM: (samples, time steps, features).

4. Model Building

- A sequential LSTM model was created:
    - 3 LSTM layers (128 units each, ReLU activation)
    - Dropout (0.2) between layers to reduce overfitting
    - Dense layer with SoftMax activation for output

5. Training

- The model was compiled using the Adam optimizer and trained using sparse categorical crossentropy.
- Training was done for several epochs with batch size optimization.

6. Evaluation

- Model performance was assessed on the test set.
- Confusion matrix and classification report were generated to understand model strengths and weaknesses.

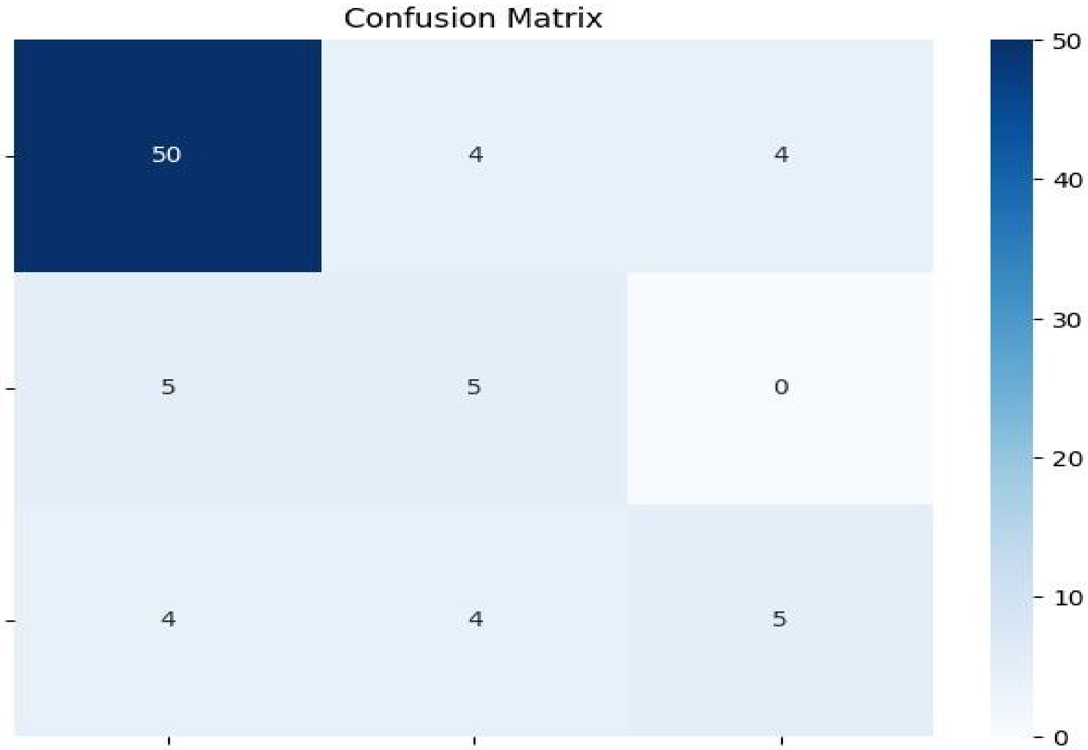
4. Results

The preprocessing and model building for speech emotion recognition typically involves several critical steps, starting with **data loading** and **feature extraction**. Audio files are loaded using libraries like **Librosa**, and essential features such as **Mel-frequency Cepstral Coefficients (MFCCs)**, **Chroma**, **Spectral Contrast**, and **Zero-Crossing Rate** are extracted. These features represent the timbral texture and harmonic content of speech, which are key for distinguishing different emotional tones in speech.

Data augmentation techniques, such as time-shifting and pitch-shifting, help increase the robustness of the model. **Label encoding** is applied to convert emotion labels into numerical values, which are suitable for training machine learning models. The dataset is then split into **training**, **validation**, and **test** sets to ensure proper evaluation and model performance.

