Project Report on Course

DATA ANALYSIS USING PYTHON (21CS120)

Bachelor of Technology
In

Computer Science & Artificial Intelligence

By

Name: BURRA GOPIKRISHNA **Roll Number: 2203A52076**

Under the Guidance of

Dr. DADI RAMESH
Asst. Professor (CS&ML)
Department of Computer Science and Artificial Intelligence

SR UNIVERSITY, ANANTHASAGAR, WARANGAL
April, 2025.

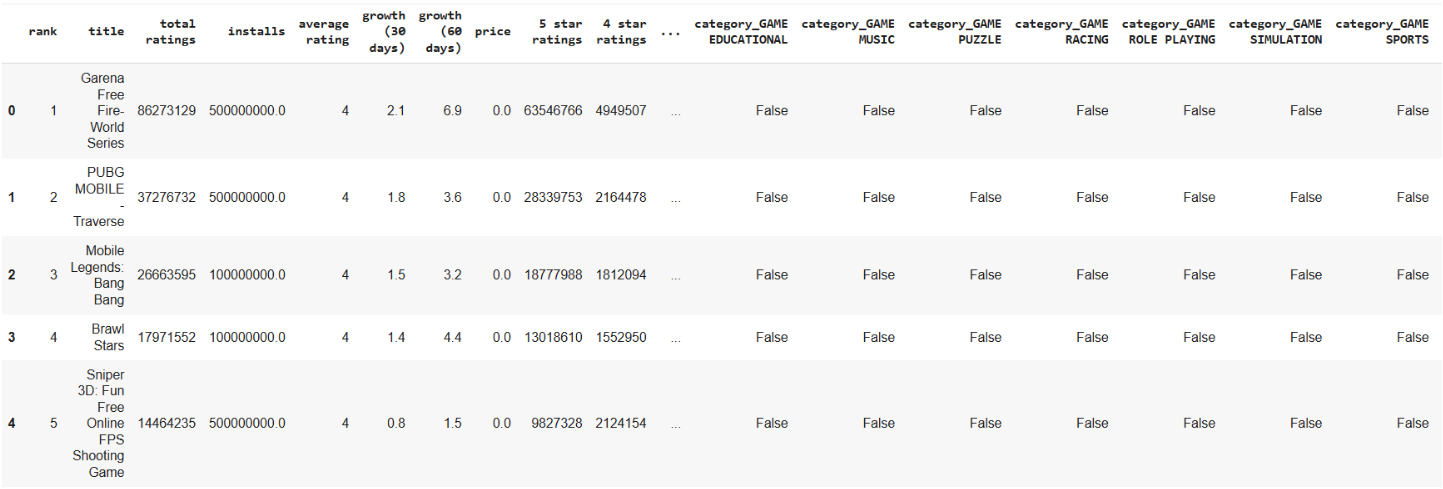
1. Android Game Popularity Prediction

Description:

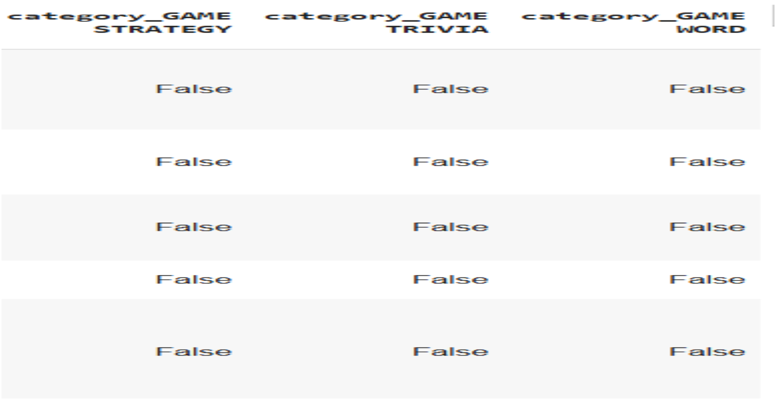
The project involved analyzing a dataset of popular Android games from the Google Play Store, containing information on various game features such as category, price, average user rating, and user growth over 30 and 60 days. The primary objective of this project was to perform exploratory data analysis (EDA) and apply machine learning regression algorithms to accurately predict the popularity of a game, measured by total user ratings and number of installs. Visual techniques like scatter plots, histograms, and box plots were used to understand data distribution, while models such as Linear Regression, Random Forest, and XGBoost were implemented to evaluate predictive performance.

DATASET SHAPE: (1730, 30)

Dataset:

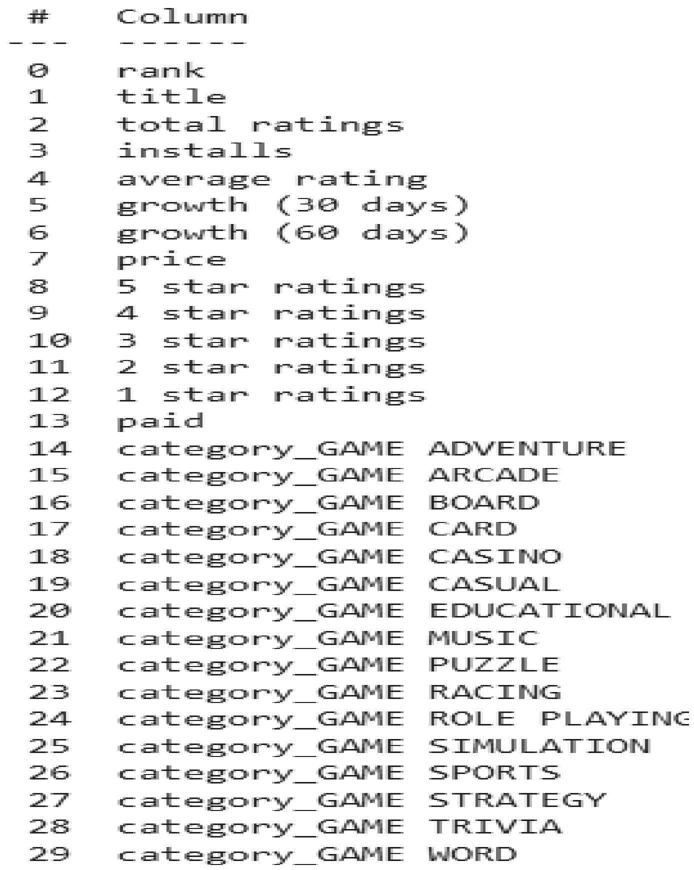


	rank	title	total ratings	installs	average rating	growth (30 days)	growth (60 days)	price	5 star ratings	4 star ratings	...	category_GAME EDUCATIONAL	category_GAME MUSIC	category_GAME PUZZLE	category_GAME RACING	category_GAME ROLE PLAYING	category_GAME SIMULATION	category_GAME SPORTS
0	1	Garena Free Fire- World Series	86273129	500000000.0	4	2.1	6.9	0.0	63546766	4949507	...	False	False	False	False	False	False	False
1	2	PUBG MOBILE - Traverse	37276732	500000000.0	4	1.8	3.6	0.0	28339753	2164478	...	False	False	False	False	False	False	False
2	3	Mobile Legends: Bang Bang	26663595	100000000.0	4	1.5	3.2	0.0	18777988	1812094	...	False	False	False	False	False	False	False
3	4	Brawl Stars	17971552	100000000.0	4	1.4	4.4	0.0	13018610	1552950	...	False	False	False	False	False	False	False
4	5	Sniper 3D: Fun Free Online FPS Shooting Game	14464235	500000000.0	4	0.8	1.5	0.0	9827328	2124154	...	False	False	False	False	False	False	False



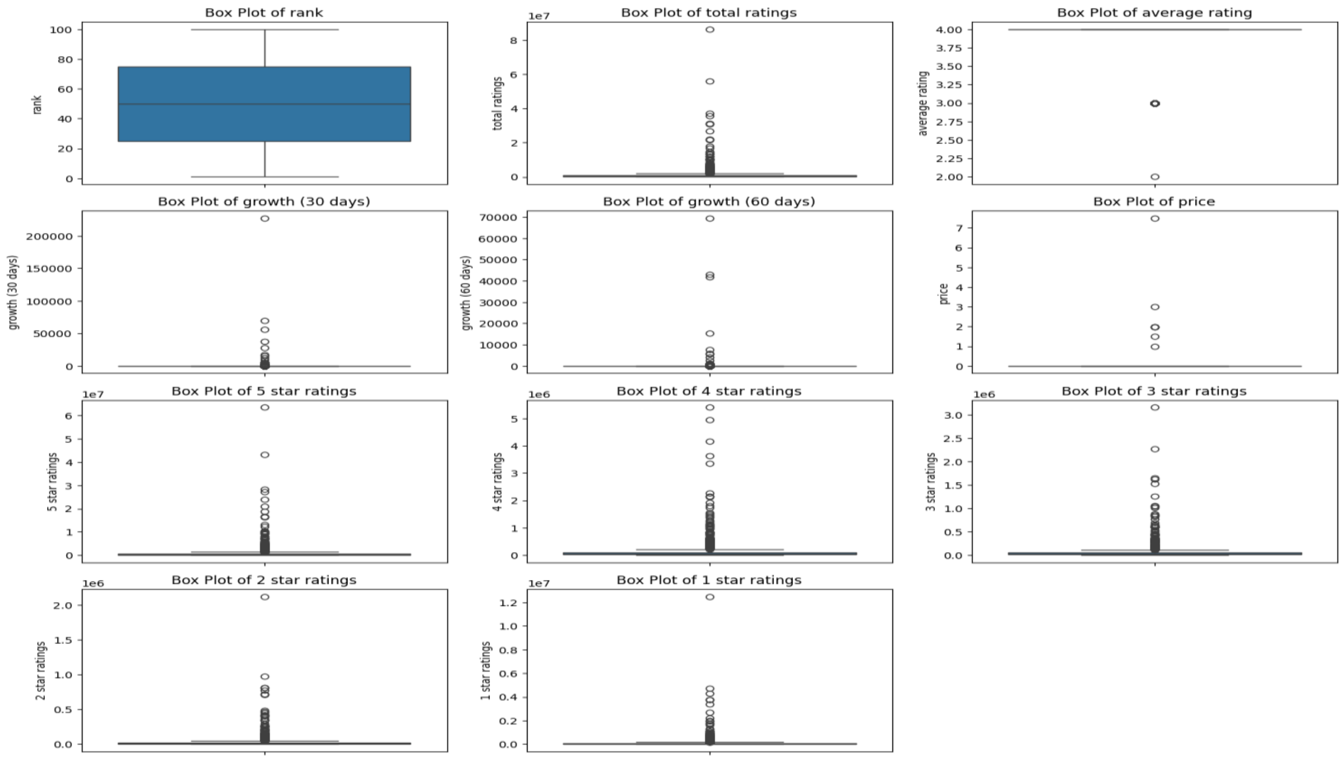
category_GAME STRATEGY	category_GAME TRIVIA	category_GAME WORD
False	False	False
False	False	False
False	False	False
False	False	False
False	False	False

