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

2203A52133 **AKULA TEJDEEP**

Under the guidance of

Dr.Ramesh Dadi

Assistant Professor, School of CS&AI.

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE

SR UNIVERSITY, ANANTHASAGAR, WARANGAL, TELANGANA

April, 2025.

CONTENTS

DATASET

Project-1: FIFA World Cup

The FIFA World Cup is the most prestigious and widely viewed sporting event in the world, bringing together nations in a celebration of football, culture, and global unity. Held every four years, the tournament showcases elite talent, historic rivalries, and unforgettable moments that shape the legacy of the sport. This report explores key data from recent editions of the FIFA World Cup, spanning from 2006 to 2022. It highlights host nations, participating teams, champions, runners-up, top scorers, attendance figures, and match statistics. By analyzing this data, we aim to uncover trends and patterns that offer insights into the evolution of the tournament, its growing popularity, and its growing popularity, and its cultural impact on a global scale.

Project-2: Blood Cancer

Blood cancer, also known as hematologic cancer, originates in the bone marrow and affects the production and function of blood cells. Among the most common types are Acute Lymphoblastic Leukemia (ALL) and Acute Myeloid Leukemia (AML), both of which rapidly progress and require prompt diagnosis and treatment. Early and accurate detection of blood cancers is crucial to improving patient survival rates.

This report explores the classification of blood cancer subtypes using a dataset comprising microscopic images of blood samples. The dataset includes four distinct categories: Acute Lymphoblastic Leukemia (ALL), Acute Myeloid Leukemia (AML), other hematologic conditions (HEM), and healthy individuals. Leveraging machine learning and deep learning techniques on this data can significantly enhance the diagnostic process, supporting medical professionals in identifying cancerous conditions more efficiently and accurately.

The goal of this project is to develop a reliable classification model that can distinguish between these categories based on image features, thereby contributing to the broader objective of automated medical diagnostics.

Project-3: Spam Email

In the digital age, email remains one of the most essential communication tools for both personal and professional use. However, with the widespread use of email comes the persistent challenge of unsolicited and malicious messages, commonly known as spam. Spam emails can clutter inboxes, waste time, and in many cases, pose serious security risks such as phishing attacks or malware distribution.

This report focuses on the analysis and classification of email data to distinguish between spam and legitimate (ham) messages. Using a dataset of labeled emails, machine learning techniques can be applied to learn patterns in email content, subject lines, and metadata. The ultimate goal is to develop an intelligent spam detection model that can accurately filter out unwanted messages and improve overall email security and user experience.

Through exploratory data analysis, feature extraction, and model training, this project demonstrates how data-driven approaches can be used to automate email classification and contribute to safer digital communication environments.

METHODOLOGY

Project 1: FIFA World Cup

1. Data Collection:

The dataset used in this analysis was obtained from a structured CSV file containing records of FIFA World Cup tournaments from 2006 to 2022. It includes key information such as host countries, winning and runner-up teams, total goals, attendance figures, matches played, and top scorers.

2. Data Preprocessing:

Initial preprocessing steps included:

- Loading the dataset into a data analysis environment.
- Handling missing values and correcting inconsistencies in team names and tournament years.
- Converting relevant columns to appropriate data types (e.g., numerical values for goals, matches, and attendance).

3. Exploratory Data Analysis (EDA):

A thorough EDA was conducted to understand trends and distributions across different tournaments. This involved:

- Analyzing winning patterns and the frequency of appearances by top teams.
- Visualizing goal statistics, match outcomes, and attendance over the years.
- Identifying standout players through top scorer data.

4. Model Training:

Three machine learning models—Linear Regression, Decision Tree Regressor, and Random Forest Regressor—were trained using the preprocessed data. The models were evaluated on a test set using performance metrics such as RMSE (Root Mean Squared Error) and R^2 (coefficient of determination).

5. Performance Measurement:

The models' performances were compared using RMSE and R^2 scores, highlighting their ability to predict housing prices. Additionally, skewness and kurtosis values were included in the model comparison to evaluate the impact of the dataset's distribution on model performance.

This methodology provided a structured approach for understanding and predicting housing prices using different machine learning models, ensuring a clear evaluation of each model's effectiveness.

