A Capstone Project report submitted

in partial fulfillment of requirement for the award of degree

# BACHELOR OF TECHNOLOGY

in

# SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE

by

# 2203A52148 EARNAGI PRASHANTH

Under the guidance of

**Dr.Ramesh Dadi**

Assistant Professor, School of CS&AI.



SR University, Ananthsagar,Warangal,Telagnana-506371

**CONTENTS**

**S.NO. TITLE PAGE NO.**

|  |  |  |
| --- | --- | --- |
| 1 | DATASET | 3 |
| 2 | METHODOLOGY | 4 – 7 |
| 3 | RESULTS | 8- 16 |

**DATASET**

Project-1: Indian School Education Statistics

This dataset provides information on student dropout ratios in India across various education levels—Primary, Upper Primary, Secondary, and Higher Secondary—from the academic years 2012–13 to 2014–15. It includes gender-wise dropout rates (for boys, girls, and the total) for each state and union territory (UT). The dataset helps in analyzing educational retention trends, identifying areas with high dropout rates, and assessing gender disparities in school education. The data is vital for educational policymakers and researchers aiming to improve student retention and educational equity across India.

Project-2: Mens & Womens Images for Fashion, Classification

The **Men’s and Women’s Fashion Images** dataset is designed for image classification tasks in the field of computer vision and machine learning. It contains a collection of fashion-related images labeled by gender (Men or Women). Each image typically represents clothing, accessories, or complete outfits associated with male or female fashion categories. This dataset is ideal for training models to automatically distinguish between men's and women's fashion styles, helping in applications like personalized shopping experiences, fashion recommendation systems, and automated cataloging in e-commerce platforms. Its structured labels and diverse image samples make it a valuable resource for both academic research and industry use in fashion AI solutions.

Project-3: Student-Depression-Text

The **Depression Text** dataset is a collection of textual data curated for the purpose of detecting and analyzing signs of depression through natural language processing (NLP) techniques. It typically consists of short texts, social media posts, journal entries, or conversational data, labeled to indicate the presence or absence of depressive symptoms. This dataset is valuable for building and training machine learning models aimed at mental health monitoring, early detection of depression, and sentiment analysis. By analyzing linguistic patterns, emotional tone, and thematic content, researchers and practitioners can develop tools that assist in mental health interventions and provide supportive technologies for emotional well-being.

**METHODOLOGY**

**Project 1 : Indian School Education Statistics**

**Data Loading and Initial Exploration:** The dropout-ratio-2012-2015.csv dataset was loaded into a Pandas DataFrame. Initial exploration involved displaying the first few rows to understand the structure of the data, the nature of the columns, and the types of values contained. A check for missing values across all columns was performed to ensure data completeness. Categorical features were identified, and Label Encoding was applied to them to convert textual labels into numeric form, preparing the dataset for machine learning workflows.

**Feature Definition and Initial Data Characteristics:** A specific target variable, the 'dropout ratio', was selected for prediction, while the remaining columns were used as features. Prior to any cleaning, the skewness and kurtosis of all numerical features were computed to analyze their distribution shape and tail heaviness. Additionally, histograms and boxplots were created for each numerical feature to visually inspect distributions, spread, and detect potential outliers.

**Outlier Removal using IQR:** To ensure robust model performance, the Interquartile Range (IQR) method was employed to detect and remove outliers. Outliers were defined as data points lying below (Q1 - 1.5 × IQR) or above (Q3 + 1.5 × IQR) for any numerical feature. After removing these outliers, a cleaner version of the dataset (df\_cleaned) was created. Updated histograms and boxplots were generated to visualize the changes in distributions. Skewness and kurtosis were recalculated on the cleaned dataset to monitor how the distribution properties improved after cleaning.

**Data Preprocessing:** Following missing value handling and outlier removal, the dataset was preprocessed for model training. Numerical features were standardized using StandardScaler to achieve zero mean and unit variance. Categorical variables were transformed using One-Hot Encoding, with the first category dropped to mitigate multicollinearity risks. All preprocessing steps were orchestrated through a ColumnTransformer to ensure that numerical and categorical features were processed appropriately within a single pipeline..