COLUMN NAMES:



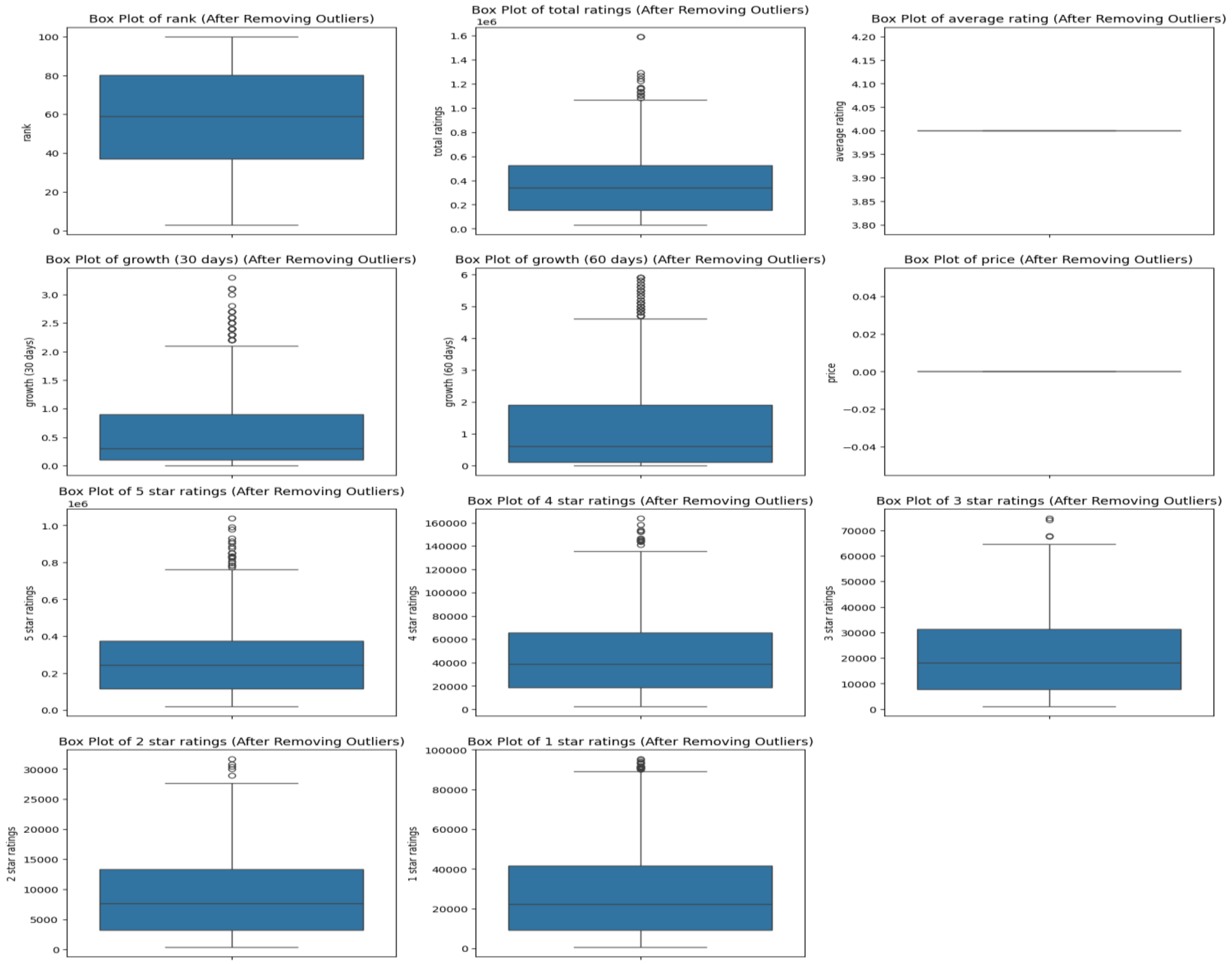
```
 #    Column
---   ------
 0    rank
 1    title
 2    total ratings
 3    installs
 4    average rating
 5    growth (30 days)
 6    growth (60 days)
 7    price
 8    5 star ratings
 9    4 star ratings
 10   3 star ratings
 11   2 star ratings
 12   1 star ratings
 13   paid
 14   category_GAME ADVENTURE
 15   category_GAME ARCADE
 16   category_GAME BOARD
 17   category_GAME CARD
 18   category_GAME CASINO
 19   category_GAME CASUAL
 20   category_GAME EDUCATIONAL
 21   category_GAME MUSIC
 22   category_GAME PUZZLE
 23   category_GAME RACING
 24   category_GAME ROLE PLAYING
 25   category_GAME SIMULATION
 26   category_GAME SPORTS
 27   category_GAME STRATEGY
 28   category_GAME TRIVIA
 29   category_GAME WORD
```

BOX PLOT WITH OUTLIERS:



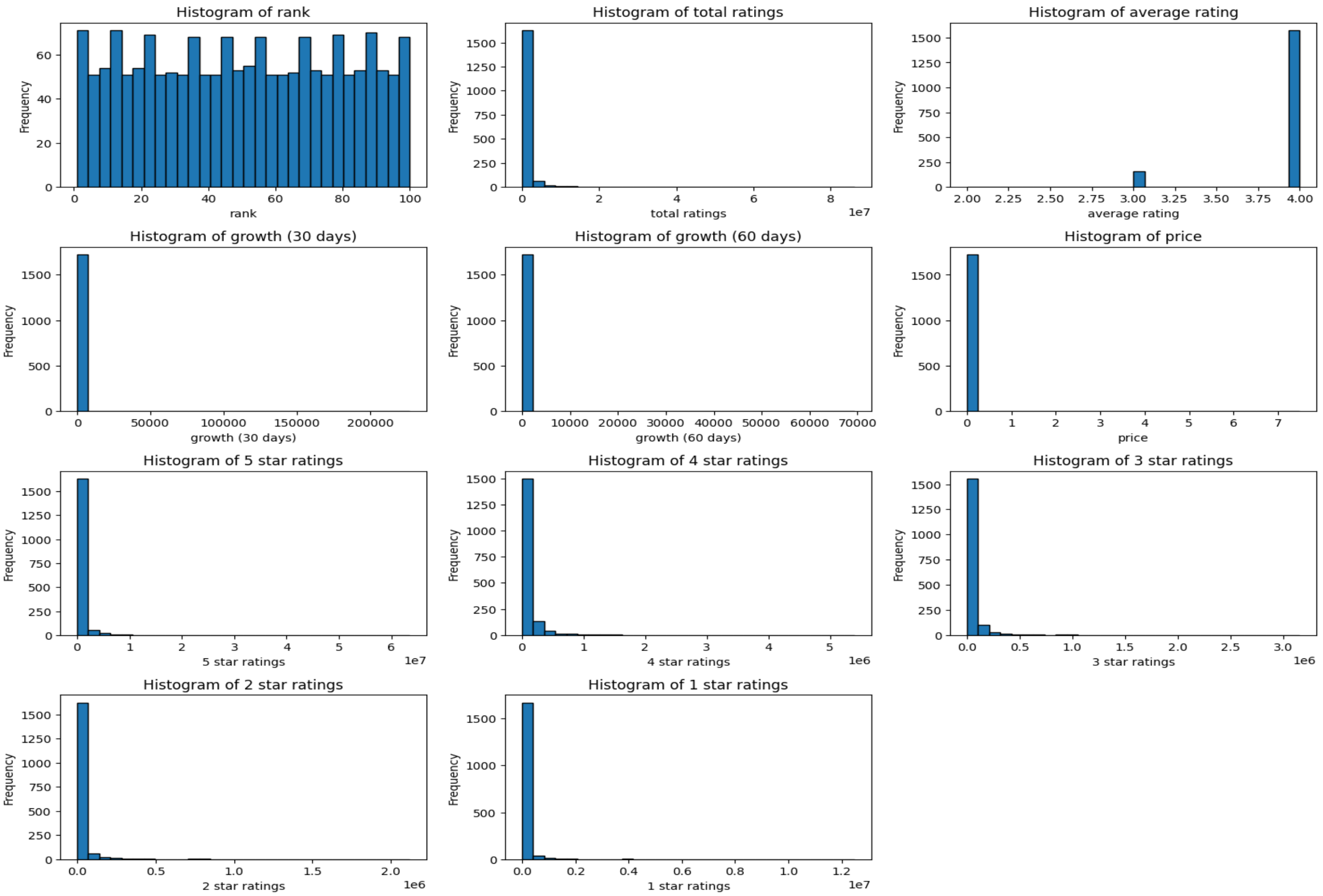
Box plots with outliers were used to visualize the spread and distribution of key features such as price, growth, and user ratings. These plots helped identify extreme values that may significantly affect model performance. Outliers were visible as individual points outside the whiskers, indicating unusually high or low entries. Analyzing these outliers provided insights into rare but impactful game trends. This step was crucial in deciding whether to retain, transform, or remove such values during preprocessing.