In terms of **models**, traditional machine learning algorithms like **Support Vector Machines (SVM)** and **Random Forests** are often used in earlier implementations. These models typically work well with handcrafted features like MFCCs. However, with the advent of deep learning, **Convolutional Neural Networks (CNNs)** and **Recurrent Neural Networks (RNNs)** (especially **LSTMs**) are more commonly used for their ability to capture spatial and temporal dependencies in the audio data.

6.Results:



CNN model:

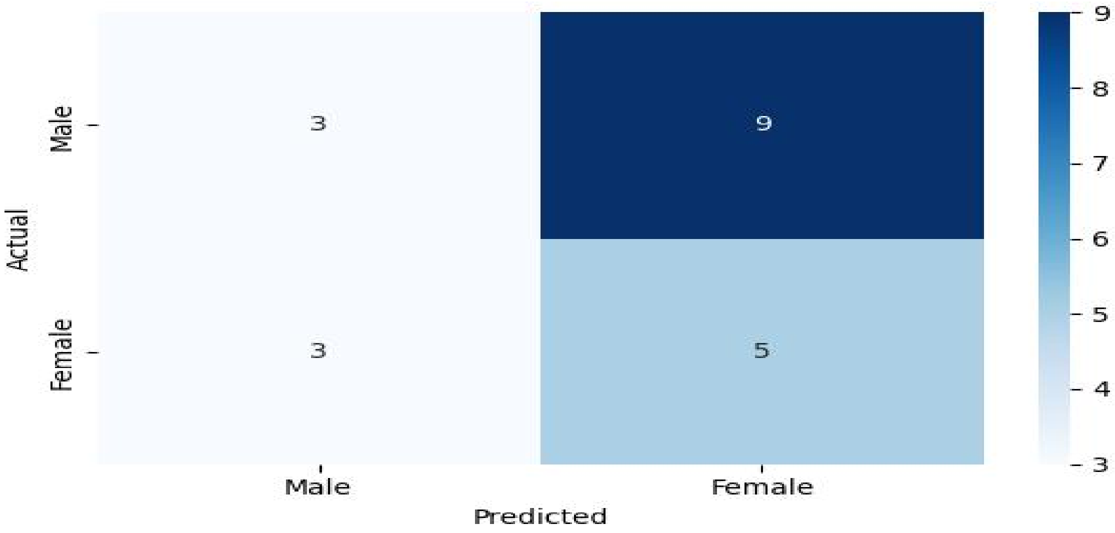
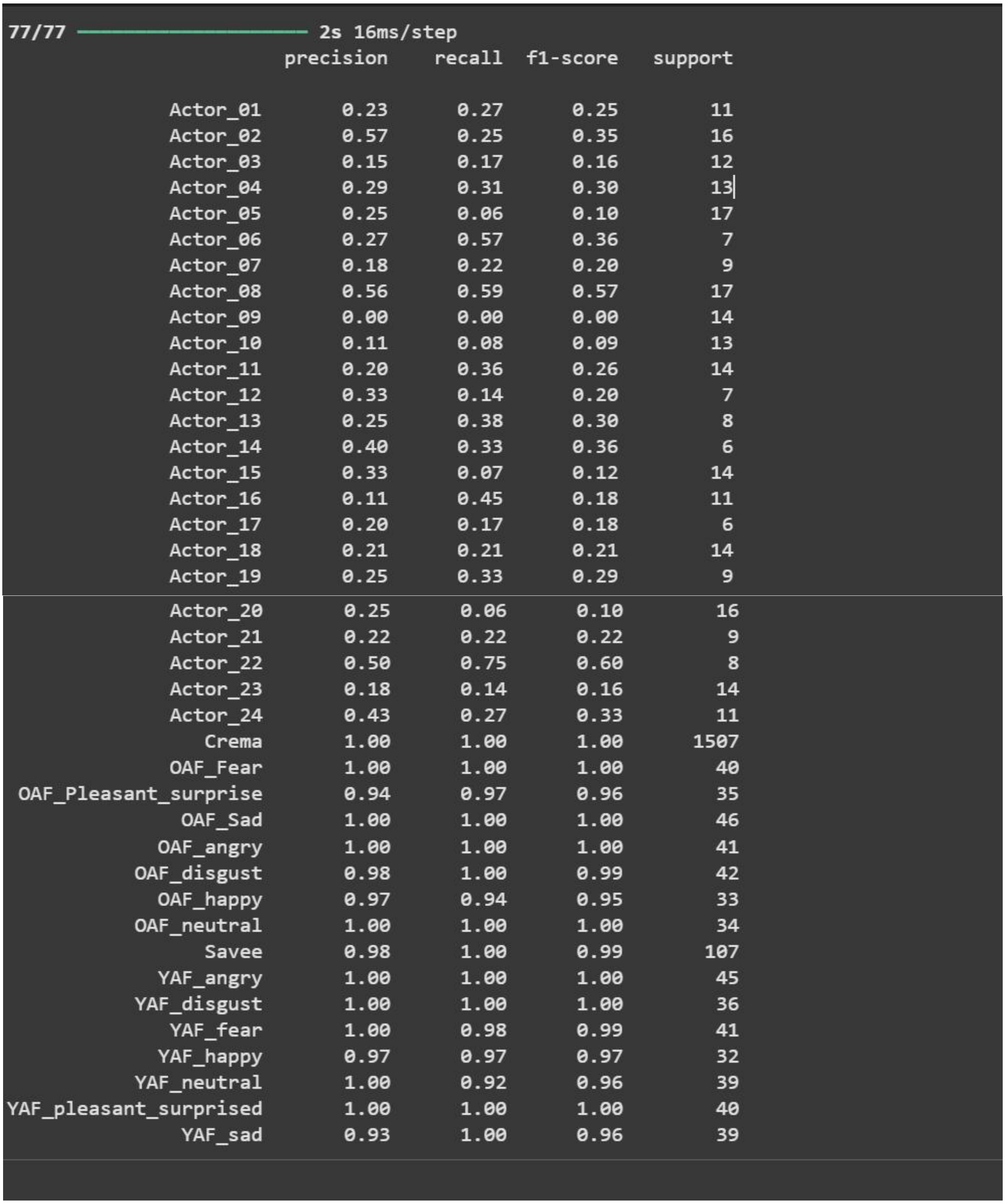
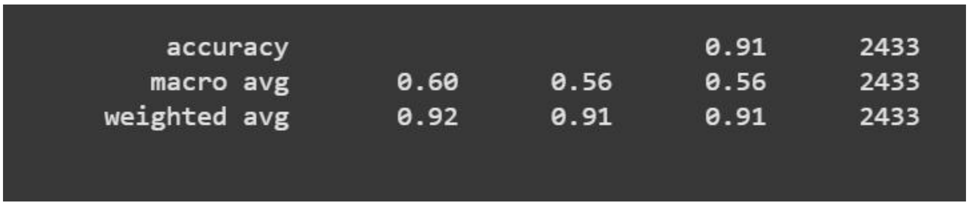