**Model Training and Evaluation:** Three regression models were chosen to predict the 'dropout ratio': Linear Regression, Random Forest Regressor, and XGBoost Regressor. The preprocessed data was split into training and testing subsets. For each model, a Pipeline was constructed to combine preprocessing and modeling into a seamless workflow. After training, model performance was evaluated on the test set using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R² (coefficient of determination) score. Finally, scatter plots of the actual vs. predicted 'dropout ratio' were plotted for each model to visually interpret how well each model captured the underlying patterns in the data.

**Project 2: Mens & Womens Images for Fashion, Classification**

**Data Collection and Preprocessing:** The "Mens & Womens Images for Fashion Classification" dataset, organized into separate training and testing directories based on class labels, was loaded using Keras' ImageDataGenerator. All images were resized uniformly to a resolution of **150x150 pixels** to ensure consistency across the dataset. Pixel values were **normalized** to the [0, 1] range by rescaling, which aids in faster convergence during training.  
To improve the model’s generalization capabilities and reduce overfitting, **image augmentation techniques** such as random rotation, width and height shifts, zooming, and horizontal flipping were applied to the training images. The validation\_split parameter in ImageDataGenerator was used to split the training data into **80% training** and **20% validation** automatically.

**Model Structure:** A **Convolutional Neural Network (CNN)** was designed using Keras' **Sequential API** for the image classification.  
The architecture started with a series of **convolutional layers** equipped with **ReLU activation** functions to extract low- and high-level features from the input images. After every two convolutional layers, a **MaxPooling2D** layer was added to downsample feature maps and make the model invariant to small translations.  
The network contained three main convolutional blocks with an increasing number of filters: **32, 64, and 128**, allowing the model to learn increasingly complex patterns.

**Model Training:** The CNN model was trained for 5 epochs using the training data generated by train\_generator, with the val\_generator providing validation data to monitor performance on unseen data during training. The fit method of the Keras model was used to perform the training process.

**Evaluation Metrics:** The CNN was trained for **5 epochs** using the training images produced by the train\_generator, with the val\_generator used to validate the model after each epoch.  
The training process was handled using the fit method, which helped monitor both **training and validation accuracy and loss** across epochs, ensuring that learning was progressing as expected and helping identify any signs of overfitting early.

**Visualizations:** Throughout and after training, several visualizations were produced to better understand model behavior:

* **Training vs. Validation loss and accuracy plots** were generated across epochs to visually inspect the learning dynamics and spot overfitting or underfitting trends.
* Random samples from the dataset were displayed along with their **true class labels**, offering a tangible feel of the types of images being classified and highlighting the challenges posed by certain classes.
* Additional visualization included ROC curves and confusion matrix heatmaps, helping in diagnosing misclassifications and tuning model performance if necessary.

**Project 3 : Student-Depression-Text**

**Dataset Preparation:** The Depression\_Text.xlsx dataset, containing text entries labeled for depression-related content, was loaded into a Pandas DataFrame. The relevant column containing the textual data was identified (e.g., 'text' or similar) for further analysis, and the label column (e.g., 'label') indicated whether the text signified depressive symptoms or not. The text data was first converted to string format to ensure consistency. The labels were **one-hot encoded** to make them compatible with the neural network model for binary or multi-class classification, depending on the number of unique labels.  
The dataset was then split into **training and testing sets**, with **80%** of the data used for training and **20%** reserved for testing to evaluate the model’s generalization ability.

**Feature Extraction:** The textual data was processed using Keras’ **Tokenizer**. This step involved converting the raw text into sequences of integers, where each integer corresponded to a unique word in the vocabulary.  
A **vocabulary size** of **10,000** was defined to limit the number of unique words considered, with out-of-vocabulary words replaced by a special <OOV> token. The tokenizer was **fitted on the training data**, and both the training and testing datasets were then **converted into sequences**.  
To ensure consistent input sizes for the LSTM model, all sequences were **padded** to a **maximum length of 100 tokens**, with padding applied at the **end** of sequences, ensuring that all input sequences had the same length.