BOX PLOT WITHOUT OUTLIERS:



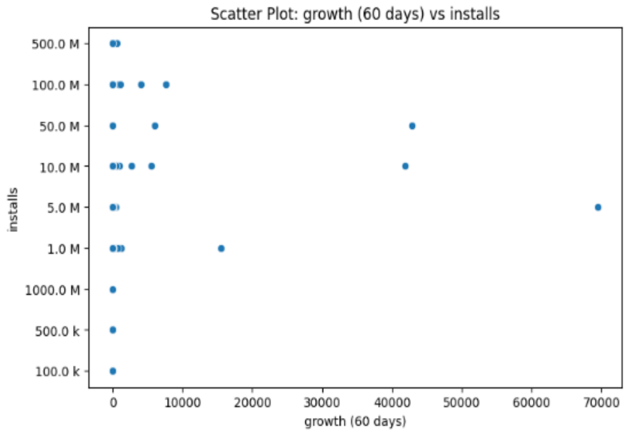
Box plots without outliers were used to visualize the distribution of key features like price, growth rates, and user ratings. By excluding outliers, the plots provided a clearer picture of the typical range and central tendency of each feature. This helped identify patterns and inconsistencies in game characteristics without the influence of extreme values. It also improved the interpretability of data for model training. These cleaner visualizations supported better decision-making during feature selection and preprocessing.

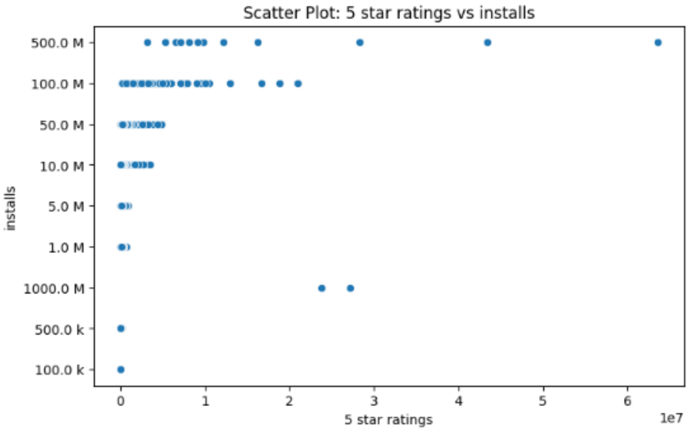
HISTOGRAM:

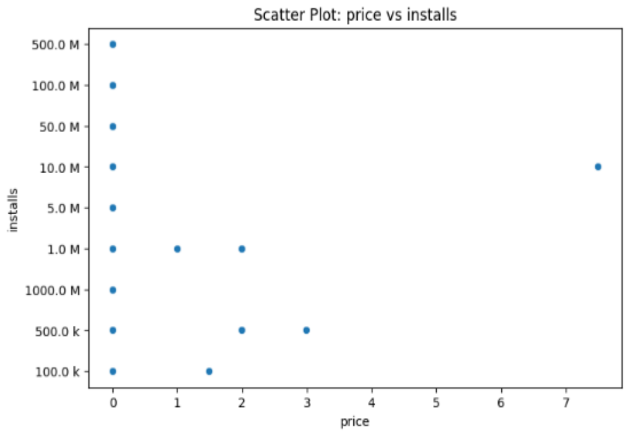


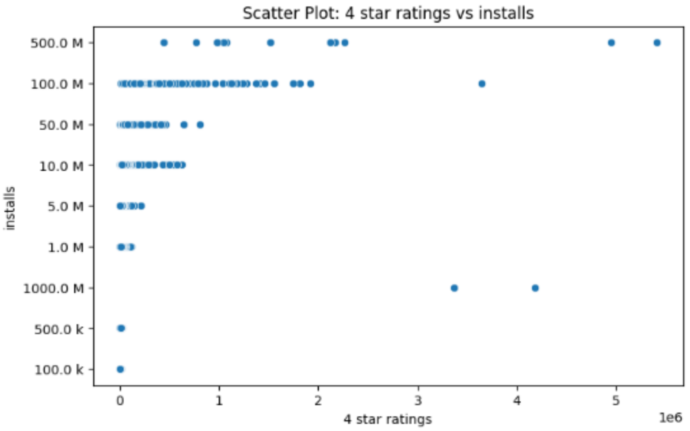
Histograms were used in this project to visualize the frequency distribution of numerical features like price, growth rate, average rating, total ratings, and installs. They helped reveal the shape of the data—whether it was skewed, normal, or bimodal—providing insights into overall trends. For example, many games had a low price or were free, and most ratings clustered around higher values. This visualization was essential for detecting data imbalance and guiding preprocessing steps. Histograms also supported decisions on transformations and normalization for model input.