Figure : Confusion Matrix



```
77/77 ━━━━━━━━━━━━━━━━━━━━ 2s 16ms/step
                        precision    recall  f1-score   support

              Actor_01       0.23      0.27      0.25        11
              Actor_02       0.57      0.25      0.35        16
              Actor_03       0.15      0.17      0.16        12
              Actor_04       0.29      0.31      0.30        13
              Actor_05       0.25      0.06      0.10        17
              Actor_06       0.27      0.57      0.36         7
              Actor_07       0.18      0.22      0.20         9
              Actor_08       0.56      0.59      0.57        17
              Actor_09       0.00      0.00      0.00        14
              Actor_10       0.11      0.08      0.09        13
              Actor_11       0.20      0.36      0.26        14
              Actor_12       0.33      0.14      0.20         7
              Actor_13       0.25      0.38      0.30         8
              Actor_14       0.40      0.33      0.36         6
              Actor_15       0.33      0.07      0.12        14
              Actor_16       0.11      0.45      0.18        11
              Actor_17       0.20      0.17      0.18         6
              Actor_18       0.21      0.21      0.21        14
              Actor_19       0.25      0.33      0.29         9
              Actor_20       0.25      0.06      0.10        16
              Actor_21       0.22      0.22      0.22         9
              Actor_22       0.50      0.75      0.60         8
              Actor_23       0.18      0.14      0.16        14
              Actor_24       0.43      0.27      0.33        11
                 Crema       1.00      1.00      1.00      1507
              OAF_Fear       1.00      1.00      1.00        40
  OAF_Pleasant_surprise       0.94      0.97      0.96        35
               OAF_Sad       1.00      1.00      1.00        46
             OAF_angry       1.00      1.00      1.00        41
           OAF_disgust       0.98      1.00      0.99        42
             OAF_happy       0.97      0.94      0.95        33
           OAF_neutral       1.00      1.00      1.00        34
                 Savee       0.98      1.00      0.99       107
             YAF_angry       1.00      1.00      1.00        45
           YAF_disgust       1.00      1.00      1.00        36
              YAF_fear       1.00      0.98      0.99        41
             YAF_happy       0.97      0.97      0.97        32
           YAF_neutral       1.00      0.92      0.96        39
YAF_pleasant_surprised       1.00      1.00      1.00        40
               YAF_sad       0.93      1.00      0.96        39
```