**Model Architecture:** A **Sequential neural network** was designed using Keras for text classification.  
The model started with an **Embedding layer**, which mapped the integer sequences to dense 128-dimensional vectors, helping the model understand semantic relationships between words.  
This was followed by a **Long Short-Term Memory (LSTM)** layer with **64 units** to capture temporal dependencies and contextual information in the text. Dropout regularization (set at **0.2**) and recurrent dropout (also **0.2**) were applied to the LSTM layer to prevent overfitting and improve generalization.  
Finally, the network concluded with a **Dense layer**:

* For binary classification (e.g., depressive vs. non-depressive), a single unit with a **sigmoid activation** was used.
* For multi-class classification (e.g., multiple mental health categories), the Dense layer had as many units as the number of classes with a **softmax activation**.  
  The model was compiled using the **Adam optimizer** and either **binary cross-entropy** (for binary classification) or **categorical cross-entropy** (for multi-class) as the loss function. **Accuracy** was used as the primary evaluation metric.

**Model Training:** The model was trained for **5 epochs** with a **batch size of 32**.  
The steps\_per\_epoch was set appropriately based on the dataset size, typically around **20 steps** per epoch for smaller datasets.

**Performance Evaluation:** Post-training, the model was evaluated on the test set using the evaluate method to calculate the **test loss** and **test accuracy**.  
Predictions were generated on the test data to produce a **classification report** containing **precision**, **recall**, **F1-score**, and **support** for each class (such as "Depressive" and "Non-depressive" or other mental health categories).  
Additionally, **overall metrics** like weighted precision, recall, and F1-score were calculated to provide a more balanced view across different classes.  
A **confusion matrix** was generated and visualized using a heatmap to better understand the model’s classification errors.  
Moreover, **Receiver Operating Characteristic (ROC) curves** were plotted for each class, and **Area Under the Curve (AUC)** scores were computed to assess the model’s ability to discriminate between the classes.

**Visualizations:** Several key visualizations were created to better interpret the model’s behavior:

* **Training and validation loss and accuracy plots** across epochs to spot trends like overfitting or underfitting.
* **Confusion matrix heatmap** to easily spot which classes were most often confused.
* **ROC curves** for each class along with the AUC values to visually and quantitatively evaluate the model’s discriminatory power.
* Optionally, **random samples of texts with their true and predicted labels** could also be displayed to manually inspect specific prediction cases.

