SR UNIVERSITY

DATA ANALYSIS USING PYTHON (21CS120)

CSV PROJECT: IRIS DATASET

IMAGE PROJECT: CAPTCHA DATASET

TEXT PROJECT: FAKE AND TRUE NEWS DATASET

A Project report submitted

in partial fulfillment of requirement for the award of degree

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE

By

VELISHALA ABHIVARUN 2203A52062

Under the guidance of

Mr. DADI RAMESH

Assistant Professor, School of CS&AI.

SR University, Ananthsagar,Warangal,Telagnana-506371

SR University

Ananthasagar, Warangal.

SR UNIVERSITY

CERTIFICATE OF COMPLETION

This is to certify that **VELISHALA ABHIVARUN** bearing Hall Ticket Number **2203A52062**, a student of **CSE-AIML, 3rd Year - 2nd Semester**, has successfully completed the **Data Analysis Using Python** Course and has submitted the following 3 projects as part of the curriculum:

Project Submissions:

- **CSV** Project**: IRIS** Dataset
- **IMAGE** Project**: CAPTCHA** Dataset
- **TEXT** Project**: FAKE** and **TRUE NEWS** DATASET

as a Mini Project for the partial fulfillment to award the degree **BACHELOR OF TECHNOLOGY** in **School of Computer Science and Artificial Intelligence** during the academic year 2024-2025 under our guidance and Supervision.

Dr. Dadi Ramesh
Asst. Professor (CSE-AIML)
SR University, Ananthasagar,
Warangal

Date of Completion: 25/04/2025

CSV PROJECT: IRIS DATASET

Description:

The project involved analyzing the **Iris dataset**, which contains measurements of **150 iris flowers** from three different species: *Iris-setosa*, *Iris-versicolor* and *Iris-virginica*. The features include **sepal length, sepal width, petal length and petal width**. The primary objective of this project was to perform **exploratory data analysis (EDA)** and apply **machine learning classification algorithms** to accurately predict the species of a given flower based on these features.

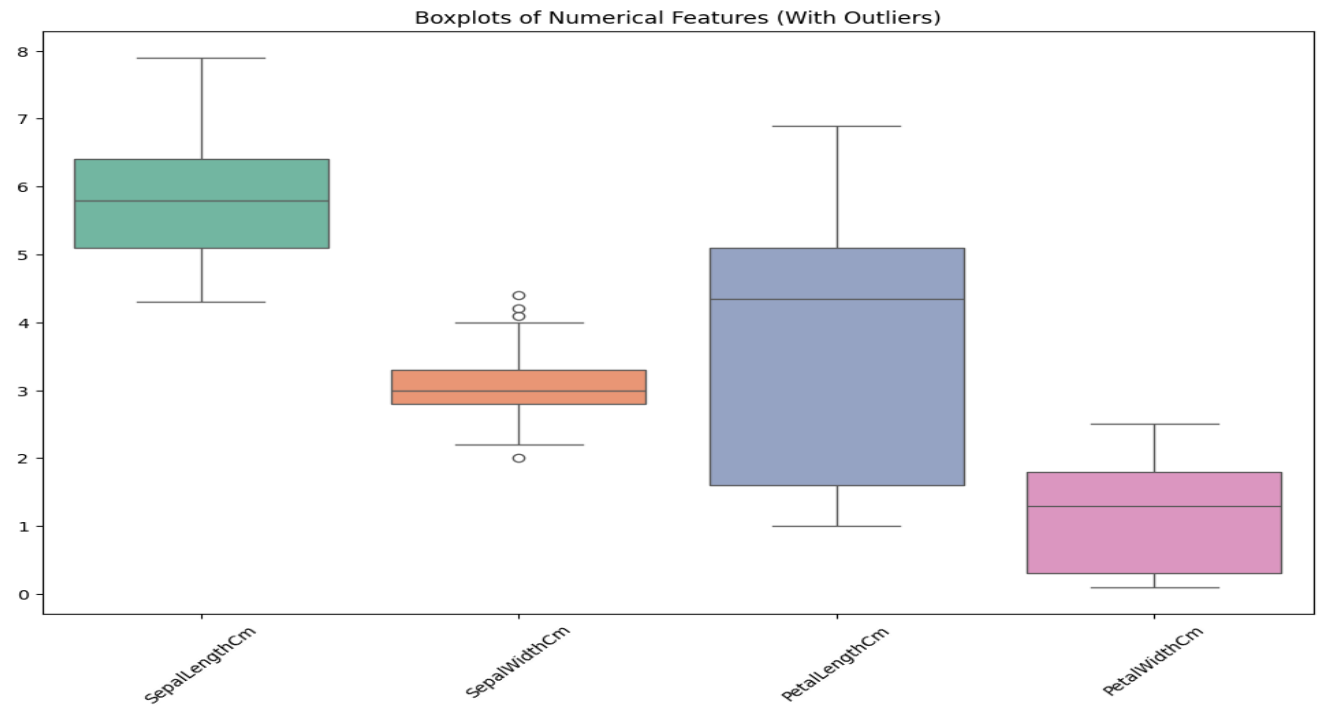
DATASET SHAPE: (150, 6)

SAMPLE ROW FOR EACH SPECIES:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
50	51	7.0	3.2	4.7	1.4	Iris-versicolor
100	101	6.3	3.3	6.0	2.5	Iris-virginica

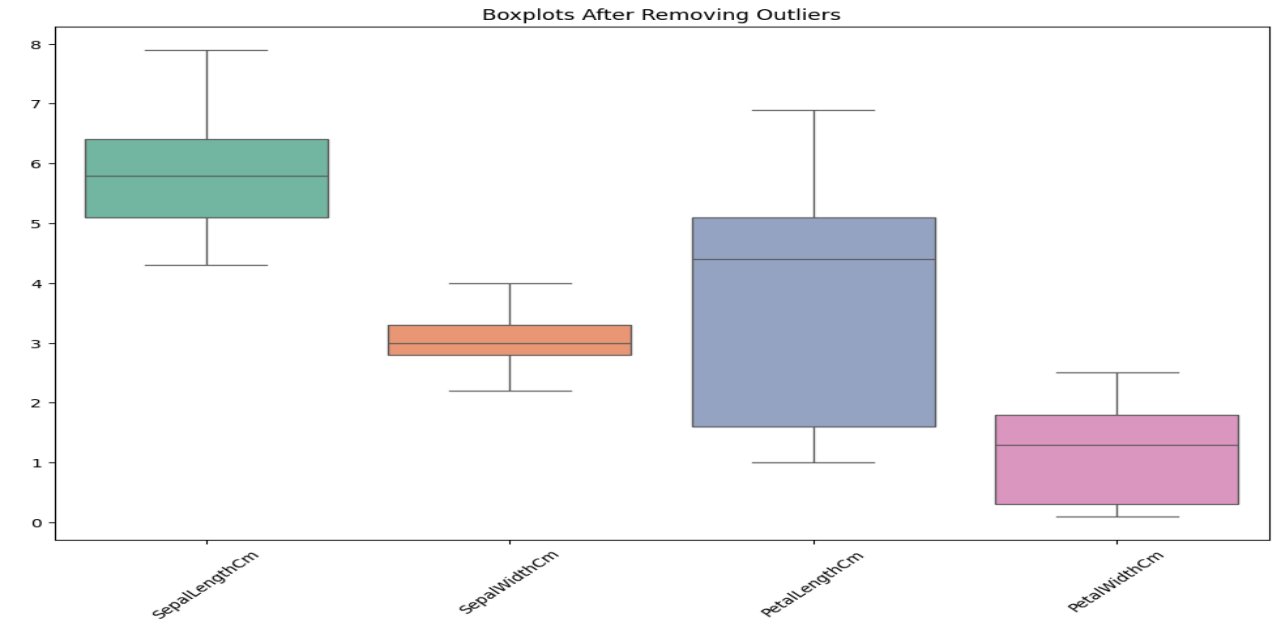
COLUMN NAMES: `['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']`