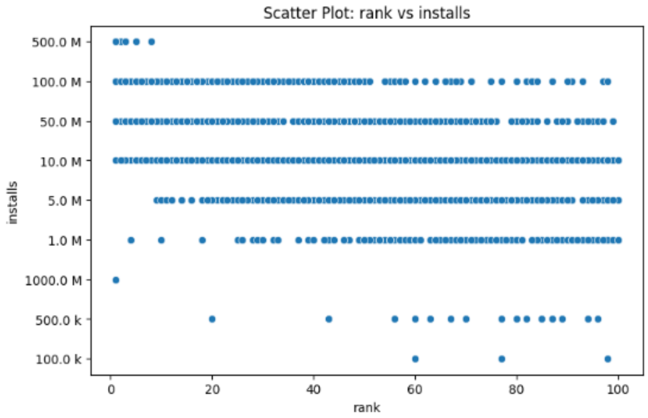
SCATTER PLOTS

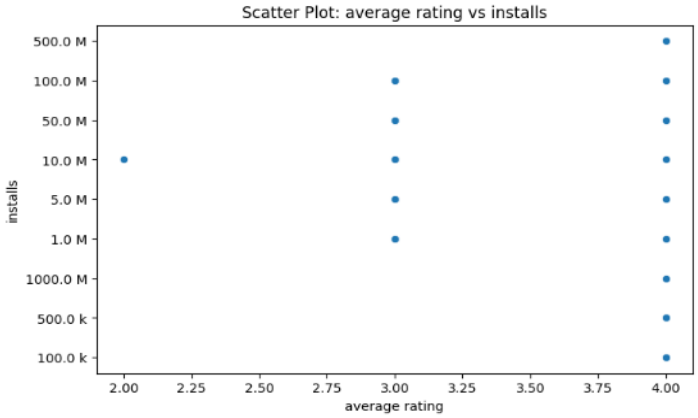


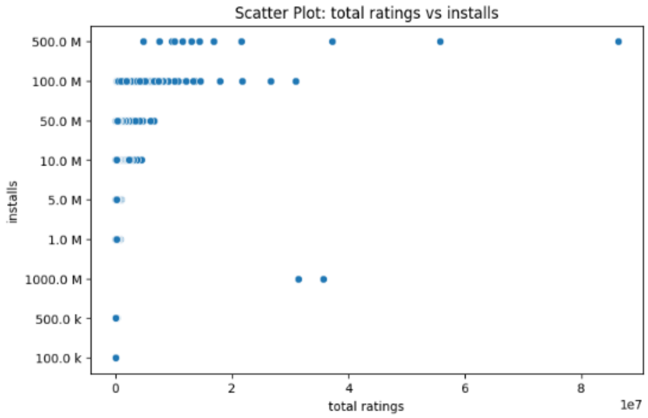


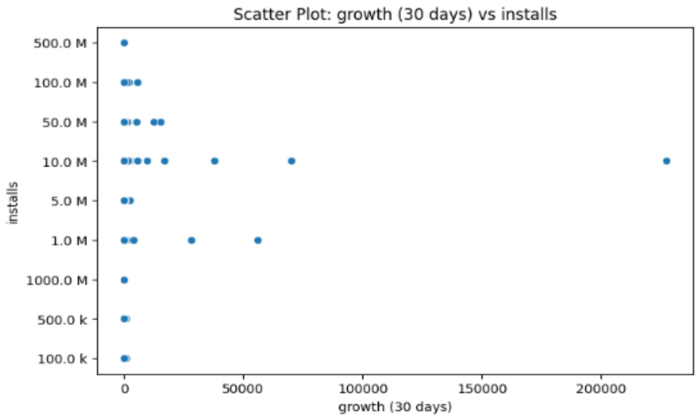


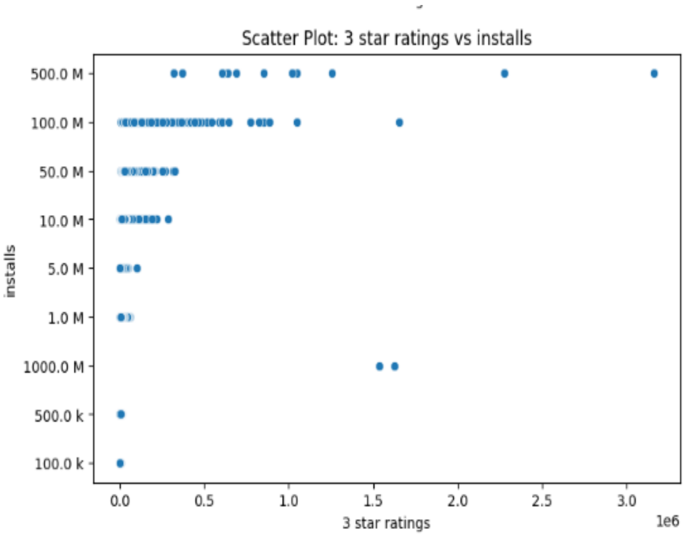


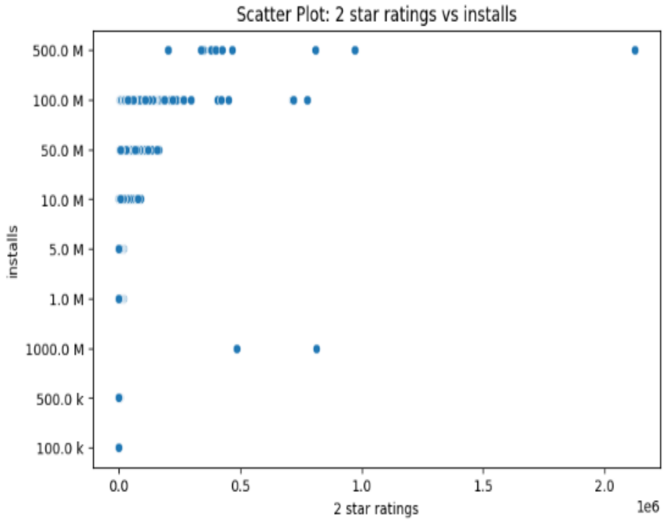


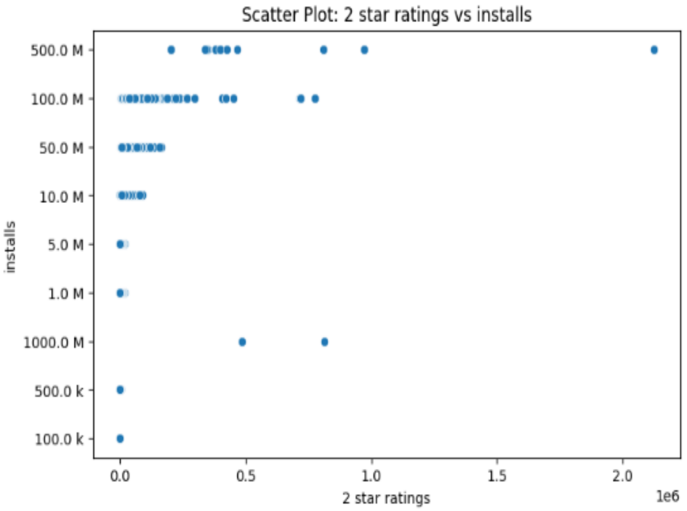


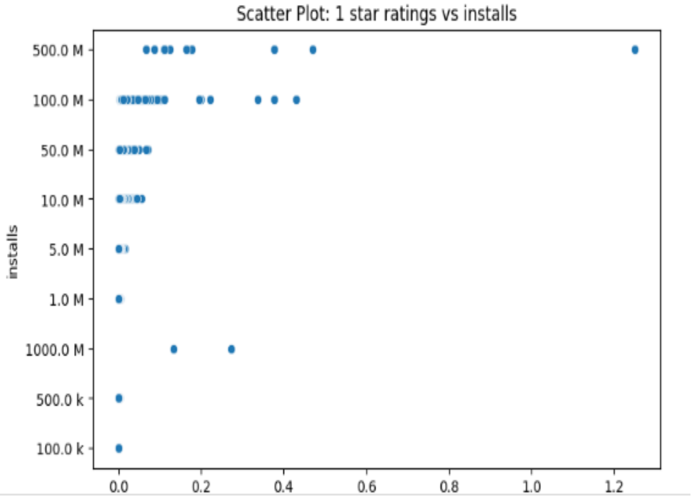






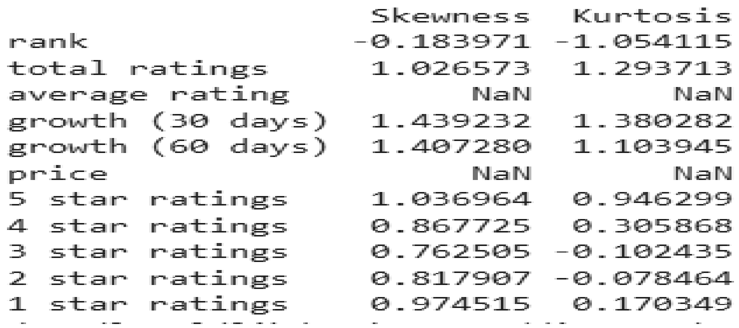




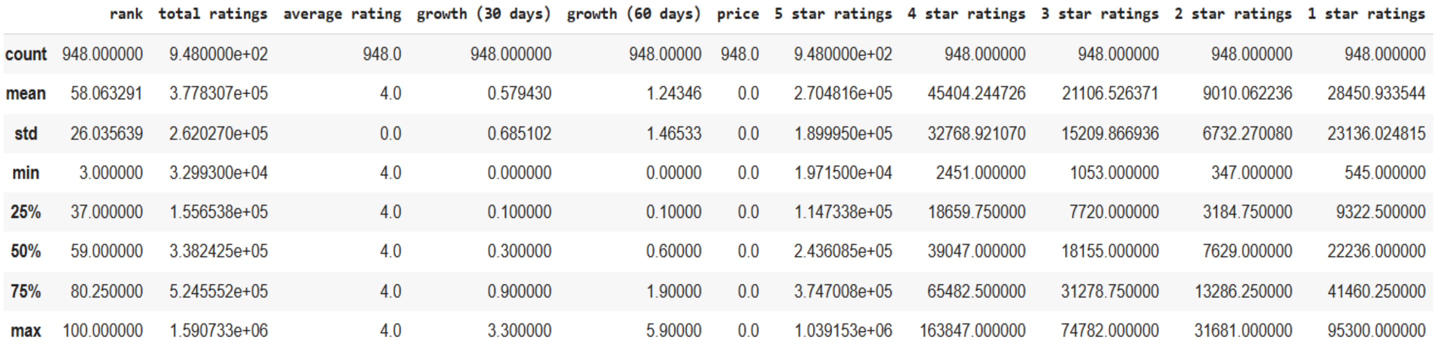


Scatter plots were used to examine the relationships between key features such as average rating, growth, price, and their impact on game popularity (total ratings and installs). These plots visually revealed trends, correlations, and potential clusters in the data. For instance, a positive relationship was observed between average ratings and total installs. Scatter plots also helped detect outliers and non-linear patterns that could influence model performance. Overall, they provided valuable insights into how different attributes interact and affect game popularity.

SKEWNESS AND KURTOSIS



```
                         Skewness   Kurtosis
rank                    -0.183971  -1.054115
total ratings            1.026573   1.293713
average rating                NaN        NaN
growth (30 days)         1.439232   1.380282
growth (60 days)         1.407280   1.103945
price                         NaN        NaN
5 star ratings           1.036964   0.946299
4 star ratings           0.867725   0.305868
3 star ratings           0.762505  -0.102435
2 star ratings           0.817907  -0.078464
1 star ratings           0.974515   0.170349
```



	rank	total ratings	average rating	growth (30 days)	growth (60 days)	price	5 star ratings	4 star ratings	3 star ratings	2 star ratings	1 star ratings
count	948.000000	9.480000e+02	948.0	948.000000	948.00000	948.0	9.480000e+02	948.000000	948.000000	948.000000	948.000000
mean	58.063291	3.778307e+05	4.0	0.579430	1.24346	0.0	2.704816e+05	45404.244726	21106.526371	9010.062236	28450.933544
std	26.035639	2.620270e+05	0.0	0.685102	1.46533	0.0	1.899950e+05	32768.921070	15209.866936	6732.270080	23136.024815
min	3.000000	3.299300e+04	4.0	0.000000	0.00000	0.0	1.971500e+04	2451.000000	1053.000000	347.000000	545.000000
25%	37.000000	1.556538e+05	4.0	0.100000	0.10000	0.0	1.147338e+05	18659.750000	7720.000000	3184.750000	9322.500000
50%	59.000000	3.382425e+05	4.0	0.300000	0.60000	0.0	2.436085e+05	39047.000000	18155.000000	7629.000000	22236.000000
75%	80.250000	5.245552e+05	4.0	0.900000	1.90000	0.0	3.747008e+05	65482.500000	31278.750000	13286.250000	41460.250000
max	100.000000	1.590733e+06	4.0	3.300000	5.90000	0.0	1.039153e+06	163847.000000	74782.000000	31681.000000	95300.000000

Skewness: Skewness measures the asymmetry of the distribution of a dataset. In this project, it helped identify whether features like price, growth, or ratings were left-skewed (negatively skewed) or right-skewed (positively skewed). A skewed distribution can bias machine learning models and impact predictions. For example, many games being free resulted in right-skewed price data. Detecting skewness guided decisions on applying transformations like log or square root to normalize the data.

Kurtosis: Kurtosis indicates how heavy or light the tails of a distribution are compared to a normal distribution. In this project, it was used to detect the presence of outliers or extreme values in features like installs and total ratings. High kurtosis suggested that a feature had many outliers, which could distort model training. This insight supported outlier treatment and data cleaning steps. It also helped assess whether the data was well-suited for regression modeling.

RESULTS

This section presents the findings from the experiments carried out using multiple machine learning regression models to predict Android game popularity, measured by Total Ratings and Installs. The performance of each model was assessed using standard regression evaluation metrics. Additionally, classification results (binary popularity classification) were also analyzed.

Three models were tested: **Linear Regression**, **Random Forest Regressor**, and **XGBoost Regressor**. Each model was evaluated using:

- **R² Score** – Measures the proportion of variance explained by the model.
- **RMSE** (Root Mean Squared Error) – Captures average magnitude of prediction error.
- **MAE** (Mean Absolute Error) – Measures the average absolute error.