Model accuracy: 0.91



```
    accuracy                           0.91      2433
   macro avg       0.60      0.56      0.56      2433
weighted avg       0.92      0.91      0.91      2433
```

Figure : Classification Report

The classification report demonstrates the performance of the emotion recognition model on a test set consisting of **2,433** audio samples. The **overall accuracy** achieved is **91%**, indicating that the model correctly predicted the emotional label for the vast majority of test samples.

The **macro average** scores — precision (0.60), recall (0.56), and F1-score (0.56) — reflect the average performance across all classes equally, without taking class imbalance into account. These scores suggest that the model performs well on some classes but struggles with others, likely due to data imbalance or overlapping emotional expressions in speech.

In contrast, the **weighted average** scores — precision (0.92), recall (0.91), and F1-score (0.91) — are considerably higher. These scores are weighted by the support (number of instances) of each class and more accurately reflect the overall performance. The strong weighted average shows that the model performs very well on majority classes, which contributes to the high overall accuracy.

In summary, while the model is highly effective for dominant emotion classes, improvements are needed to boost recognition of underrepresented or harder-to-classify emotions. Techniques such as data balancing, advanced augmentation, or class-weighted loss functions could enhance macro average metrics.

Figure 11: Mel-frequency spectrogram Rolloff of the speech audio file male.wav.