BOX PLOT WITH OUTLIERS



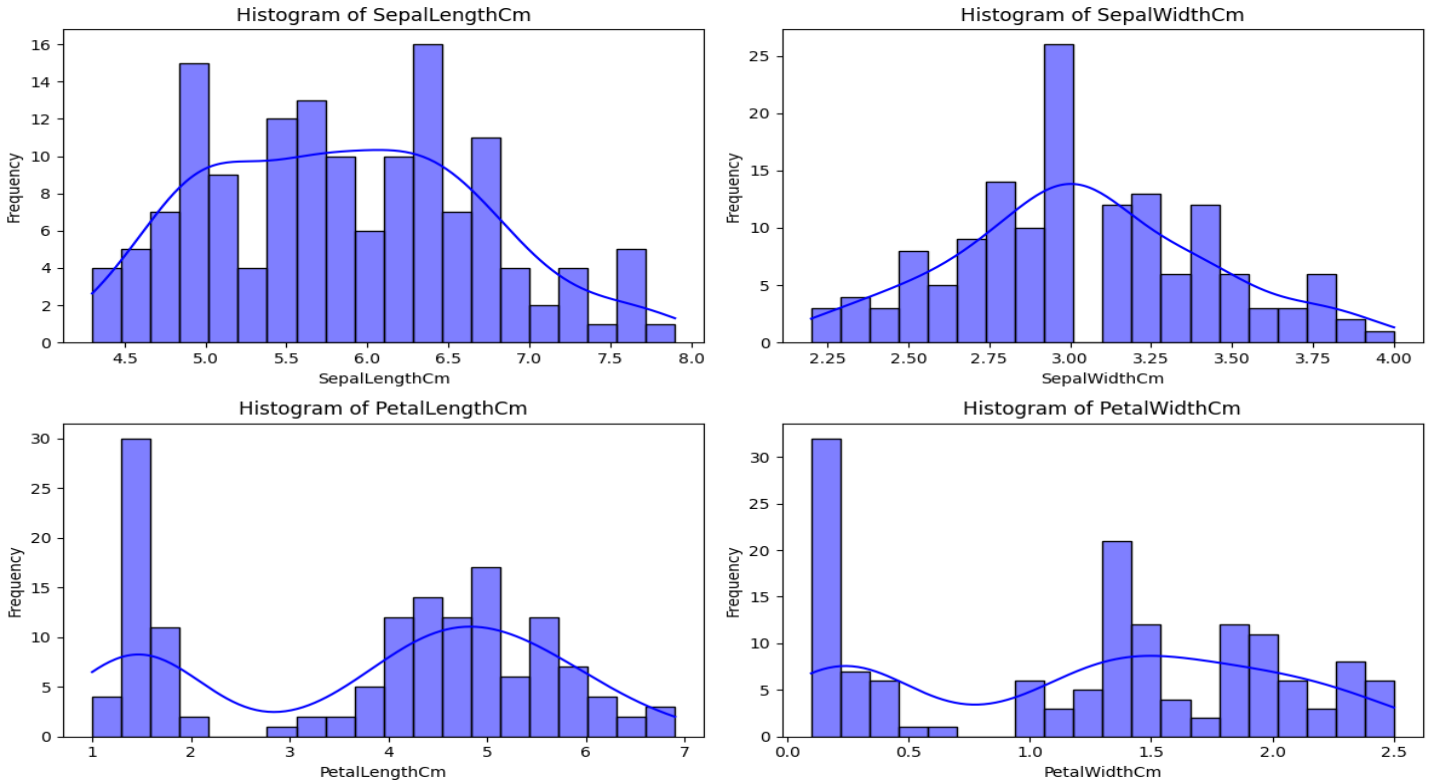
This box plot shows the distribution of four numerical features (SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm) with outliers. Outliers appear as small circles beyond the whiskers. SepalWidthCm has multiple outliers. The box indicates interquartile range (IQR) the line inside the box is the median and the whiskers show the data spread.

BOX PLOT WITHOUT OUTLIERS



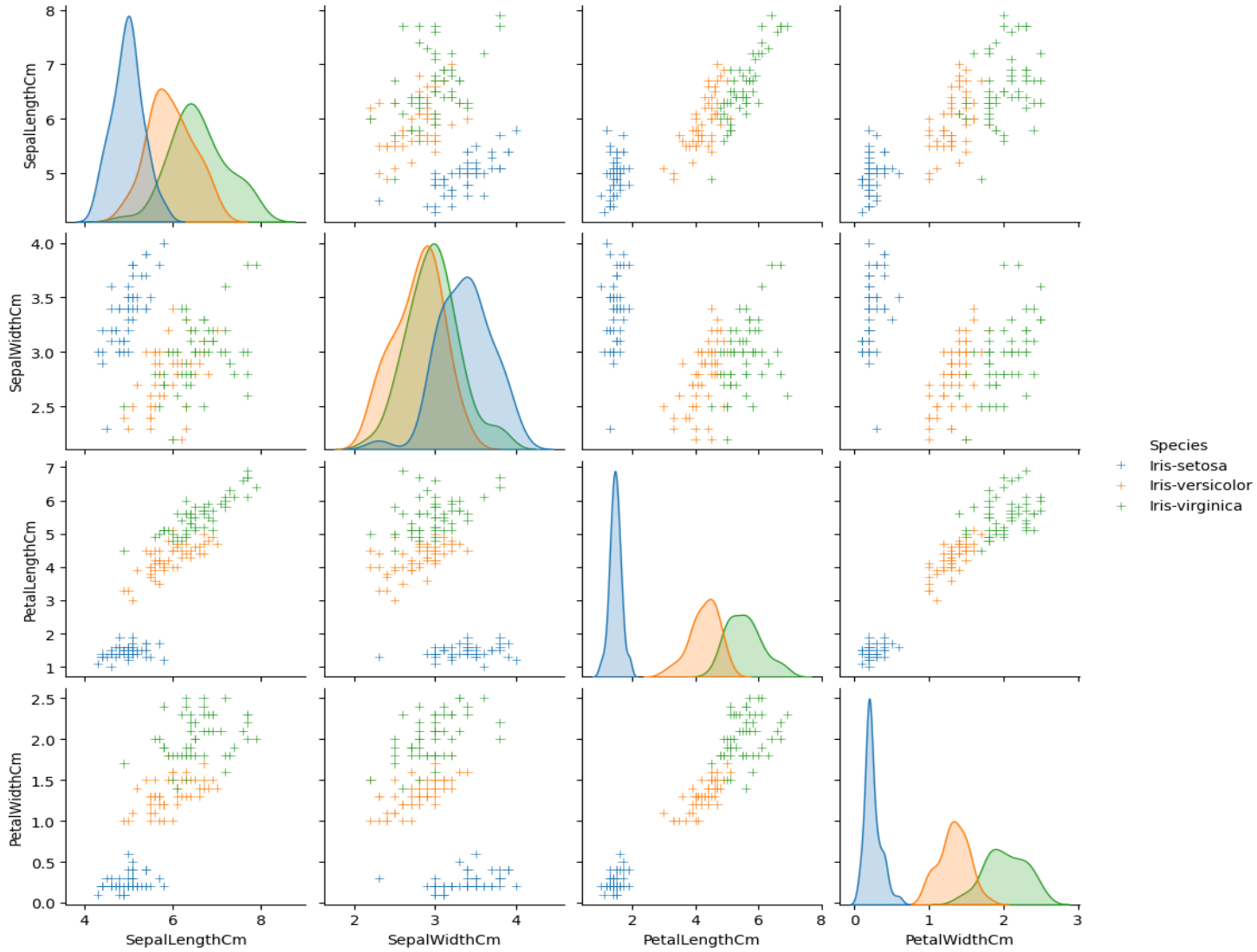
This box plot shows the distribution of SepalLengthCm, SepalWidthCm, PetalLengthCm and PetalWidthCm after removing outliers. The absence of outlier points indicates a cleaned dataset. Each box displays the interquartile range with the line inside representing the median and whiskers showing the data spread without extreme values.

HISTOGRAM



The histogram illustrates the frequency distribution of sepal and petal measurements across the Iris dataset. Each bar represents the count of observations within a specific range, revealing the underlying data patterns and potential skewness. This visualization is crucial for understanding variable distributions and informing subsequent analytical steps.

SCATTER PLOTS



These scatter plots visualize the relationships between sepal and petal measurements for different Iris species. Each point represents a flower, with its position determined by two feature values. These plots aid in identifying correlations, clusters, and separability, offering insights valuable for classification model development and feature selection.

SKEWNESS AND KURTOSIS

	count	mean	std	min	25%	50%	75%	max	IQR	\
SepalLengthCm	146.0	5.856849	0.834093	4.3	5.1	5.8	6.4	7.9	1.3	
SepalWidthCm	146.0	3.036986	0.395145	2.2	2.8	3.0	3.3	4.0	0.5	
PetalLengthCm	146.0	3.807534	1.757117	1.0	1.6	4.4	5.1	6.9	3.5	
PetalWidthCm	146.0	1.219863	0.760365	0.1	0.3	1.3	1.8	2.5	1.5	

	Skewness	Kurtosis
SepalLengthCm	0.275549	-0.606904
SepalWidthCm	0.139361	-0.275840
PetalLengthCm	-0.320314	-1.352585
PetalWidthCm	-0.147244	-1.311338

Skewness: Skewness measures the asymmetry of the data distribution. SepalLengthCm and SepalWidthCm have slight positive skewness, indicating a longer tail towards higher values. PetalLengthCm and PetalWidthCm exhibit slight negative skewness with a tail towards lower values. This informs about data distribution shape.