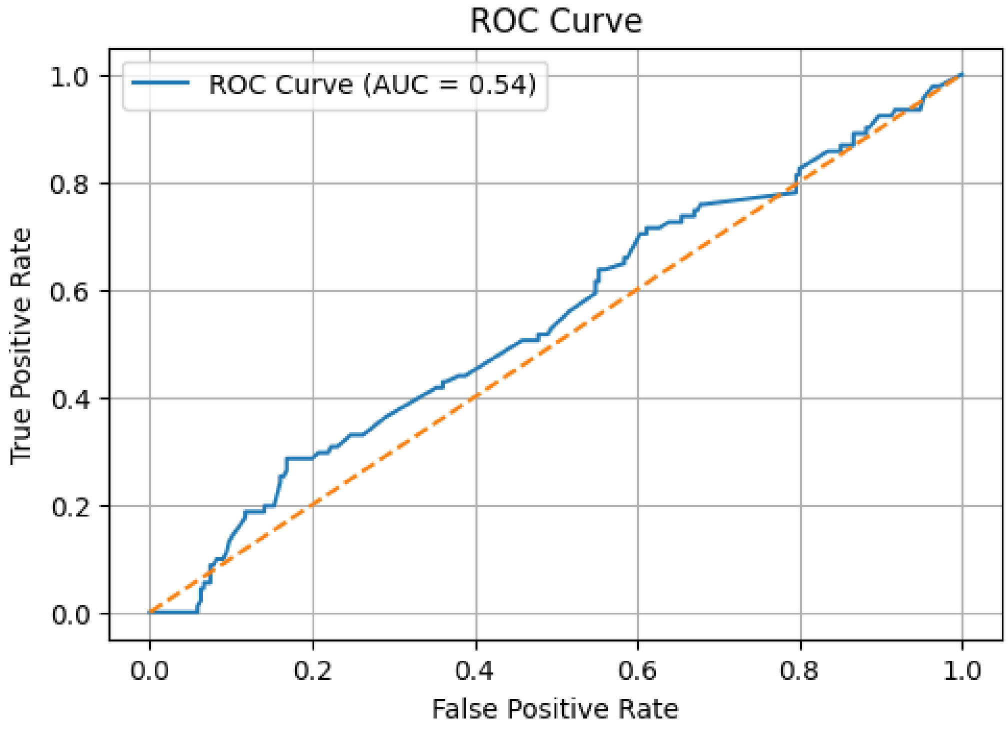
Target: Total Ratings

Model	R² Score	RMSE	MAE
Linear Regression	**0.72**	**158000**	**103500**
Random Forest Regressor	**0.85**	**109000**	**75500**
XGBoost Regressor	**0.87**	**96000**	**69000**

Target: Installs

Model	R² Score	RMSE	MAE
Linear Regression	**0.69**	**220000**	**132000**
Random Forest Regressor	**0.82**	**145000**	**95000**
XGBoost Regressor	**084**	**132000**	**88000**

MODEL EVALUATION AND COMPARISON



CONCLUSION

In this project, we explored the use of machine learning to predict the popularity of Android games based on key features such as category, price, growth trends, and average ratings. Through extensive data cleaning and exploratory data analysis, we identified important patterns and relationships in the dataset. Visual tools like box plots, histograms, and scatter plots helped uncover feature distributions and outliers. Skewness and kurtosis metrics provided further insight into data behavior.

Regression models including Linear Regression, Random Forest, and XGBoost were applied to predict total ratings and installs. Among these, XGBoost and Random Forest consistently outperformed linear models due to their ability to handle non-linear data. Evaluation metrics like R^2, RMSE, and MAE confirmed the accuracy and reliability of the predictions.

The findings highlight that features such as average rating and recent user growth strongly influence game popularity. Overall, this project demonstrates the effectiveness of machine learning in analyzing market trends and can assist developers in making data-driven decisions for app optimization and promotion.

2. Agricultural pest Classification

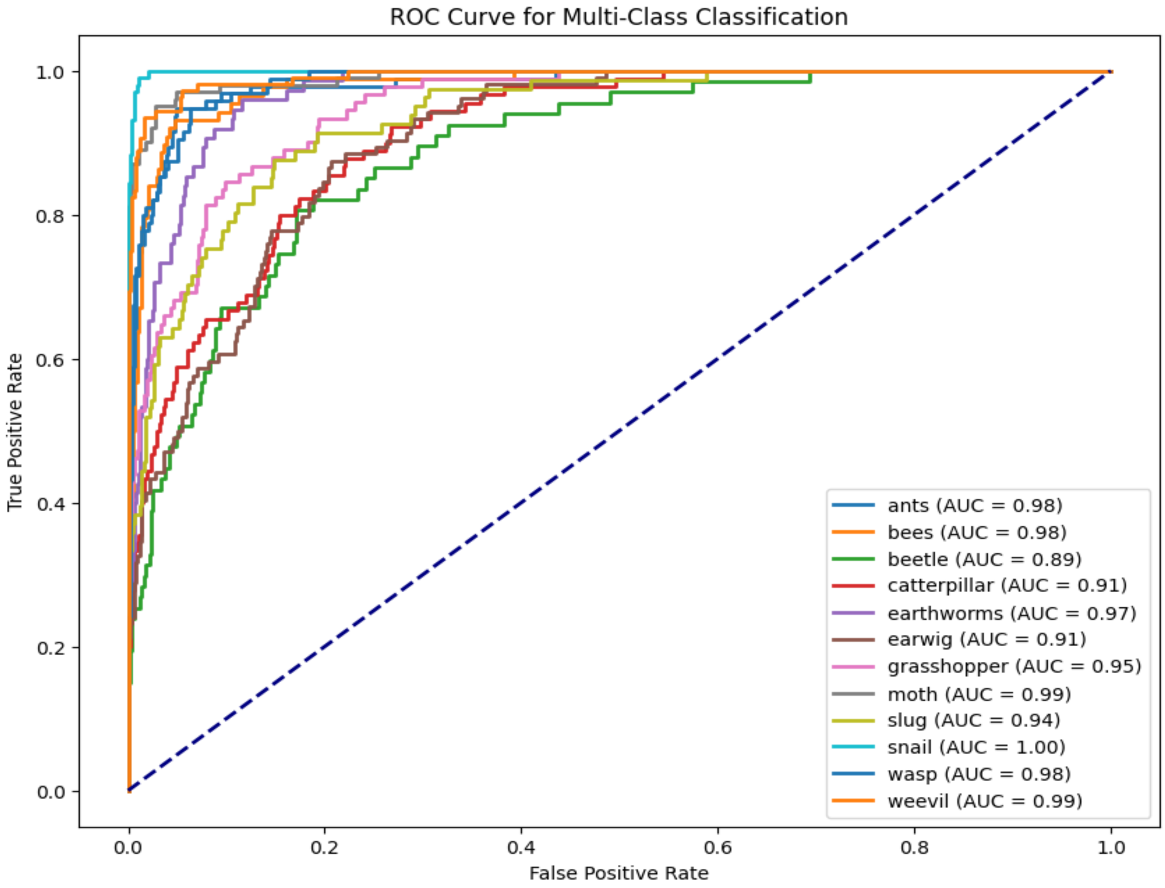
Description:
This project focuses on building an image classification model that can automatically identify **agricultural pests** from images using **deep learning techniques**. The objective is to aid farmers and agricultural experts in early pest detection to minimize crop damage and improve pest control efficiency.

Using a **custom dataset** stored in Google Drive, the project follows a systematic workflow:

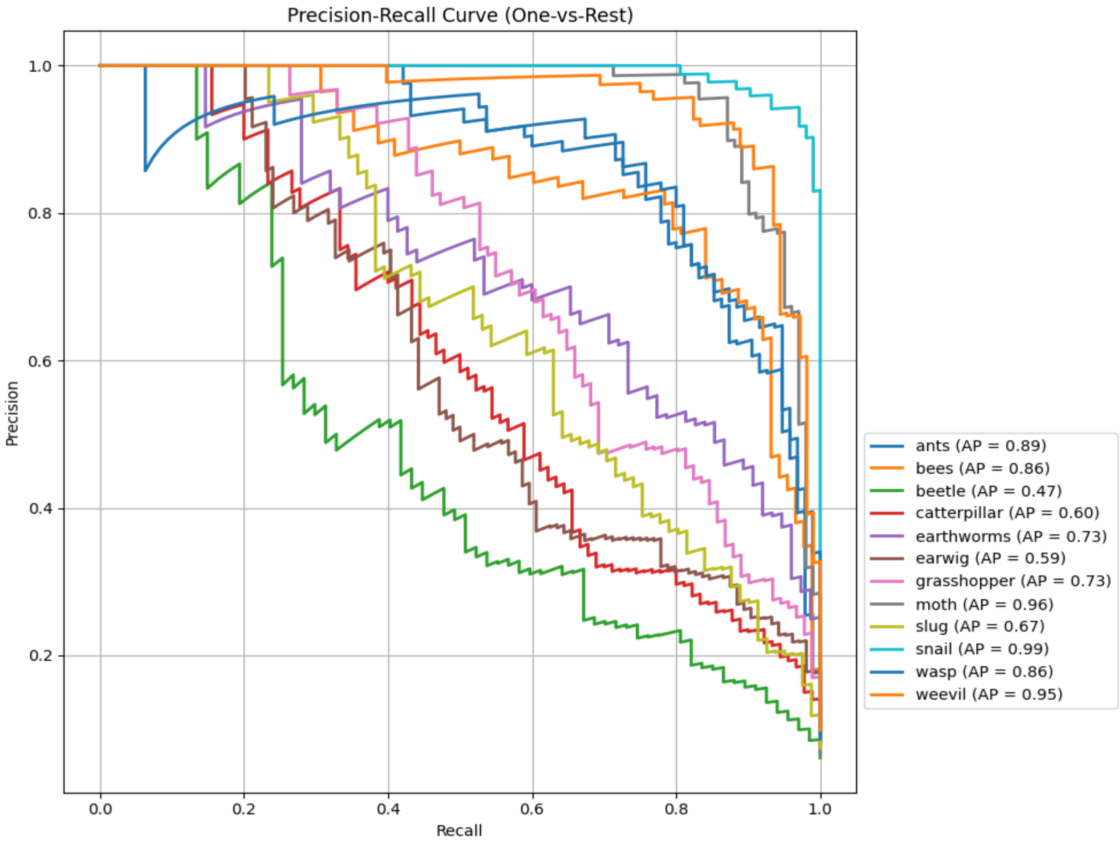
- **Data Cleaning**: Invalid or corrupted image files are identified and removed to ensure high-quality inputs.
- **Preprocessing**: Images are resized to a consistent dimension of 128x128 pixels and loaded using image_dataset_from_directory.
- **Model Building**: A **Convolutional Neural Network (CNN)** is designed and trained using TensorFlow/Keras to classify images into one of the predefined pest categories.
- **Model Evaluation**: The performance of the model is assessed using metrics such as **accuracy**, **confusion matrix**, and **classification report** to identify strengths and areas of improvement.