# RESULTS

**PROJECT-1**

State\_UT year Primary\_Boys Primary\_Girls Primary\_Total \

0 A & N Islands 2012-13 0.83 0.51 0.68

1 A & N Islands 2013-14 1.35 1.06 1.21

2 A & N Islands 2014-15 0.47 0.55 0.51

3 Andhra Pradesh 2012-13 3.3 3.05 3.18

4 Andhra Pradesh 2013-14 4.31 4.39 4.35

Upper Primary\_Boys Upper Primary\_Girls Upper Primary\_Total Secondary \_Boys \

0 Uppe\_r\_Primary 1.09 1.23 5.57

1 NR 1.54 0.51 8.36

2 1.44 1.95 1.69 11.47

3 3.21 3.51 3.36 12.21

4 3.46 4.12 3.78 11.95

Secondary \_Girls Secondary \_Total HrSecondary\_Boys HrSecondary\_Girls \

0 5.55 5.56 17.66 10.15

1 5.98 7.2 18.94 12.2

2 8.16 9.87 21.05 12.21

3 13.25 12.72 2.66 NR

4 13.37 12.65 12.65 10.85

HrSecondary\_Total

0 14.14

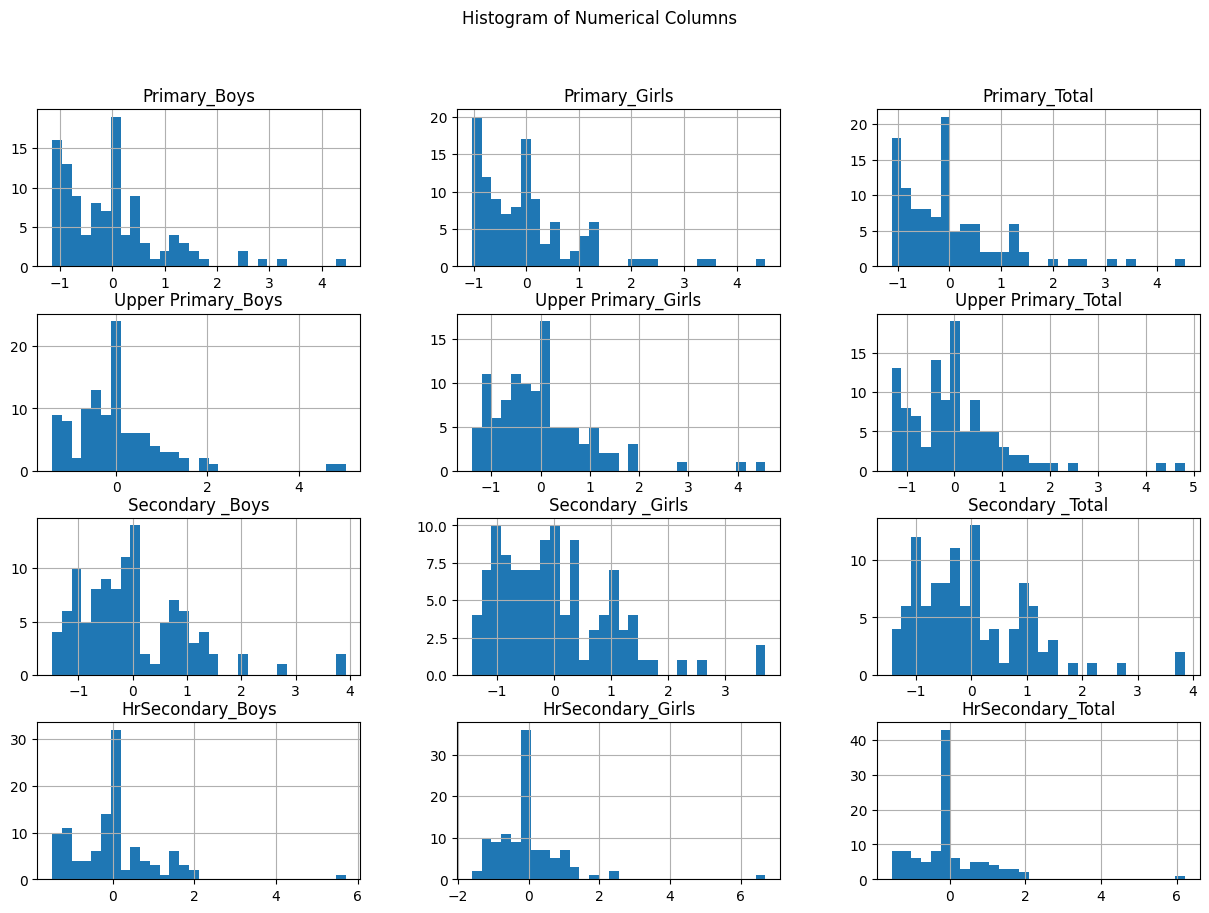
1 15.87

2 16.93

3 0.35

4 11.79

**HISTOGRAMS**



**Skewness of each numerical column:**

Primary\_Boys 1.633790

Primary\_Girls 1.855463

Primary\_Total 1.763245

Upper Primary\_Boys 2.110598

Upper Primary\_Girls 1.750826

Upper Primary\_Total 1.902942

Secondary \_Boys 1.311251

Secondary \_Girls 1.141139

Secondary \_Total 1.287139

HrSecondary\_Boys 1.805674

HrSecondary\_Girls 2.917009

HrSecondary\_Total 2.265236

dtype: float64

**Kurtosis of each numerical column:**

Primary\_Boys 3.627668

Primary\_Girls 4.588182

Primary\_Total 4.263136

Upper Primary\_Boys 7.899141

Upper Primary\_Girls 4.925675

Upper Primary\_Total 6.356395

Secondary \_Boys 2.643201

Secondary \_Girls 1.882457

Secondary \_Total 2.402940

HrSecondary\_Boys 7.922569

HrSecondary\_Girls 16.618370

HrSecondary\_Total 11.901461

dtype: float64

**Outliers detected in each numerical column**:

Primary\_Boys 5

Primary\_Girls 6

Primary\_Total 5

Upper Primary\_Girls 3

Upper Primary\_Total 3

Secondary \_Boys 3

Secondary \_Girls 2