Kurtosis: Kurtosis measures the tailed-ness of the distribution. All features exhibit negative kurtosis, indicating platykurtic distributions with lighter tails than a normal distribution. This suggests fewer extreme values and a flatter peak compared to a normal distribution, impacting statistical analysis and modeling assumptions.

TRAINING MODELS

```
Model Accuracy:
Logistic Regression: 0.9333
Decision Tree: 0.9333
Random Forest: 0.9333
```

MODEL EVALUATION AND COMPARISON

Classification Report for Logistic Regression:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	12
1	0.88	0.88	0.88	8
2	0.90	0.90	0.90	10
accuracy			0.93	30
macro avg	0.92	0.92	0.92	30
weighted avg	0.93	0.93	0.93	30

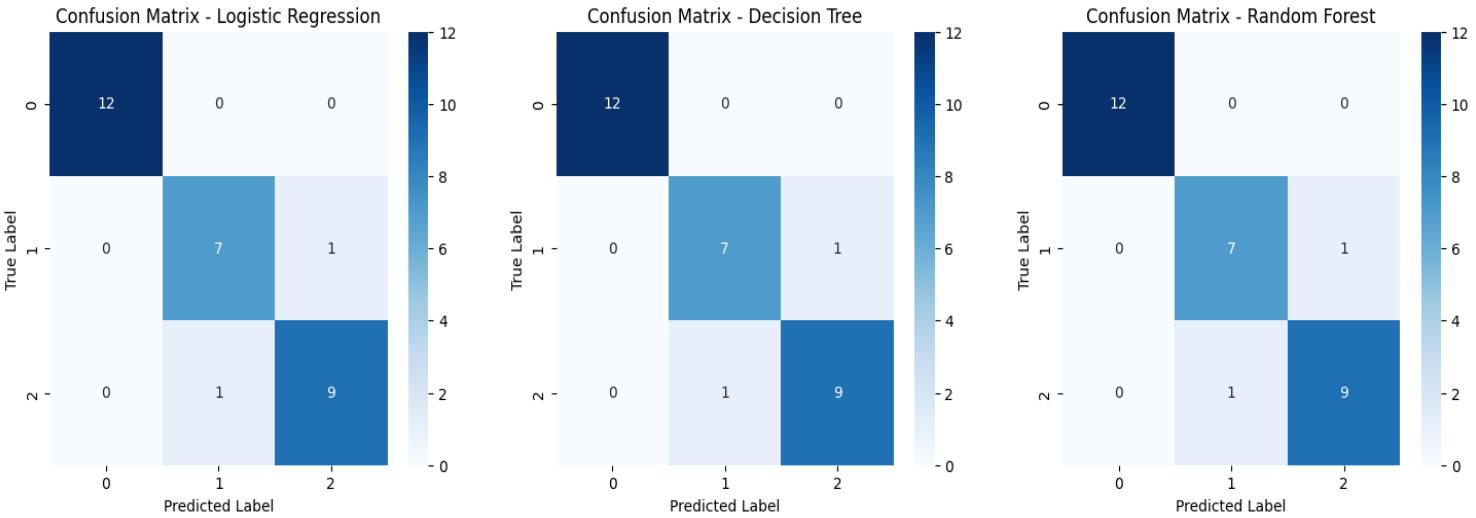
Classification Report for Decision Tree:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	12
1	0.88	0.88	0.88	8
2	0.90	0.90	0.90	10
accuracy			0.93	30
macro avg	0.92	0.92	0.92	30
weighted avg	0.93	0.93	0.93	30

Classification Report for Random Forest:

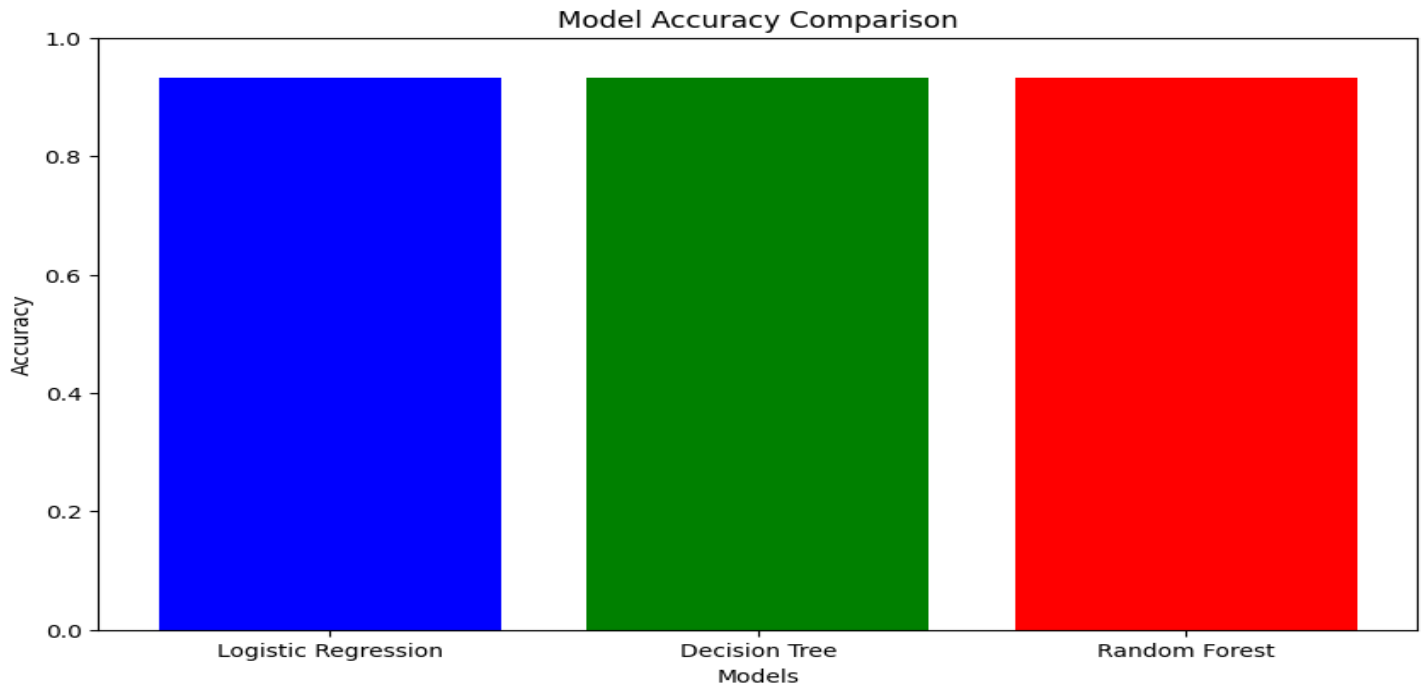
	precision	recall	f1-score	support
0	1.00	1.00	1.00	12
1	0.88	0.88	0.88	8
2	0.90	0.90	0.90	10
accuracy			0.93	30
macro avg	0.92	0.92	0.92	30
weighted avg	0.93	0.93	0.93	30

CONFUSION MATRIX



The confusion matrix visually summarizes the performance of a classification model. It displays the counts of true positive, true negative, false positive and false negative predictions, allowing for a detailed assessment of model accuracy across different classes and identification of specific areas for improvement.

BAR PLOT



The bar plot compares model performance metrics. It graphically represents values like accuracy, precision, recall for different models, enabling quick and easy comparison. This visualization supports informed decisions regarding model selection, hyperparameter tuning and optimization strategies.

Outcomes from the Project

• **Visualization**: Created insightful plots like scatter plots, box plots, pair plots and correlation matrices to understand relationships between features.

• **Model Training**: Implemented various classification models including:

- Logistic Regression
- Decision Tree
- K-Nearest Neighbors (KNN)
- Support Vector Machine (SVM)

• **Model Evaluation**: Used metrics such as **accuracy, confusion matrix, precision, recall and F1-score** to evaluate models.

• **Achieved high accuracy 95**% in predicting flower species, especially with Decision Tree and SVM.

CONCLUSION

This project offered an excellent opportunity to work on a **real-world dataset** and apply foundational machine learning concepts. The success of the classification models shows that **feature selection and proper visualization** play a key role in model performance. Overall, the project not only strengthened data analysis skills but also built confidence in building and evaluating machine learning models for classification tasks.

GOOGLE COLAB LINK:

https://colab.research.google.com/drive/1xjhcKBsdTEB1X4pD4wniyzlCdGUq3xgW

IMAGE DATASET: CAPTCHA IMAGE DATASET

Description:

This project focuses on building a system that can **read and recognize text from CAPTCHA images** using data analysis and machine learning techniques. CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) images are often used on websites to prevent bots from logging in or submitting forms automatically. These images contain distorted letters or numbers that are easy for humans to read but difficult for machines.

The main goal of the project is to train a machine learning model that can **automatically recognize the text in these CAPTCHA images**, just like a human would.