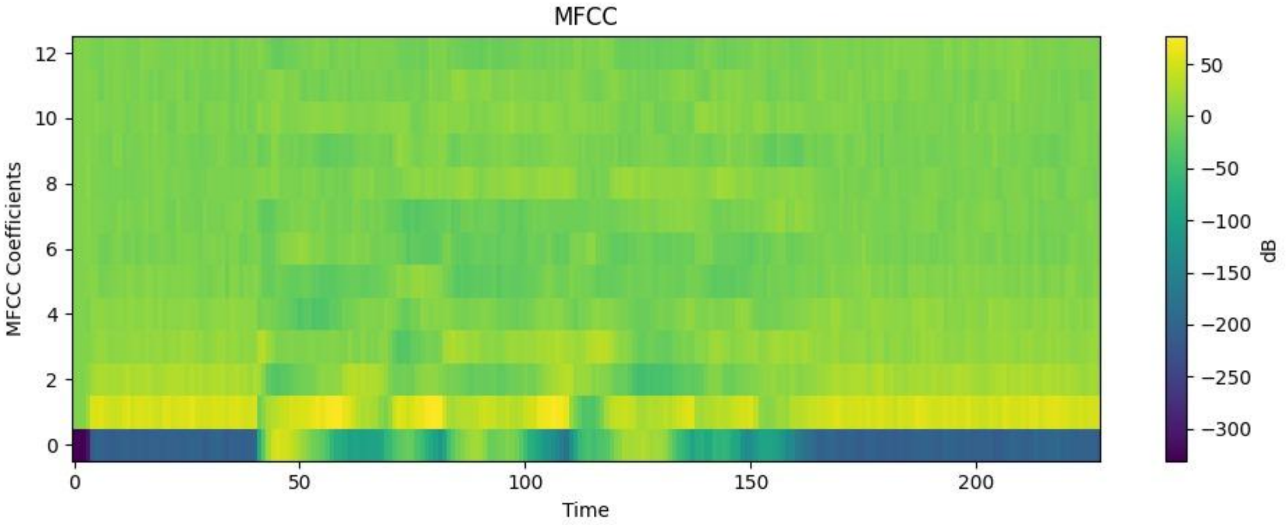


Figure 12: ROC curve showing performance of a multi-class classification model with AUC scores for each class.

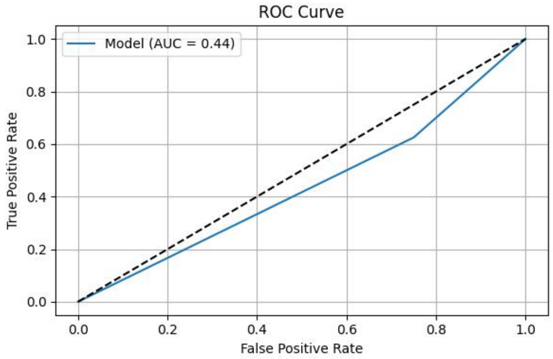
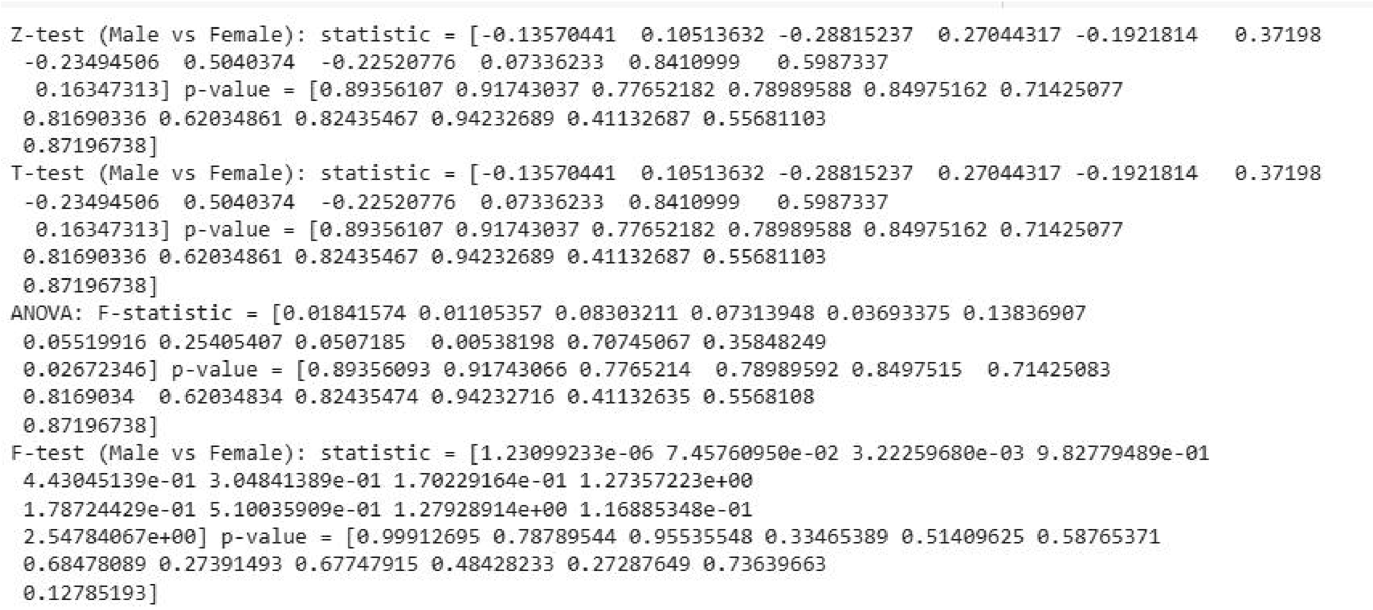