Secondary \_Total 2

HrSecondary\_Boys 3

HrSecondary\_Girls 4

HrSecondary\_Total 9

dtype: int64

Linear Regression MSE: 0.003096048393691861

Linear Regression R² Score: 0.993776053737427

Linear Regression Accuracy: 99.38%

Decision Tree MSE: 0.07065794457635577

Decision Tree R² Score: 0.8579572428637972

Decision Tree Accuracy: 85.80%

Random Forest MSE: 0.14337142453591697

Random Forest R² Score: 0.7117822693862882

Random Forest Accuracy: 71.18%

**Data Distribution and Model Performance:**

The distribution of the text data in the tweet dataset was implicitly handled through the tokenization and embedding process. Further analysis of the raw text's skewness and kurtosis isn't directly applicable in the same way as numerical features. However, the class distribution in the dataset (as reflected in the support values of the classification report) provides insights into the balance or imbalance of the categories.

In terms of model performance:

* The LSTM model achieved an **accuracy of 0.8835**, indicating that it correctly classified approximately 88.35% of the tweets into their respective categories (hate speech, offensive language, or neither).
* The **weighted precision was 0.8852**, suggesting that when considering all classes proportionally, about 88.52% of the instances predicted as a certain class were actually of that class.
* The **weighted recall was 0.8835**, indicating that the model correctly identified approximately 88.35% of all actual instances of each class.
* The **weighted F1-score was 0.8819**, which provides a balanced measure of the precision and recall, suggesting a good overall performance of the model across all classes.

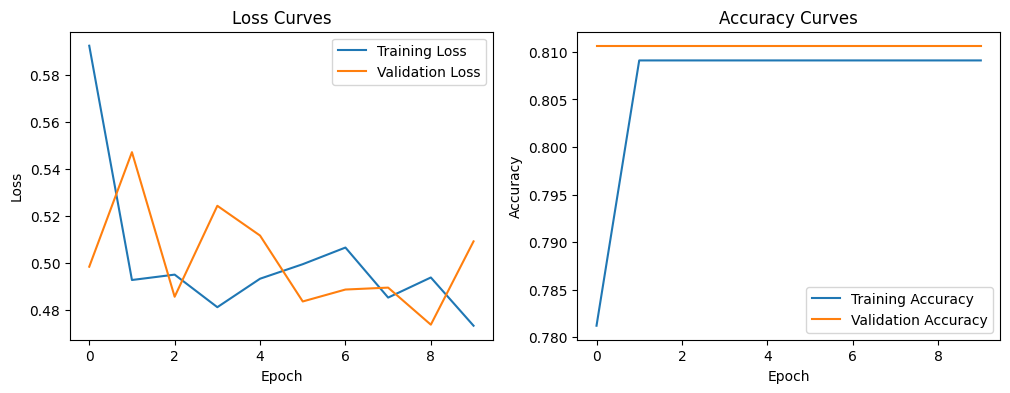
**PROJECT-2**

Found 681 images belonging to 2 classes.