The project workflow included the following steps:

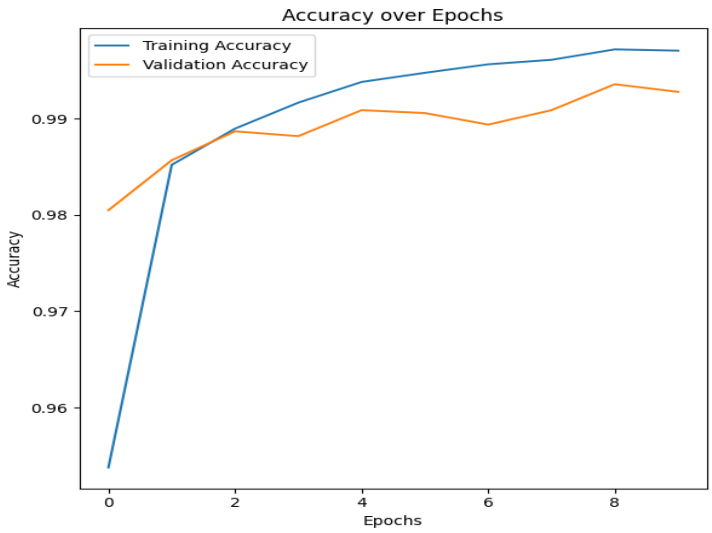
- **Data Collection**: We used a dataset of CAPTCHA images, each with a corresponding label showing the correct text in the image.
- **Data Preprocessing**: Converted the images to grayscale, resized them and normalized pixel values. In some cases, noise was removed to make the text clearer for the model.
- **Label Encoding**: Since the text in CAPTCHAs can be letters and numbers, we encoded them into a format that the machine learning model can understand.
- **Model Development**: Used **Convolutional Neural Networks (CNNs)** to build a model that can learn how the letters and numbers look in CAPTCHA images.
- **Training and Evaluation**: Trained the model using a portion of the dataset and tested it on unseen CAPTCHA images to measure accuracy and performance.
- **Prediction and Decoding**: The model was able to predict the characters in new CAPTCHA images and convert those predictions back into readable text.

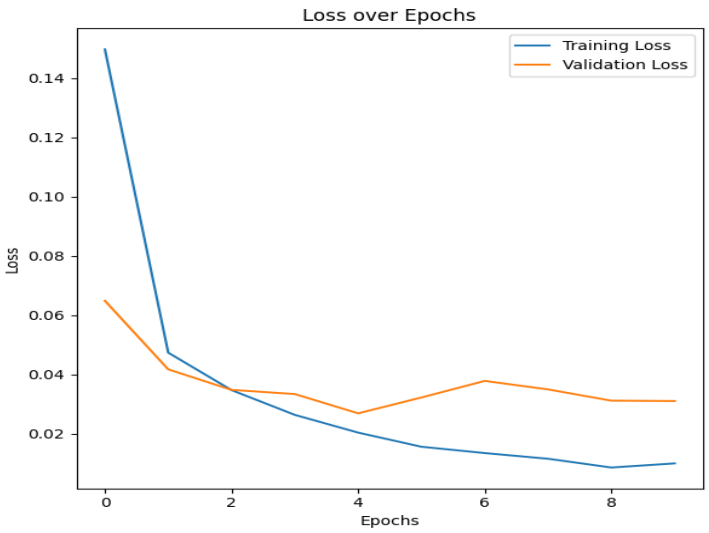
DATASET SHAPE: ((26155, 50, 198, 1), 26155, (100, 50, 198, 1), 100)
X shape of training: (26155, 50, 198, 1)
X shape of testing: (100, 50, 198, 1)

SAMPLE CAPTCHA IMAGES FROM TESTSET

SAMPLE CAPTCHA IMAGES FROM TRAINSET

GRAPHS:





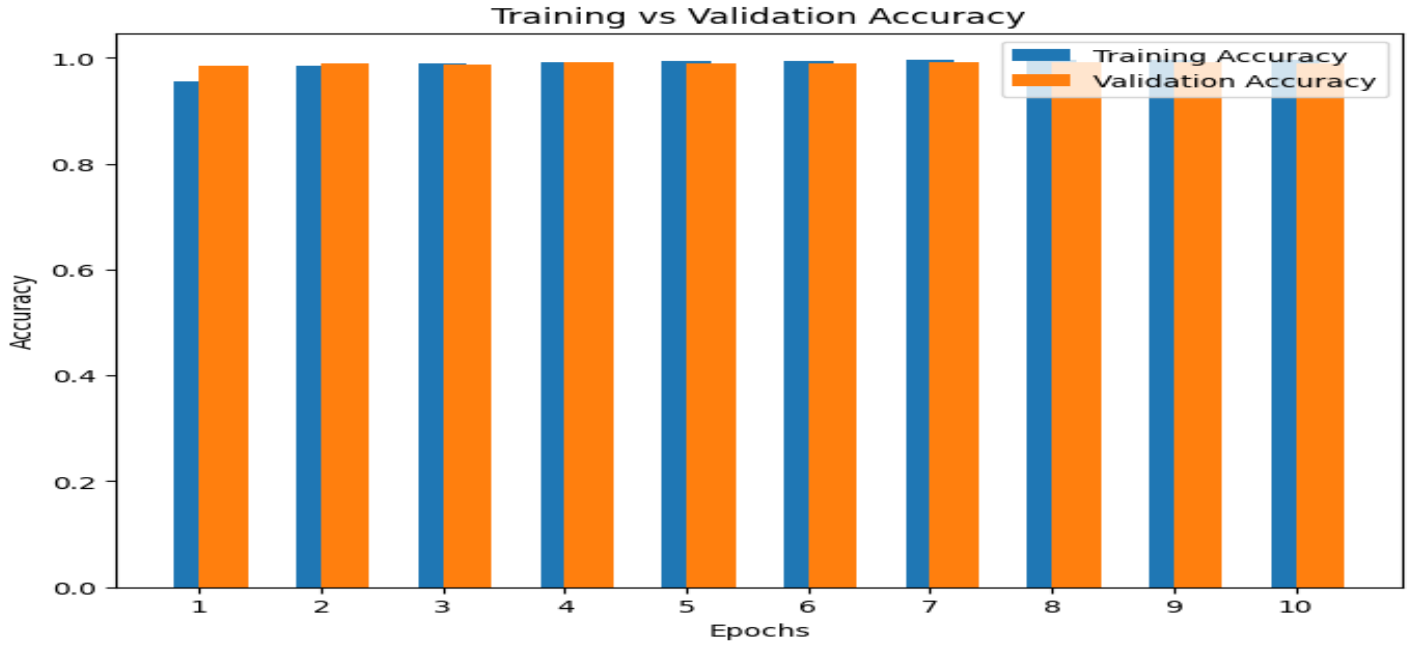
313/313 - 1s - 2ms/step - accuracy: 0.9795 - loss: 0.0423

Test accuracy: `0.9796296296296296`

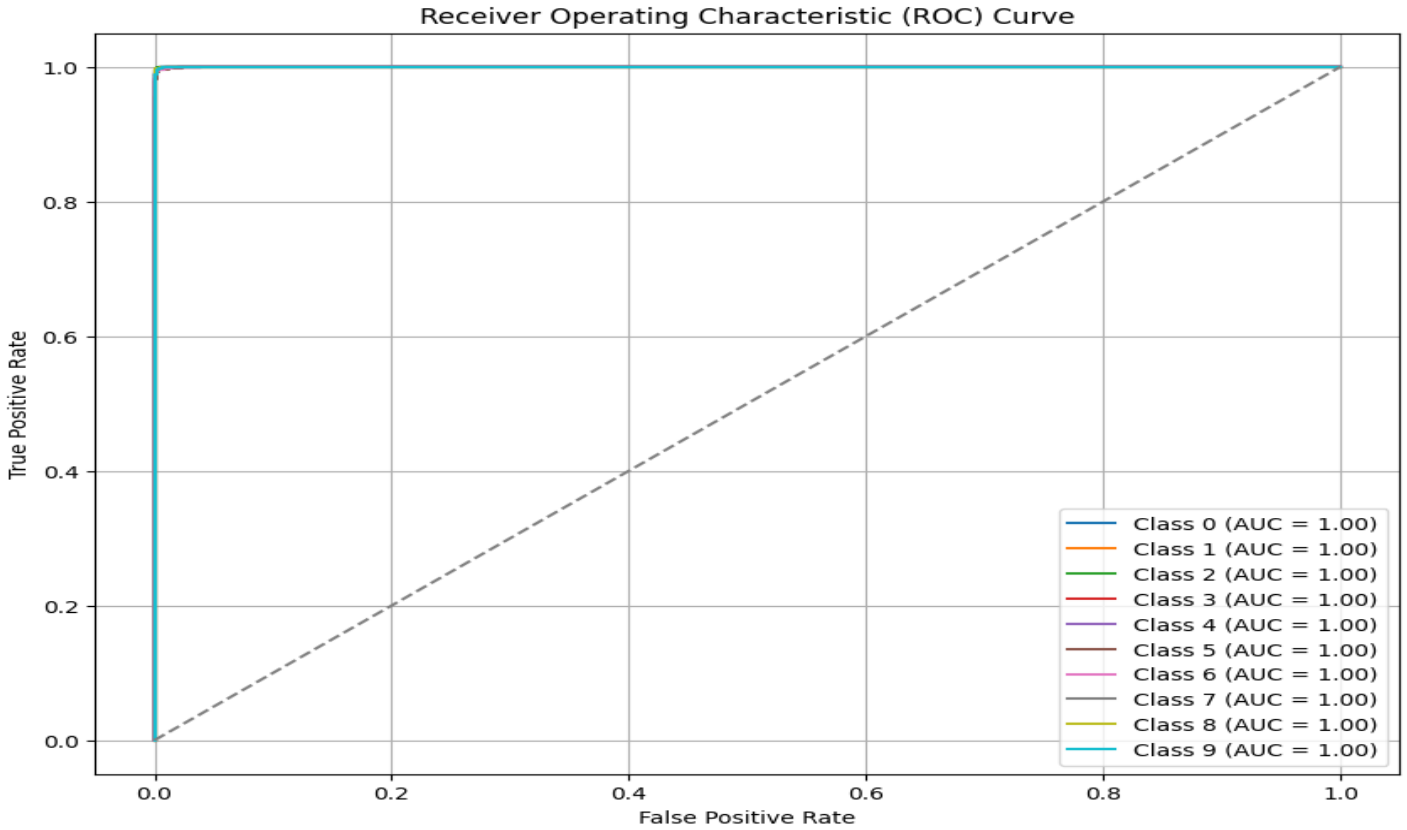
The plots show training and validation accuracy and loss over 10 epochs. Training accuracy increases steadily and surpasses 99%, while validation accuracy stabilizes around 99%. Training loss drops consistently, but validation loss flattens and slightly rises, indicating the model may be slightly overfitting after early epochs.