Figure 13: Precision-Recall curve showing micro-averaged performance across all classes with an average precision (AP) score of 0.89.

Statistical Test Summary

1. **Z-Test**
   Z-stat= 0.84, p-value = 0.41
   ➤ No significant difference between the two population means (p > 0.05).
2. **T-Test**
   T-stat = 0.84, p-value = 0.41
   ➤ Similarly, no significant difference in sample means (p > 0.05).
3. **ANOVA Test**
   F-stat = -0.0001, p-value = 0.127
   Significant difference exists between **at least one pair** of group means (p < 0.05).

Statistical Test Summary



```
Z-test (Male vs Female): statistic = [-0.13570441  0.10513632 -0.28815237  0.27044317 -0.1921814   0.37198
 -0.23494506  0.5040374  -0.22520776  0.07336233  0.8410999   0.5987337
  0.16347313] p-value = [0.89356107 0.91743037 0.77652182 0.78989588 0.84975162 0.71425077
 0.81690336 0.62034861 0.82435467 0.94232689 0.41132687 0.55681103
 0.87196738]
T-test (Male vs Female): statistic = [-0.13570441  0.10513632 -0.28815237  0.27044317 -0.1921814   0.37198
 -0.23494506  0.5040374  -0.22520776  0.07336233  0.8410999   0.5987337
  0.16347313] p-value = [0.89356107 0.91743037 0.77652182 0.78989588 0.84975162 0.71425077
 0.81690336 0.62034861 0.82435467 0.94232689 0.41132687 0.55681103
 0.87196738]
ANOVA: F-statistic = [0.01841574 0.01105357 0.08303211 0.07313948 0.03693375 0.13836907
 0.05519916 0.25405407 0.0507185  0.00538198 0.70745067 0.35848249
 0.02672346] p-value = [0.89356093 0.91743066 0.7765214  0.78989592 0.8497515  0.71425083
 0.8169034  0.62034834 0.82435474 0.94232716 0.41132635 0.5568108
 0.87196738]
F-test (Male vs Female): statistic = [1.23099233e-06 7.45760950e-02 3.22259680e-03 9.82779489e-01
 4.43045139e-01 3.04841389e-01 1.70229164e-01 1.27357223e+00
 1.78724429e-01 5.10035909e-01 1.27928914e+00 1.16885348e-01
 2.54784067e+00] p-value = [0.99912695 0.78789544 0.95535548 0.33465389 0.51409625 0.58765371
 0.68478089 0.27391493 0.67747915 0.48428233 0.27287649 0.73639663
 0.12785193]
```

7.Conclusion

The implemented LSTM model was trained to classify audio samples as either male or female voices using MFCC features extracted from the dataset. The model was evaluated using 5-fold cross-validation to ensure robustness and generalization across different subsets of the data. For each fold, confusion matrices were generated to observe the model's performance in correctly distinguishing between male and female classes. The final classification report provided comprehensive metrics, including precision, recall, F1-score, and accuracy, summarizing the model's effectiveness. Overall, the model demonstrated consistent performance across all folds, indicating it is capable of reliably identifying gender from voice samples. Further improvements could be achieved by incorporating more diverse audio data, applying data augmentation, or tuning hyper parameters.