6. Tools and Technologies

The analysis was carried out using Python, with the help of libraries such as:

- Pandas for data manipulation
- Matplotlib and Seaborn for visualizations
- NumPy for numerical computations

Project 2: Blood Cancer

1. Data Preparation

- **Dataset Structure**: The dataset consists of labeled microscopic blood smear images categorized into cancerous and non-cancerous classes.
- **Image Preprocessing**: Each image was resized to 128×128 pixels and normalized to scale pixel values between 0 and 1.
- **Labeling**: Cancerous images were labeled as 1 and non-cancerous images as 0.
- **Splitting**: The combined dataset was split into training and testing sets with an 80-20 ratio, using a random state to ensure reproducibility.

2. Model Architecture

A CNN was constructed using the TensorFlow/Keras framework with the following layers:

- **Convolutional Layers**: Two Conv2D layers (32 and 64 filters respectively) for feature extraction with ReLU activation.
- **Pooling Layers**: MaxPooling layers to reduce spatial dimensions.
- **Fully Connected Layers**: A Flatten layer followed by a Dense layer (64 units, ReLU) and an output layer (1 unit, sigmoid) for binary classification.

3. Training Configuration

- **Loss Function**: Binary Crossentropy
- **Optimizer**: Adam
- **Batch Size**: 32
- **Epochs**: 10
- **Validation Split**: 20% of the training data was used for validation during training.

4. Performance Evaluation

- **Accuracy & Loss Curves**: Plotted across epochs for training and validation sets.
- **Test Accuracy**: Final performance on the test set was evaluated using accuracy.
- **Confusion Matrix**: Visualized correct and incorrect predictions.
- **Classification Report**: Included precision, recall, F1-score, and support for each class.
- **ROC Curve and AUC**: Assessed the model's ability to discriminate between classes across thresholds.

5. Statistical Analysis

To further validate model predictions, the following statistical tests were applied:

- **T-Test**: Compared predicted probabilities for cancerous vs. non-cancerous images.
- **Z-Test**: Evaluated whether the model's accuracy significantly exceeds a baseline (e.g., 50%).
- **ANOVA (F-test)**: Analyzed variance in prediction probabilities split across confidence ranges (low, medium, high).

6. Tools and Libraries

- **TensorFlow/Keras**: Deep learning framework
- **Matplotlib**: Data visualization
- **NumPy**: Numerical operations
- **SciPy & scikit-learn**: Statistical testing and evaluation

Project 3: Spam Email

1. Dataset Collection and Preprocessing

The dataset used in this study consists of labeled emails indicating whether each email is spam or not. The dataset was read from a CSV file (emails.csv) and preprocessed for natural language processing tasks. Preprocessing involved:

- **Lowercasing** all text for normalization.
- **Removing punctuation and non-alphabetic characters** using regular expressions.
- **Tokenizing** the cleaned text into individual words using nltk's word_tokenize.

Each email was associated with a binary label: 1 for spam and 0 for non-spam (ham).

2. Data Splitting

The data was split into training and testing sets using an 80/20 ratio with stratified random sampling to ensure balanced class distribution across sets.

3. Word Embedding with Word2Vec

A **Word2Vec** model was trained on the tokenized training dataset using the gensim library. The parameters included:

- Vector size: 100
- Context window: 5
- Minimum word frequency: 1
- Workers: 4 (for parallelization)

Each tokenized email was transformed into a fixed-length sequence of word vectors. If a sentence had fewer tokens than the predefined max_len (50), it was padded with zero vectors. If longer, it was truncated.

4. Dataset Preparation for PyTorch

The email data was converted into a PyTorch Dataset class to be used with a DataLoader for batch training. Each data point consisted of:

- A 50×100 matrix (for 50 words, each with 100-dimensional vector)
- A corresponding binary label (spam or not)

5. Model Architecture

An **LSTM-based binary classifier** was implemented using PyTorch. The architecture consisted of:

- A single-layer **LSTM** taking 100-dimensional word vectors as input
- A fully connected layer projecting the LSTM output to a scalar
- A **sigmoid activation** to squash the output between 0 and 1 for binary classification

6. Training Process

The model was trained for 10 epochs with the following setup:

- Loss function: **Binary Cross-Entropy Loss**
- Optimizer: **Adam** with a learning rate of 0.001
- Batch size: 32

At each epoch, training loss and accuracy were calculated and logged. Accuracy was determined by thresholding predictions at 0.5 and comparing them with true labels.