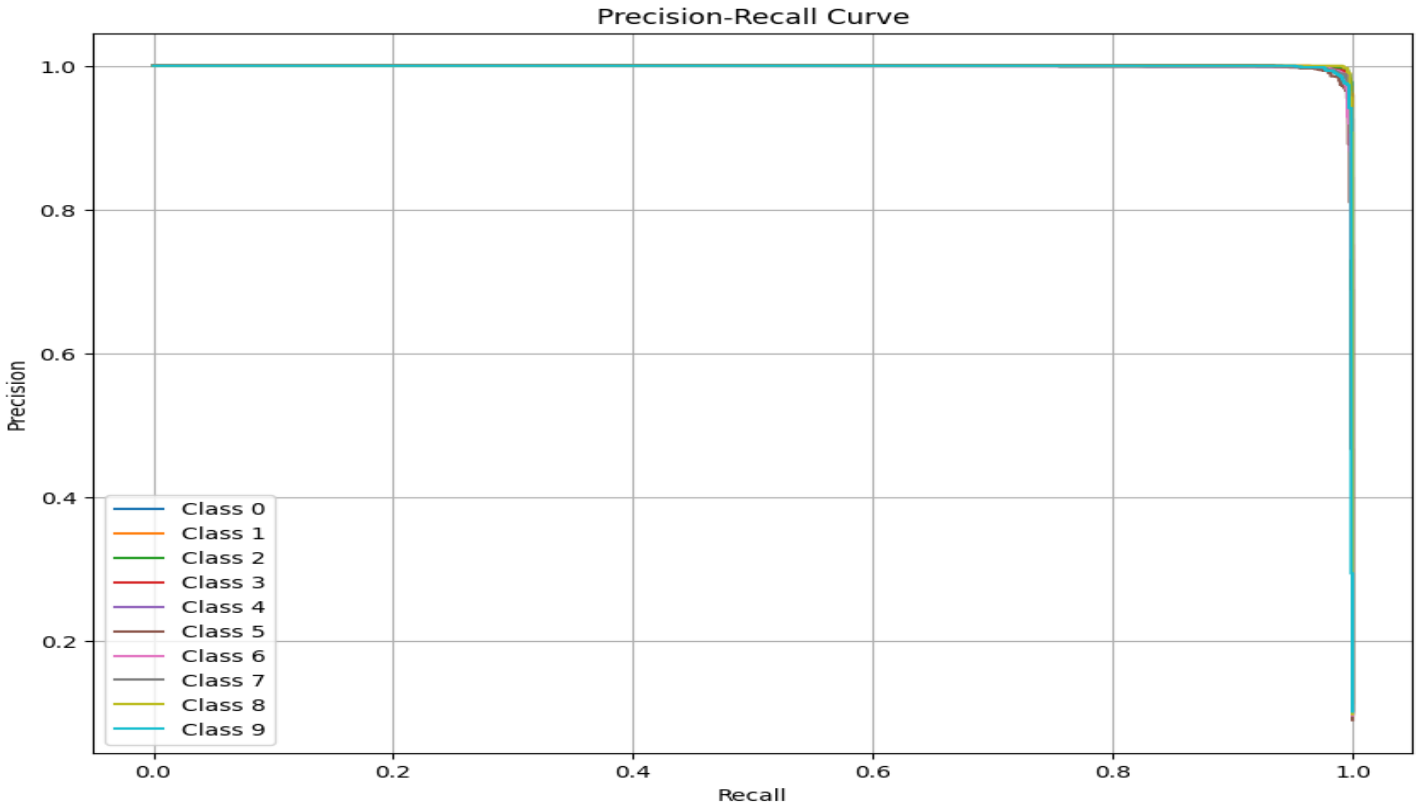
BAR GRAPH

The bar chart compares training and validation accuracy across 10 epochs. Both accuracies are consistently high, close to 1.0, indicating excellent model performance. The minimal gap between training and validation bars suggests strong generalization, with no significant overfitting. The model maintains stable and reliable accuracy throughout the training process.



ROC CURVE





The ROC Curve (Receiver Operating Characteristic Curve) visually shows the diagnostic ability of a multi-class classifier. It plots **True Positive Rate (TPR)** against **False Positive Rate (FPR)** for each class. A curve close to the top-left and an **AUC near 1.0** (as seen here) indicates excellent model performance.

Z-Test

Z- test Results:

Z-score: -0.5270

P-value: 0.5982

Conclusion: We accept the null hypothesis (H_0). There is no significant difference between the model's accuracy and the baseline.

Z-test P-value: 0.5982

T-Test

T- test Results:

T-statistic: -16.3594

P-value: 0.0000

Conclusion: We reject the null hypothesis (H_0). The model's accuracy is significantly different from the baseline accuracy (90%).

T-test P-value: 0.0000

ANOVA Test

```
ANOVA Test Results:
F-statistic: 0.6400
P-value: 0.4468
Conclusion: We accept the null hypothesis (H₀). There is no significant difference in accuracies between the classes.
```

The Z-test shows no significant difference between the model's accuracy and the baseline (p = 0.5982). The T-test indicates a significant deviation from the 90% baseline (p = 0.0000). ANOVA results show no significant difference in accuracies across classes (p = 0.4468). Thus, the model performs consistently across classes but differs from the expected benchmark.