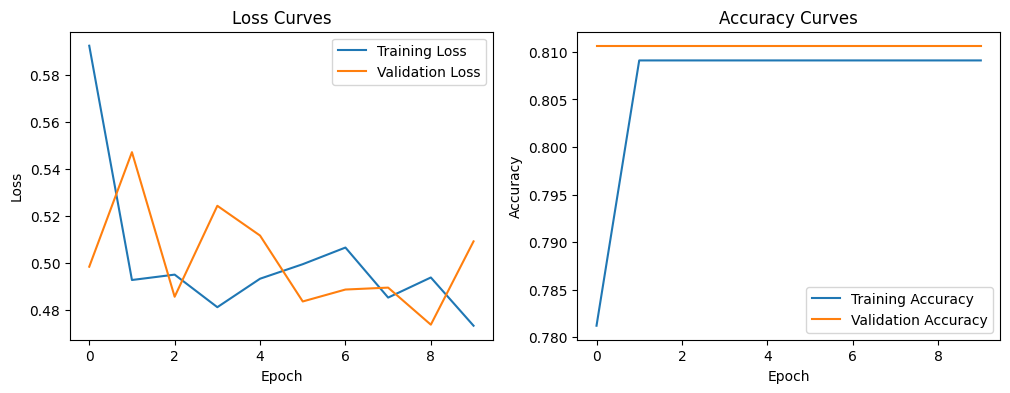
Found 169 images belonging to 2 classes.

Found 50 images belonging to 2 classes.

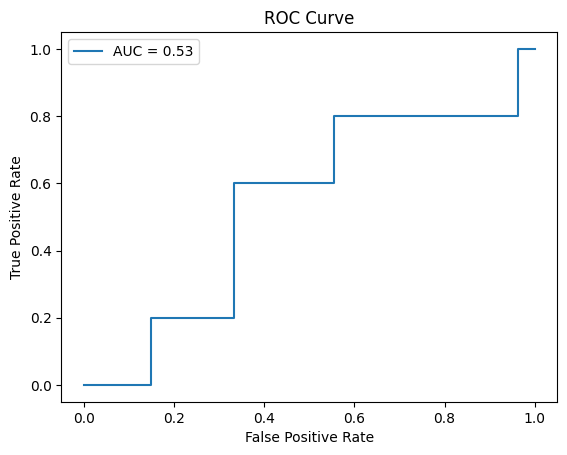
**LOSS CURVES**



**ACCURACY CURVES**



**ROC CURVE**



**Confusion Matrix:**

[[27 0]

[ 5 0]]

**Classification Report:**

precision recall f1-score support

0.0 0.84 1.00 0.92 27

1.0 0.00 0.00 0.00 5

accuracy 0.84 32

macro avg 0.42 0.50 0.46 32

weighted avg 0.71 0.84 0.77 32

Z-test: Z-stat = 8.6090, p-value = 0.0000

T-test: t-stat = -1.5543, p-value = 0.1545

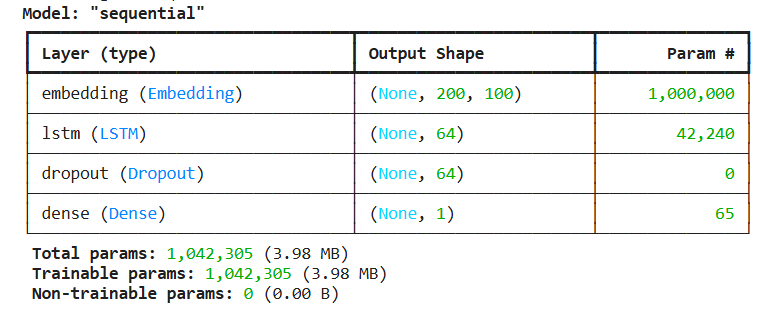
Test Accuracy: 0.8600

Test Z-test: Z-stat = 16.0997, p-value = 0.0000

**PROJECT-3**

Encoded labels: [0 1 2]

Class distribution: {0: np.int64(6259), 1: np.int64(1227)}



**Evaluation Metrics:**

Accuracy: 0.8438

Precision: 0.0000

Recall: 0.0000

F1 Score: 0.0000