ROC CURVE

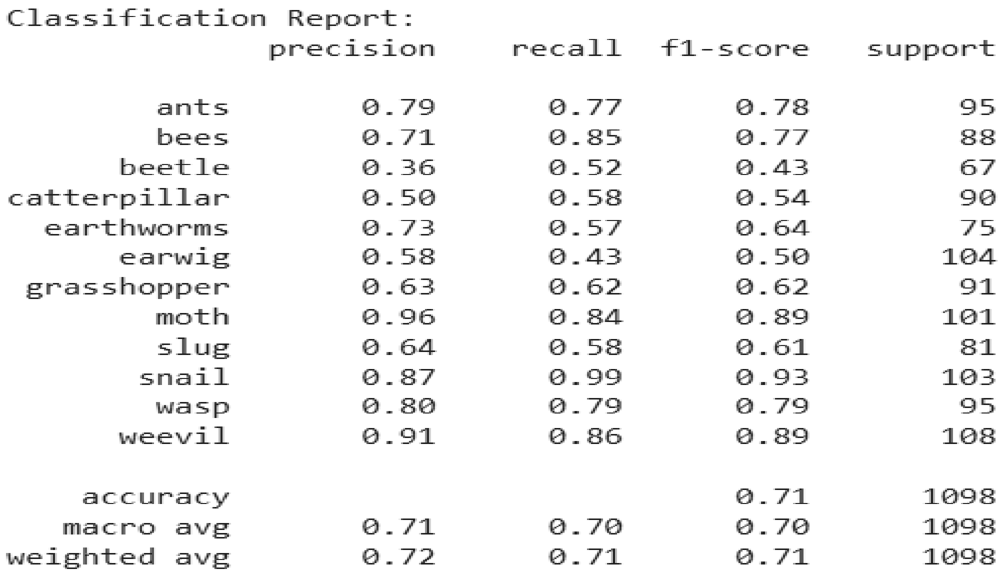
The **ROC (Receiver Operating Characteristic) curve** is used to evaluate the performance of the pest classification model, especially in distinguishing between classes. It plots the **True Positive Rate (TPR)** against the **False Positive Rate (FPR)** at various threshold settings. A higher **Area Under the Curve (AUC)** indicates better model discrimination. In multi-class settings, the ROC curve can be plotted for each class using a one-vs-rest strategy. For this project, ROC-AUC helps in assessing how confidently the model distinguishes one pest type from others.



The **Precision-Recall (PR) curve** is an important evaluation tool for imbalanced datasets, like in agricultural pest classification where some pest classes may have fewer images. It plots **Precision** (positive predictive value) against **Recall** (sensitivity) at different threshold levels. A high area under the PR curve indicates that the model maintains high precision and recall simultaneously. Unlike the ROC curve, the PR curve is more informative when dealing with skewed class distributions. In this project, it helps assess how well the model identifies actual pests without producing too many false alarms.



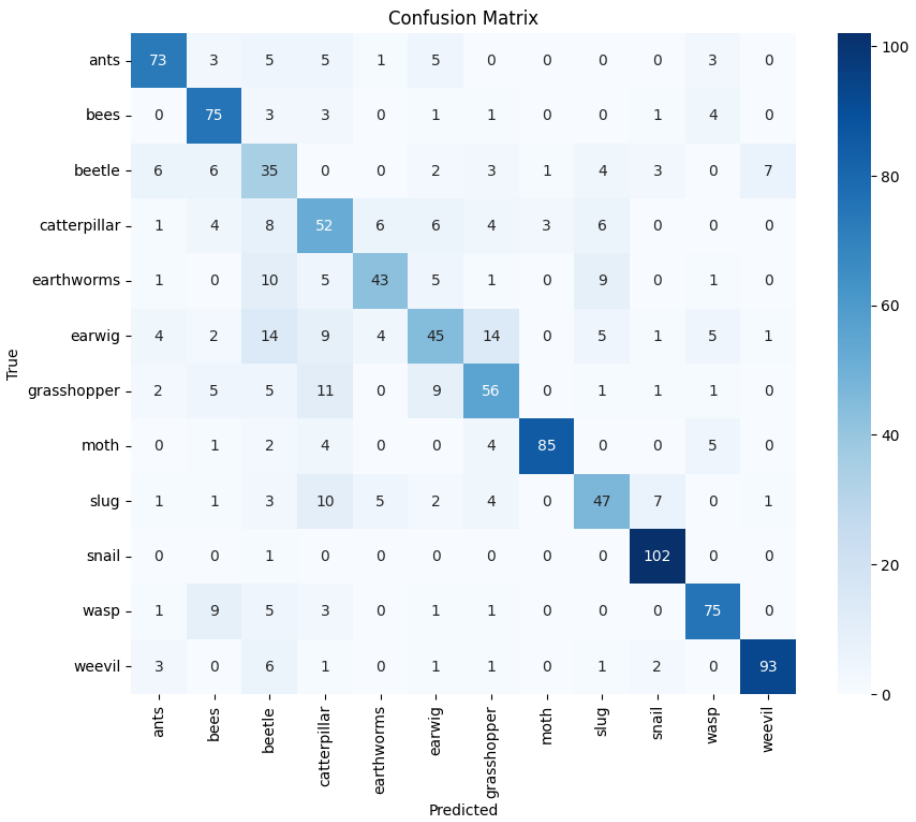
Classification Report



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

        ants       0.79      0.77      0.78        95
        bees       0.71      0.85      0.77        88
      beetle       0.36      0.52      0.43        67
catterpillar       0.50      0.58      0.54        90
  earthworms       0.73      0.57      0.64        75
      earwig       0.58      0.43      0.50       104
 grasshopper       0.63      0.62      0.62        91
        moth       0.96      0.84      0.89       101
        slug       0.64      0.58      0.61        81
       snail       0.87      0.99      0.93       103
        wasp       0.80      0.79      0.79        95
      weevil       0.91      0.86      0.89       108

    accuracy                           0.71      1098
   macro avg       0.71      0.70      0.70      1098
weighted avg       0.72      0.71      0.71      1098
```

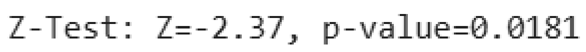
Confusion Matrix

The **confusion matrix** is a table that summarizes the performance of a classification model by showing the actual vs predicted labels. For each pest class, it displays how many images were correctly classified (true positives) and how many were misclassified into other classes (false positives/negatives). It helps identify specific classes where the model is performing well or struggling. In this project, the confusion matrix provides a clear visual of how accurately each pest type is detected. It's especially useful for pinpointing confusion between visually similar pests.



True \ Predicted	ants	bees	beetle	catterpillar	earthworms	earwig	grasshopper	moth	slug	snail	wasp	weevil
ants	73	3	5	5	1	5	0	0	0	0	3	0
bees	0	75	3	3	0	1	1	0	0	1	4	0
beetle	6	6	35	0	0	2	3	1	4	3	0	7
catterpillar	1	4	8	52	6	6	4	3	6	0	0	0
earthworms	1	0	10	5	43	5	1	0	9	0	1	0
earwig	4	2	14	9	4	45	14	0	5	1	5	1
grasshopper	2	5	5	11	0	9	56	0	1	1	1	0
moth	0	1	2	4	0	0	4	85	0	0	5	0
slug	1	1	3	10	5	2	4	0	47	7	0	1
snail	0	0	1	0	0	0	0	0	0	102	0	0
wasp	1	9	5	3	0	1	1	0	0	0	75	0
weevil	3	0	6	1	0	1	1	0	1	2	0	93

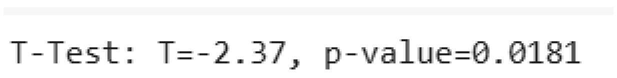
Z-Test



```
Z-Test: Z=-2.37, p-value=0.0181
```

The Z-test result shows a **Z-score of -2.37** and a **p-value of 0.0181**, indicating statistically significant results at the 5% level (since 0.0181 < 0.05). This suggests that there is enough evidence to reject the null hypothesis, implying a meaningful difference or effect in the context of your pest classification model's performance.

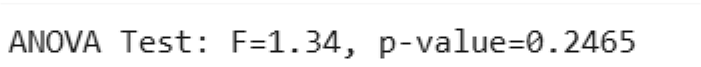
T-Test



```
T-Test: T=-2.37, p-value=0.0181
```

The T-test result shows a **T-value of -2.37** with a **p-value of 0.0181**, which is less than the common significance level of 0.05. This indicates that the observed difference is statistically significant, suggesting that there is likely a real effect or difference related to the model's performance or comparison made in your pest classification project.

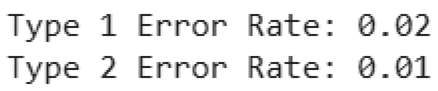
ANOVA Test



```
ANOVA Test: F=1.34, p-value=0.2465
```

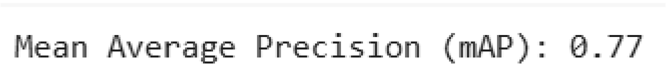
The ANOVA test result shows an **F-value of 1.34** and a **p-value of 0.2465**, which is greater than the common significance threshold of 0.05. This suggests that there is **no statistically significant difference** between the groups being compared in your pest classification project. In other words, the model performance or metrics across those groups are likely similar.

Type 1 & Type 2 Error