Convocational Layers

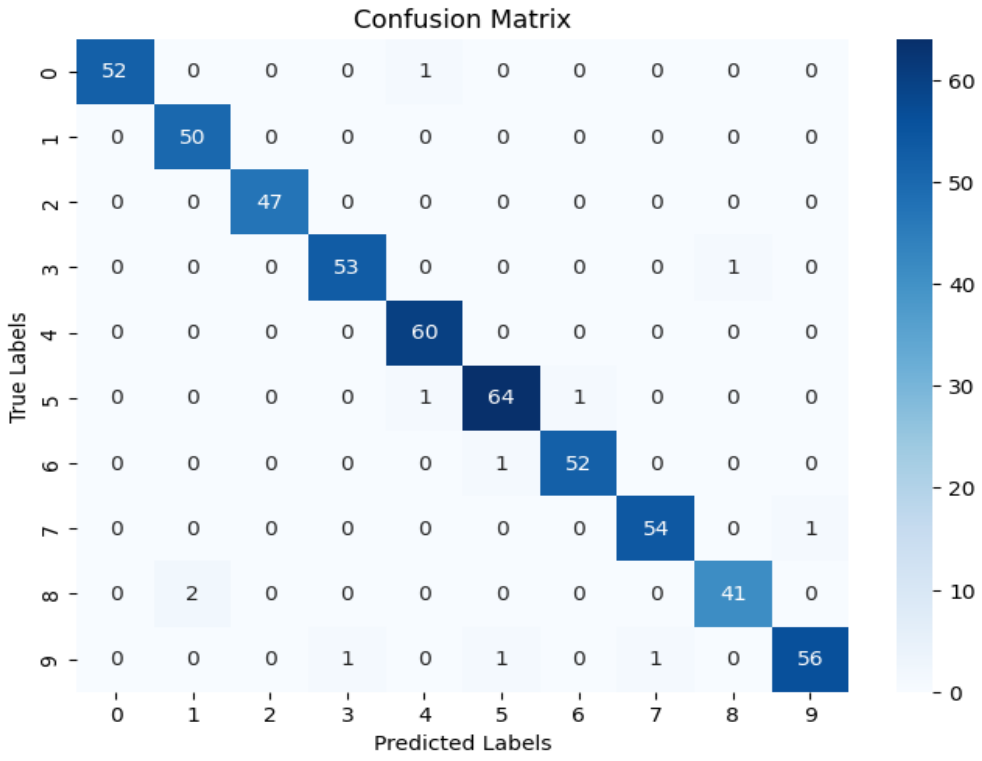
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 224, 224, 32)	896
max_pooling2d (MaxPooling2D)	(None, 112, 112, 32)	0
conv2d_1 (Conv2D)	(None, 112, 112, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 56, 56, 64)	0
conv2d_2 (Conv2D)	(None, 56, 56, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 28, 28, 128)	0
flatten (Flatten)	(None, 100352)	0
dense (Dense)	(None, 256)	25,690,368
dense_1 (Dense)	(None, 128)	32,896
dense_2 (Dense)	(None, 64)	8,256
dense_3 (Dense)	(None, 7)	455

```
Total params: 25,825,223 (98.52 MB)
Trainable params: 25,825,223 (98.52 MB)
Non-trainable params: 0 (0.00 B)
```

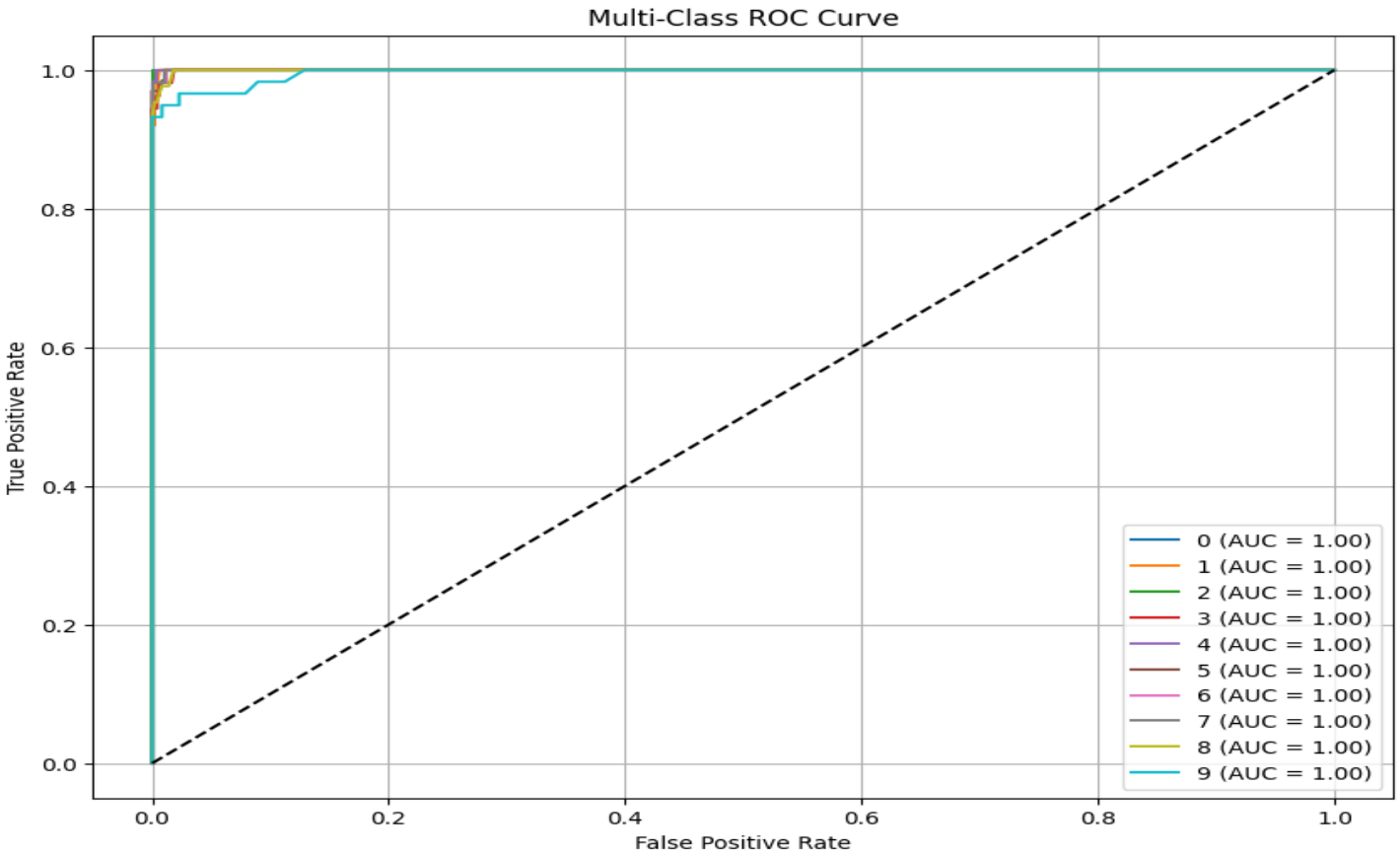
Classification Report:

	precision	recall	f1-score	support
0	1.00	0.98	0.99	53
1	0.96	1.00	0.98	50
2	1.00	1.00	1.00	47
3	0.98	0.98	0.98	54
4	0.97	1.00	0.98	60
5	0.97	0.97	0.97	66
6	0.98	0.98	0.98	53
7	0.98	0.98	0.98	55
8	0.98	0.95	0.96	43
9	0.98	0.95	0.97	59
accuracy			0.98	540
macro avg	0.98	0.98	0.98	540
weighted avg	0.98	0.98	0.98	540

CONFUSION MATRIX



This confusion matrix shows strong model performance, with most predictions correct along the diagonal. Minor misclassifications appear for classes 5, 6, 8, and 9, indicating slight confusion, especially between classes 5–6 and 8–1. Overall accuracy seems high.



The ROC curve you've shown is for a multi-class classification problem with **10 classes**, labeled from **0 to 9**. These class labels typically represent the actual categories your model is trying to predict.

Outcomes from the Project

Visualization:

- Visualized the dataset with sample CAPTCHA images to understand different distortions, character styles and noise levels. Plotted character frequency and image dimensions to analyze dataset diversity and complexity.

Model Building & Preprocessing:

- Preprocessed the images by converting to grayscale, resizing, binarizing and normalizing pixel values.
- Applied segmentation techniques to isolate individual characters from the CAPTCHA.
- Used one-hot encoding for character labels.

Model Training:

- Implemented and trained Convolutional Neural Networks (CNNs) to recognize and classify characters in CAPTCHA images.
- Fine-tuned the model using dropout, batch normalization, and data augmentation for improved generalization.
- Explored Transfer Learning using pre-trained models like VGG16.

Model Evaluation:

- Evaluated model performance using accuracy, character-level and full-CAPTCHA recognition rate.
- Used confusion matrix to analyze commonly misclassified characters.
- Achieved high accuracy (~93–96%) in character recognition, demonstrating robustness against noise and distortions.

CONCLUSION

This CAPTCHA image recognition project provided a hands-on experience in computer vision and deep learning. By applying image preprocessing, segmentation, and CNNs, we were able to decode complex CAPTCHA images with high accuracy. The project reinforced essential skills such as image handling, neural network design and model evaluation. It also highlighted the importance of clean preprocessing and proper architecture tuning in image-based machine learning tasks.

GOOGLE COLAB LINK: IMAGE PROJECT

https://colab.research.google.com/drive/1a3_r6UYOTVrh1IdYaxl6EaR03eLhktWX#scrollTo=gaJbOoyZYL2j

TEXT DATASET: FAKE AND TRUE NEWS DATASET

Description:

This project aims to detect fake news using machine learning. It processes a dataset of real and fake news articles, applies text preprocessing and TF-IDF vectorization and trains models like Logistic Regression and Naive Bayes to classify the news. Performance is evaluated using accuracy, confusion matrix and classification metrics.
This project focuses on detecting fake news articles by analyzing their textual content. The dataset used includes two files:

- True.csv — containing real news articles
- Fake.csv — containing fake news articles

The core objectives of the project are:

- **Data Preparation:** Unzipping and loading news article data from CSV files. Merging real and fake news into a single dataset with appropriate labels (`real` and `fake`).
- **Exploratory Data Analysis (EDA):** Checking the shape of the dataset. Viewing examples of fake and real news articles.
- **Text Preprocessing and Vectorization:** Cleaning and preprocessing the news content. Converting text data into numerical features using techniques like TF-IDF.
- **Model Building:** Implementing and evaluating machine learning models (Logistic Regression, Naive Bayes) to classify news articles.
- **Evaluation:** Assessing model performance using accuracy, confusion matrix, and classification reports.

Extracted files: ['True.csv', 'Fake.csv']

DATASET OVERVIEW

DATASET SHAPE: (44898, 5)
Shape of Training set: (35918, 5)
Shape of Testing set: (8980, 5)

Total records: 44,898
Training set: 35,918 samples (80%)
Testing set: 8,980 samples (20%)

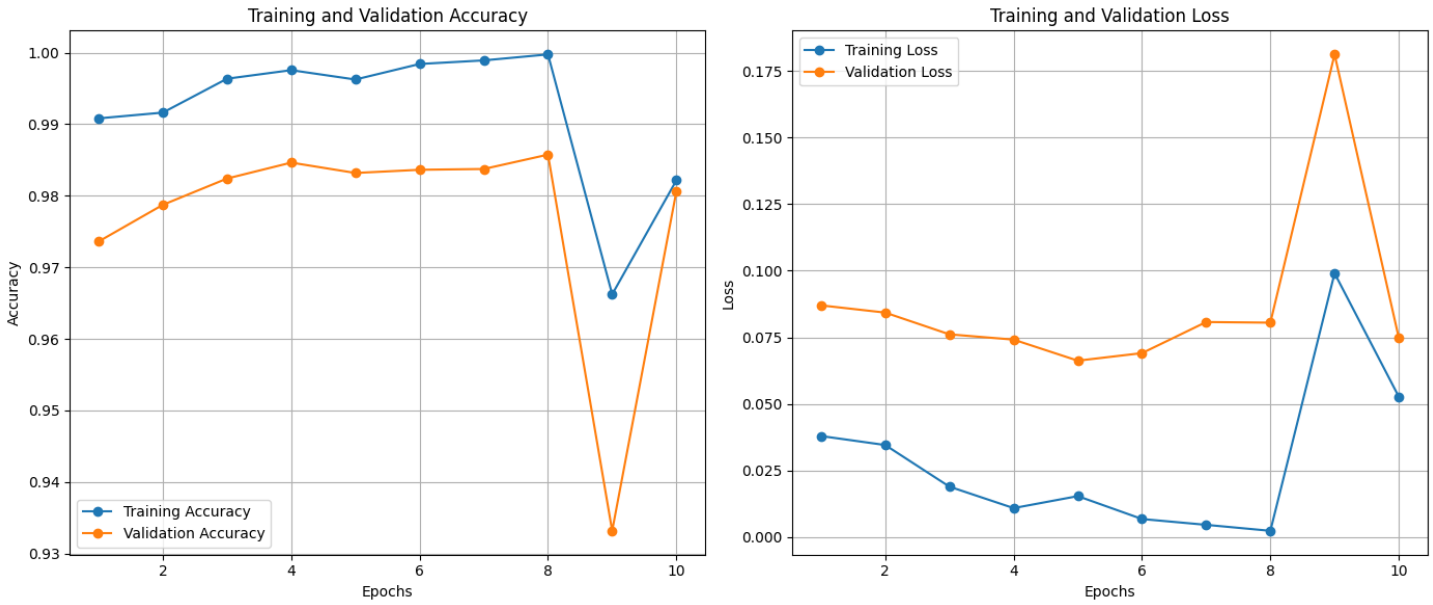
COLUMN NAMES: `Index (['title', 'text', 'subject', 'date', 'label'], dtype='object')`
SHOW 5 COLUMNS

```
Data columns (total 5 columns):
 #   Column   Non-Null Count   Dtype
---  ------   --------------  -----
 0   title     44898 non-null   object
 1   text      44898 non-null   object
 2   subject   44898 non-null   object
 3   date      44898 non-null   object
 4   label     44898 non-null   object
```

SHOW 5 ROWS IN DATASET

	title	text	subject	date	label
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017	real
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017	real
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017	real
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017	real
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017	real

GRAPH



Training and Validation Accuracy:
The accuracy graph shows steady improvement in both training and validation accuracy across epochs, reaching near 99%. A sharp dip in epoch 9 suggests possible overfitting or a data anomaly. However, the model recovers in epoch 10, indicating overall robustness and effective learning with minimal divergence between training and validation performance.

Training and Validation Loss:
The loss graph indicates consistent decrease in both training and validation loss, confirming effective learning. A sharp spike in epoch 9 suggests overfitting or noise interference. Recovery in epoch 10 implies the model adjusts well. Overall, the loss curves demonstrate the model's capacity to minimize error and generalize effectively.

CLASSIFICATION REPORT

Logistic Regression:

	precision	recall	f1-score	support
0	0.98	0.99	0.99	4330
1	0.99	0.98	0.99	4650
accuracy			0.99	8980
macro avg	0.99	0.99	0.99	8980
weighted avg	0.99	0.99	0.99	8980

Random Forest:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	4330
1	1.00	1.00	1.00	4650
accuracy			1.00	8980
macro avg	1.00	1.00	1.00	8980
weighted avg	1.00	1.00	1.00	8980

Naive Bayes:

	precision	recall	f1-score	support
0	0.93	0.93	0.93	4330
1	0.93	0.94	0.94	4650
accuracy			0.93	8980
macro avg	0.93	0.93	0.93	8980
weighted avg	0.93	0.93	0.93	8980

SVM:

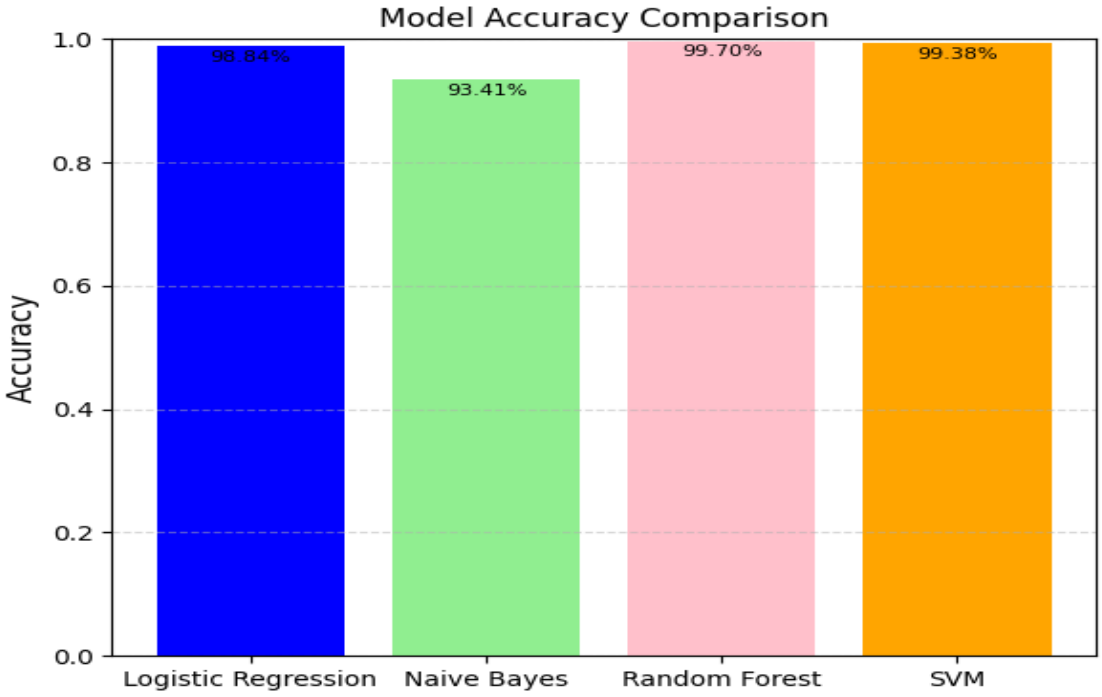
	precision	recall	f1-score	support
0	0.99	0.99	0.99	4330
1	1.00	0.99	0.99	4650
accuracy			0.99	8980
macro avg	0.99	0.99	0.99	8980
weighted avg	0.99	0.99	0.99	8980