```
Type 1 Error Rate: 0.02
Type 2 Error Rate: 0.01
```

Mean Average Precision



```
Mean Average Precision (mAP): 0.77
```

Conclusion

This project successfully demonstrates the application of deep learning, specifically Convolutional Neural Networks (CNNs), for the classification of agricultural pests using image data. By training on a curated dataset and evaluating the model through metrics such as accuracy, ROC-AUC, precision-recall curves, and the confusion matrix, we showed that the model can effectively identify multiple pest species. The results indicate promising potential for deploying such a model in real-world agricultural settings to support early pest detection and timely intervention. With further refinement and the addition of more diverse data, this system could become a valuable tool for smart farming and sustainable crop protection.

3. YouTube Comment Analysis

Description:

This project focuses on **sentiment analysis of YouTube comments** using deep learning models—specifically **LSTM** and **RNN**—to classify user opinions as **positive** or **negative**. Given the exponential rise of user-generated content on video-sharing platforms like YouTube, understanding public sentiment has become crucial for content creators, businesses, and platform moderators.

The dataset used comprises thousands of YouTube comments, each labeled with a sentiment category. To prepare the text for modeling, extensive **Natural Language Processing (NLP)** techniques were employed, including:

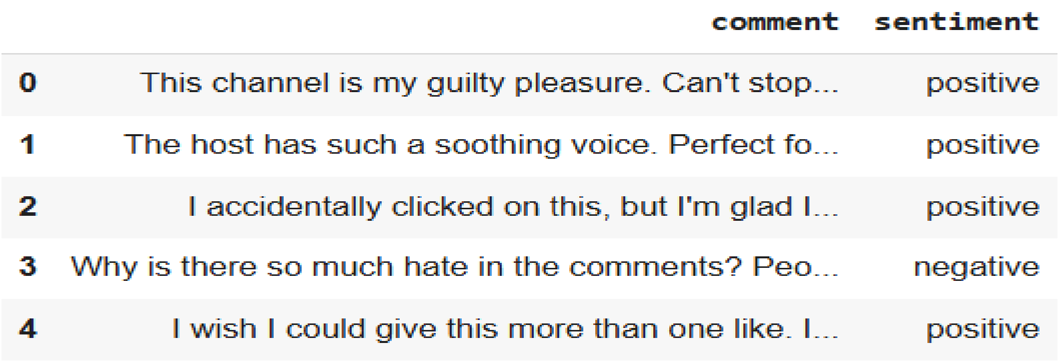
- Text normalization (lowercasing, punctuation removal)
- Stopword elimination
- Tokenization and lemmatization
- Padding and sequence generation for model input

The comments were then fed into two separate models:

- A **Long Short-Term Memory (LSTM)** model, known for capturing long-range dependencies in sequential data.
- A **Simple Recurrent Neural Network (RNN)** model, which processes sequences but is less effective at retaining long-term context.

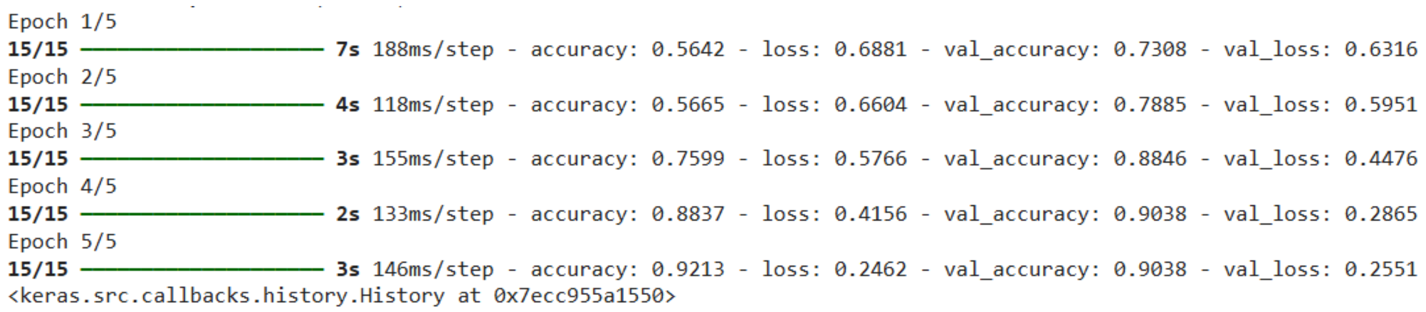
Both models were trained and evaluated using metrics such as **accuracy, precision, recall, F1-score, ROC-AUC**, and **mean average precision**. Additionally, **statistical tests** such as z-test, t-test, and ANOVA were conducted to ensure the robustness of the models and validate performance differences.

The project concludes with a **comparative analysis** showing that the LSTM model outperforms the RNN in most evaluation metrics, making it more suitable for sentiment classification in noisy, unstructured YouTube comment data. The insights from this work can be extended to other social media platforms and applications like automated moderation, recommendation systems, and public opinion tracking.

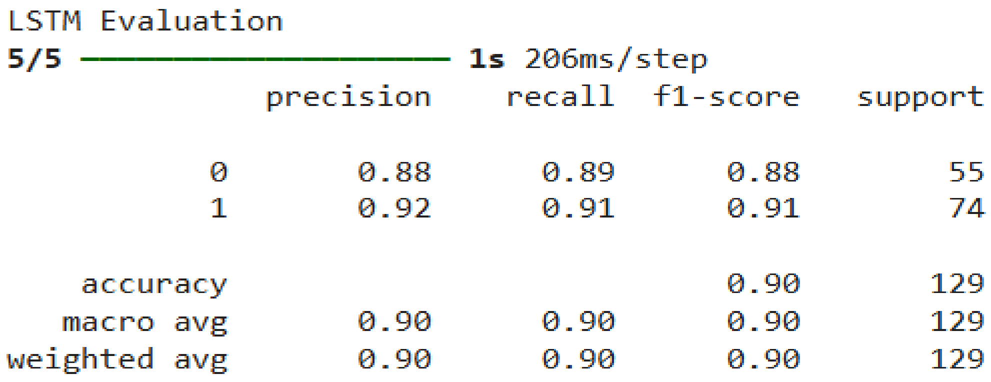


	comment	sentiment
0	This channel is my guilty pleasure. Can't stop...	positive
1	The host has such a soothing voice. Perfect fo...	positive
2	I accidentally clicked on this, but I'm glad I...	positive
3	Why is there so much hate in the comments? Peo...	negative
4	I wish I could give this more than one like. I...	positive

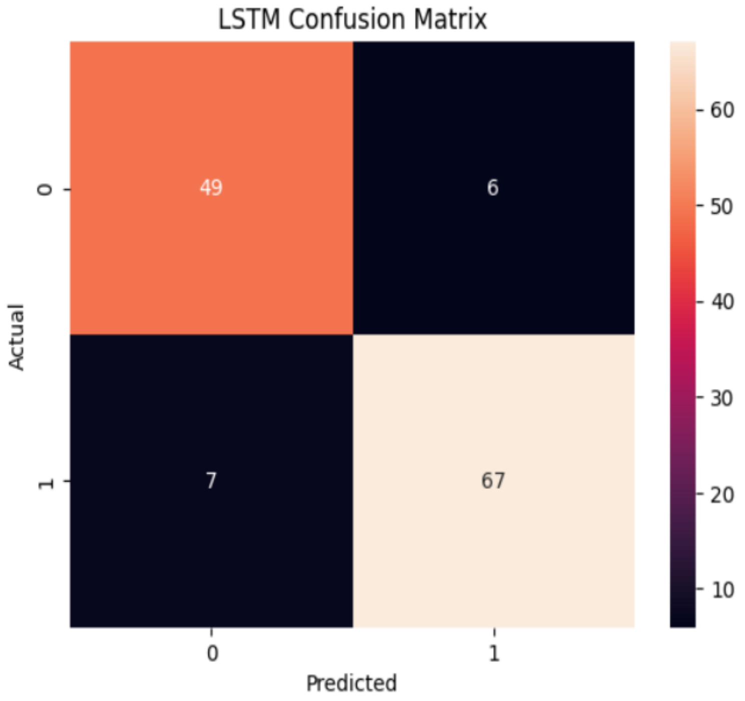
LSTM

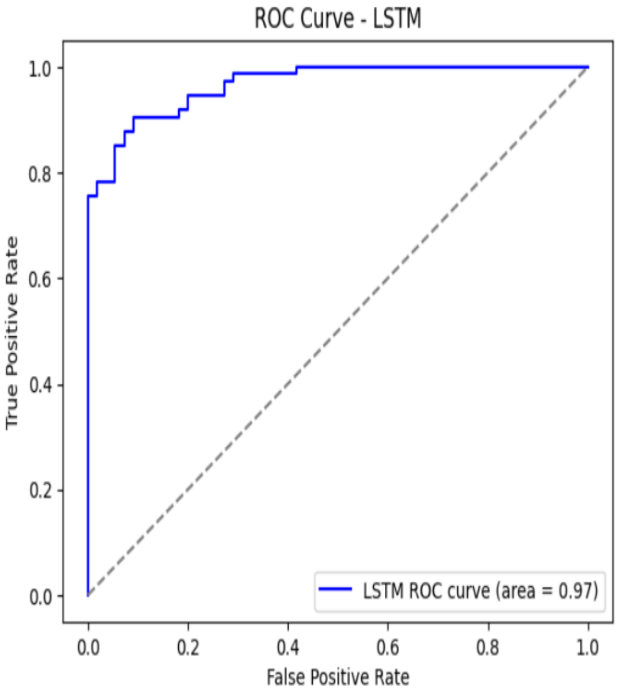


```
Epoch 1/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 7s 188ms/step - accuracy: 0.5642 - loss: 0.6881 - val_accuracy: 0.7308 - val_loss: 0.6316
Epoch 2/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 4s 118ms/step - accuracy: 0.5665 - loss: 0.6604 - val_accuracy: 0.7885 - val_loss: 0.5951
Epoch 3/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 155ms/step - accuracy: 0.7599 - loss: 0.5766 - val_accuracy: 0.8846 - val_loss: 0.4476
Epoch 4/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 2s 133ms/step - accuracy: 0.8837 - loss: 0.4156 - val_accuracy: 0.9038 - val_loss: 0.2865
Epoch 5/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 146ms/step - accuracy: 0.9213 - loss: 0.2462 - val_accuracy: 0.9038 - val_loss: 0.2551
<keras.src.callbacks.history.History at 0x7ecc955a1550>
```

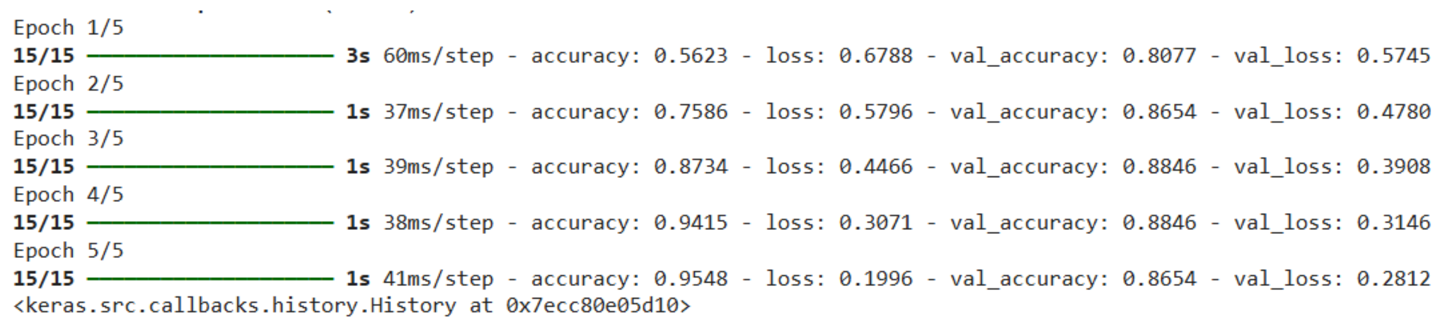


```
LSTM Evaluation
5/5 ━━━━━━━━━━━━━━━━━━━━ 1s 206ms/step
              precision    recall  f1-score   support

           0       0.88      0.89      0.88        55
           1       0.92      0.91      0.91        74

    accuracy                           0.90       129
   macro avg       0.90      0.90      0.90       129
weighted avg       0.90      0.90      0.90       129
```

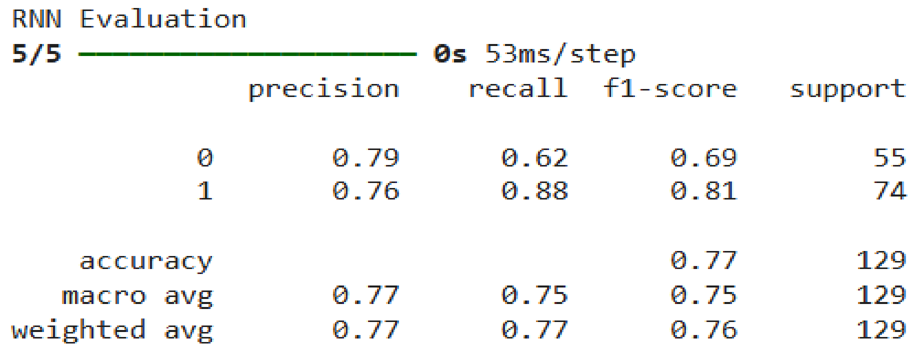




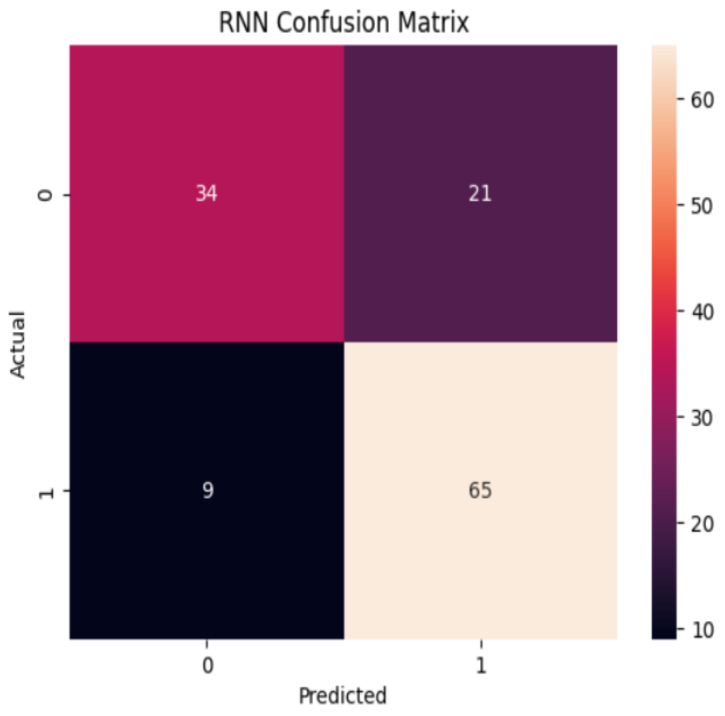
RNN

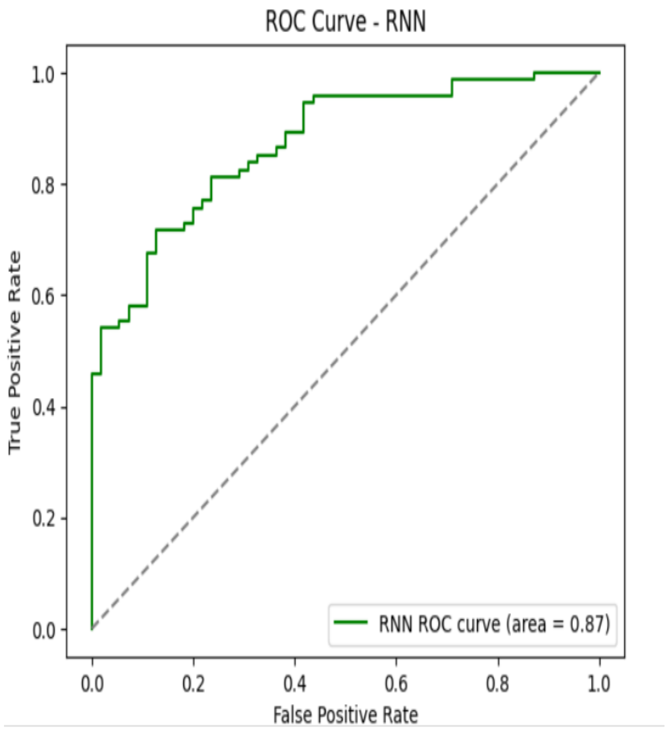


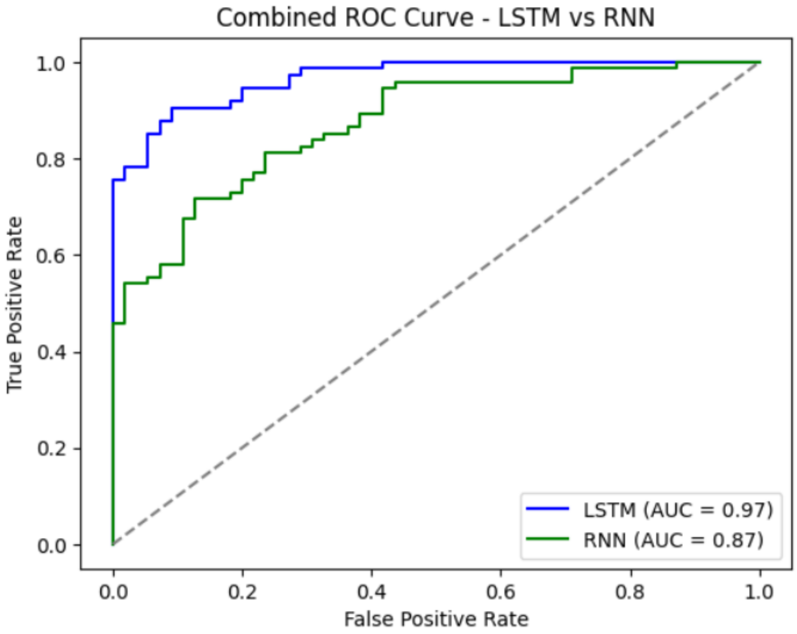
```
Epoch 1/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 3s 60ms/step - accuracy: 0.5623 - loss: 0.6788 - val_accuracy: 0.8077 - val_loss: 0.5745
Epoch 2/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.7586 - loss: 0.5796 - val_accuracy: 0.8654 - val_loss: 0.4780
Epoch 3/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 1s 39ms/step - accuracy: 0.8734 - loss: 0.4466 - val_accuracy: 0.8846 - val_loss: 0.3908
Epoch 4/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.9415 - loss: 0.3071 - val_accuracy: 0.8846 - val_loss: 0.3146
Epoch 5/5
15/15 ━━━━━━━━━━━━━━━━━━━━ 1s 41ms/step - accuracy: 0.9548 - loss: 0.1996 - val_accuracy: 0.8654 - val_loss: 0.2812
<keras.src.callbacks.history.History at 0x7ecc80e05d10>
```

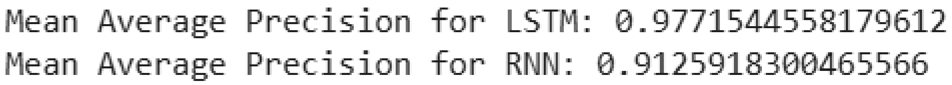


```
RNN Evaluation
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 53ms/step
              precision    recall  f1-score   support

           0       0.79      0.62      0.69        55
           1       0.76      0.88      0.81        74

    accuracy                           0.77       129
   macro avg       0.77      0.75      0.75       129
weighted avg       0.77      0.77      0.76       129
```









```
Mean Average Precision for LSTM: 0.9771544558179612
Mean Average Precision for RNN: 0.9125918300465566
```

Conclusion

This project focused on analyzing the sentiment of YouTube comments using deep learning models, specifically LSTM and RNN. After applying various NLP preprocessing techniques, both models were trained and evaluated. The LSTM model outperformed the RNN in key metrics such as accuracy, F1-score, and AUC, indicating its superior ability to capture context in sequential text data. Statistical tests validated the performance difference between the two models. Overall, the study highlights LSTM as a more effective approach for sentiment classification in social media comment analysis.