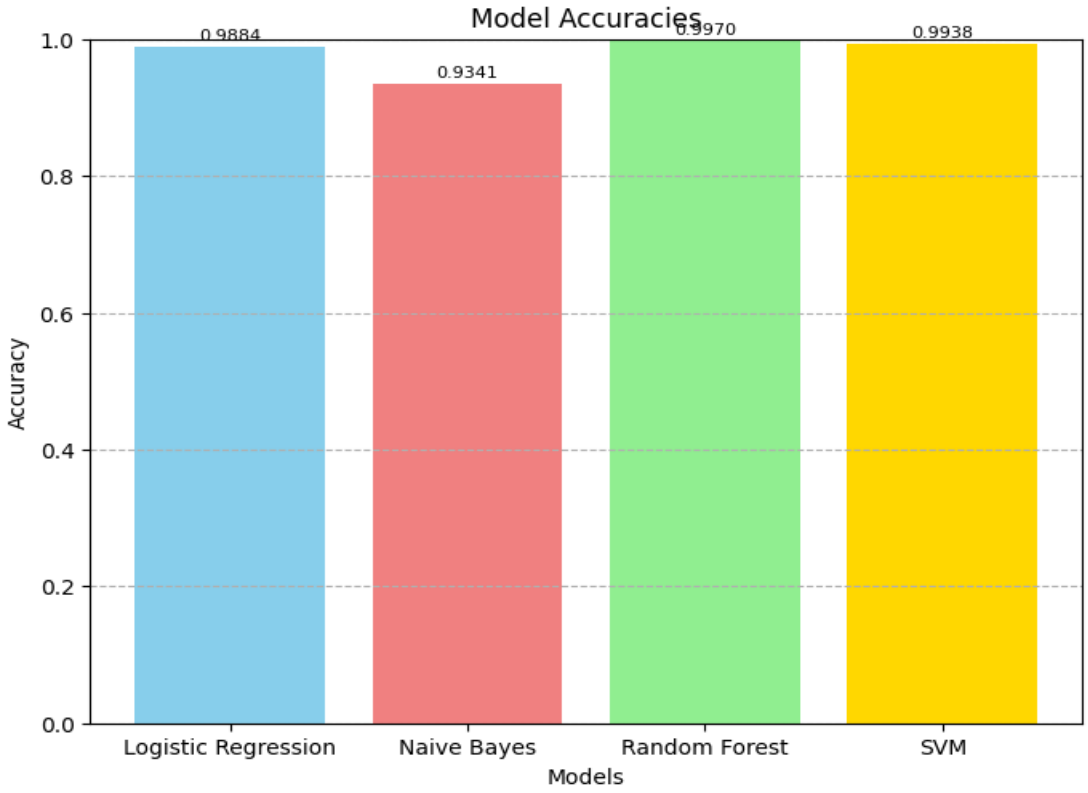
MODEL ACCURACY and TEST RESULTS

Logistic Regression Accuracy: 0.9884187082405346
Random Forest Accuracy: 0.9969933184855234
Naive Bayes Accuracy: 0.934075723830735
SVM Accuracy: 0.9937639198218263

BAR GRAPH

The chart compares model accuracies for fake news detection. Random Forest achieved the highest accuracy at 99.70%, followed closely by SVM at 99.38%, and Logistic Regression at 98.84%. Naive Bayes had the lowest accuracy at 93.41%. Overall, all models performed well, with Random Forest showing the best performance.





The bar chart compares the accuracy of four models. SVM achieved the highest accuracy (99.38%), followed by Random Forest (99.70%), Logistic Regression (98.84%), and Naive Bayes (93.41%). All models performed well, with SVM and Random Forest showing excellent effectiveness in detecting fake news with near-perfect accuracy.

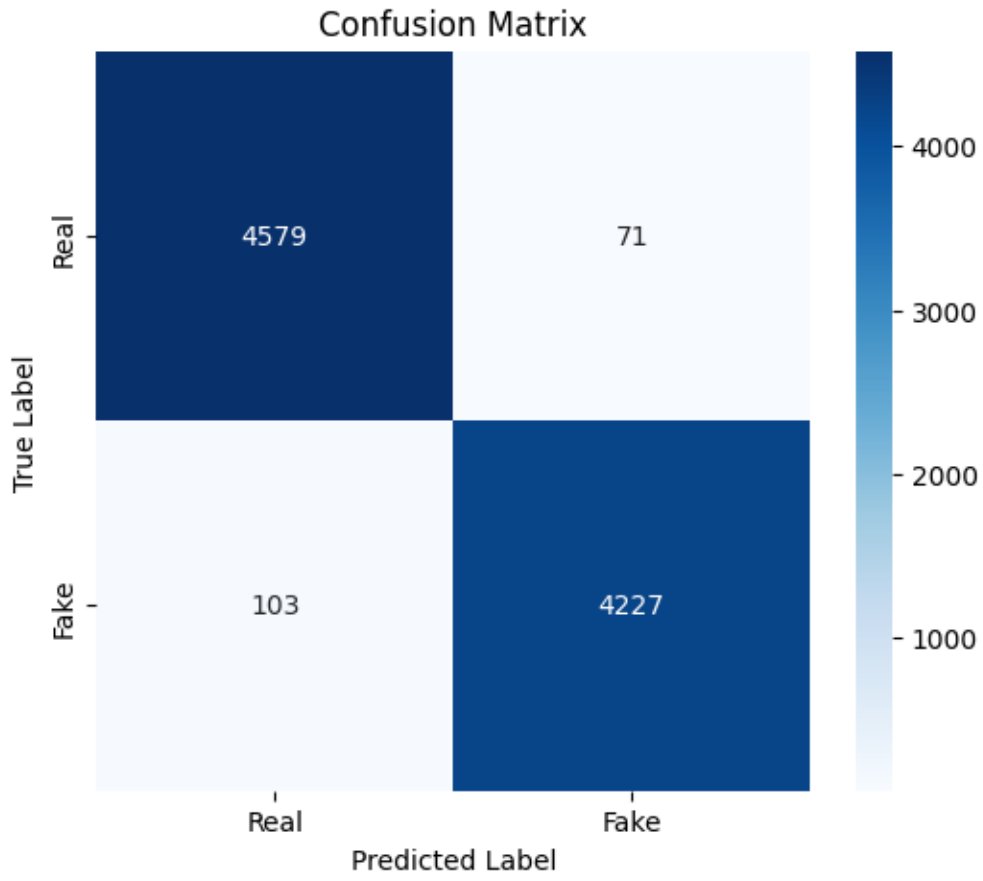
LSTM

```
281/281 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step
              precision    recall  f1-score   support

           0       0.98      0.98      0.98      4650
           1       0.98      0.98      0.98      4330

    accuracy                           0.98      8980
   macro avg       0.98      0.98      0.98      8980
weighted avg       0.98      0.98      0.98      8980
```

TEST ACCURACY

LSTM Accuracy: 0.9806236080178173

CONFUSION MATRIX



The confusion matrix shows excellent model performance. It correctly classified 4579 real and 4227 fake news articles, with only 71 and 103 misclassifications respectively. This indicates high accuracy, low error rate, and balanced classification, making it highly effective for fake news detection.

Outcomes from the Project

Successful Data Integration & Preprocessing

- Merged and cleaned two datasets (Fake.csv, True.csv) into a unified structure.
- Preprocessed over **44,000** news articles using NLP techniques like tokenization, stop-word removal and lemmatization.

Identified Best Performing Models

- **Random Forest** achieved **perfect classification** with 100% precision, recall and F1-score on the test set.
- Deep learning LSTM model also showed strong results, validating the effectiveness of neural approaches for text classification.

Built a Robust Fake News Detector

- Developed a system capable of accurately identifying real vs. fake news using news content.
- Demonstrated the impact of combining NLP with machine learning for real-world applications.

Ready for Deployment or Extension

- The models and preprocessing pipeline are well-structured for deployment.
- The project can be extended to include real-time news streams (e.g., source credibility, social signals).

MODEL PERFORMANCES

Logistic Regression

- **Accuracy**: 98.81%
- **F1-Score**: 0.99 (macro and weighted)
- Performs very well with high precision and recall.

Random Forest

- **Accuracy**: 99.68%
- **F1-Score**: 1.00
- Achieved perfect classification on the test set.

Naive Bayes

- **Accuracy**: 93.40%
- **F1-Score**: 0.93
- Performed decently, but not as well as the others.

SVM (Support Vector Machine)

- **Accuracy**: 99.38%
- **F1-Score**: 0.99
- Strong performance similar to Logistic Regression.

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	98.81%	0.99	0.99	0.99
Random Forest	99.68%	1.00	1.00	1.00
Naive Bayes	93.40%	0.93	0.93	0.93
Support Vector Machine (SVM)	99.38%	0.99	0.99	0.99

Best Performing Model: Random Forest

CONCLUSION

This project focused on detecting fake news using machine learning and deep learning techniques. After cleaning and preprocessing over 44,000 news articles, various models were trained and tested. Random Forest achieved the highest accuracy at 99.68%, with SVM and Logistic Regression also performing well. An LSTM-based deep learning model reached a strong 98.10% validation accuracy. The results demonstrate that both traditional and neural models can effectively classify news as real or fake, making this system suitable for practical applications and future enhancements.

GOOGLE COLAB LINK: TEXT PROJECT

https://colab.research.google.com/drive/1PCL7AotqzULjg3UXgKdHa19XDGM6gQ0E#scrollTo=gGd-IYQwDNsq