7. Performance Visualization

Training loss and accuracy were plotted over epochs to monitor the learning process and ensure the model was converging.

RESULTS

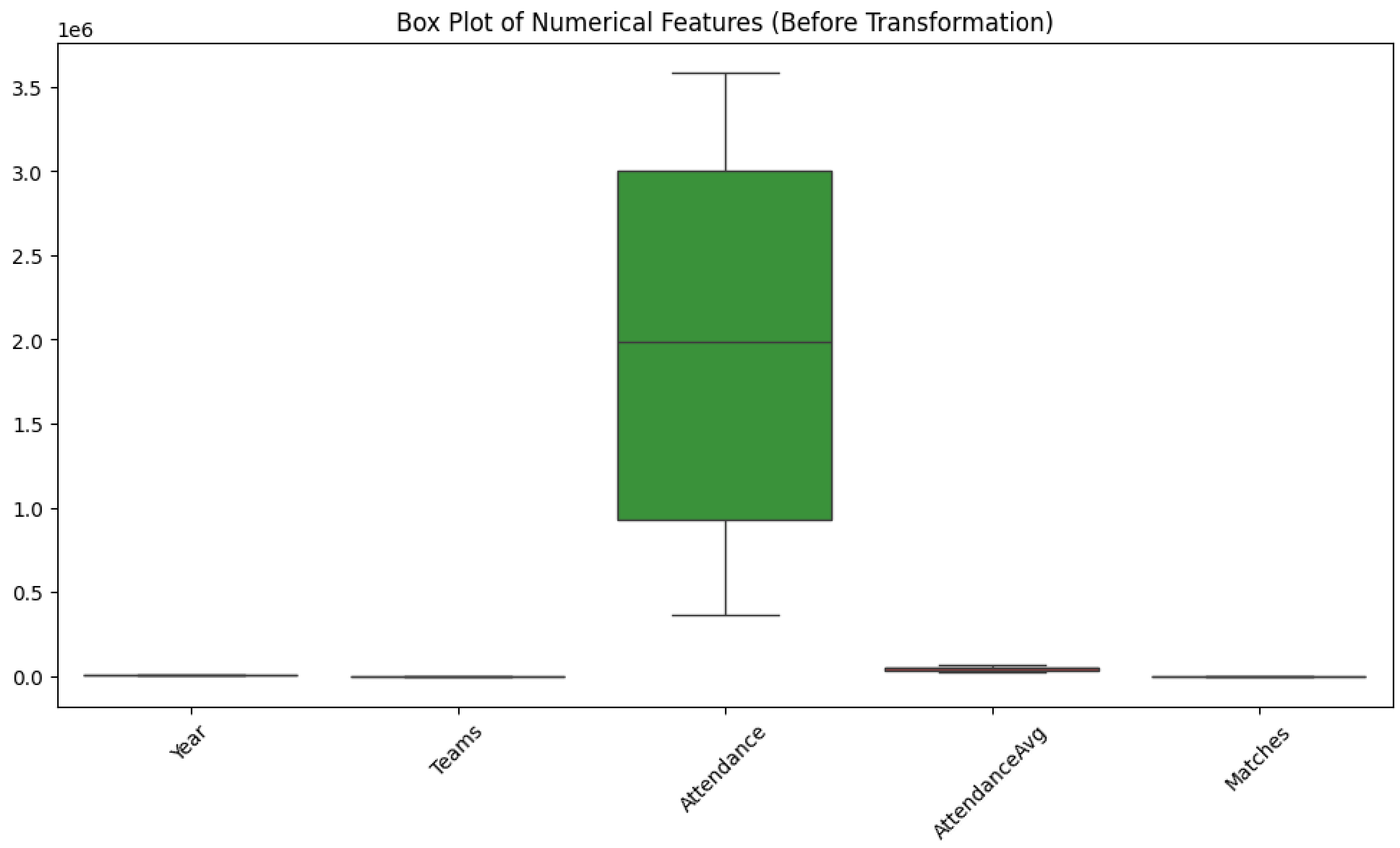
PROJECT-1

Skewness before transformation:

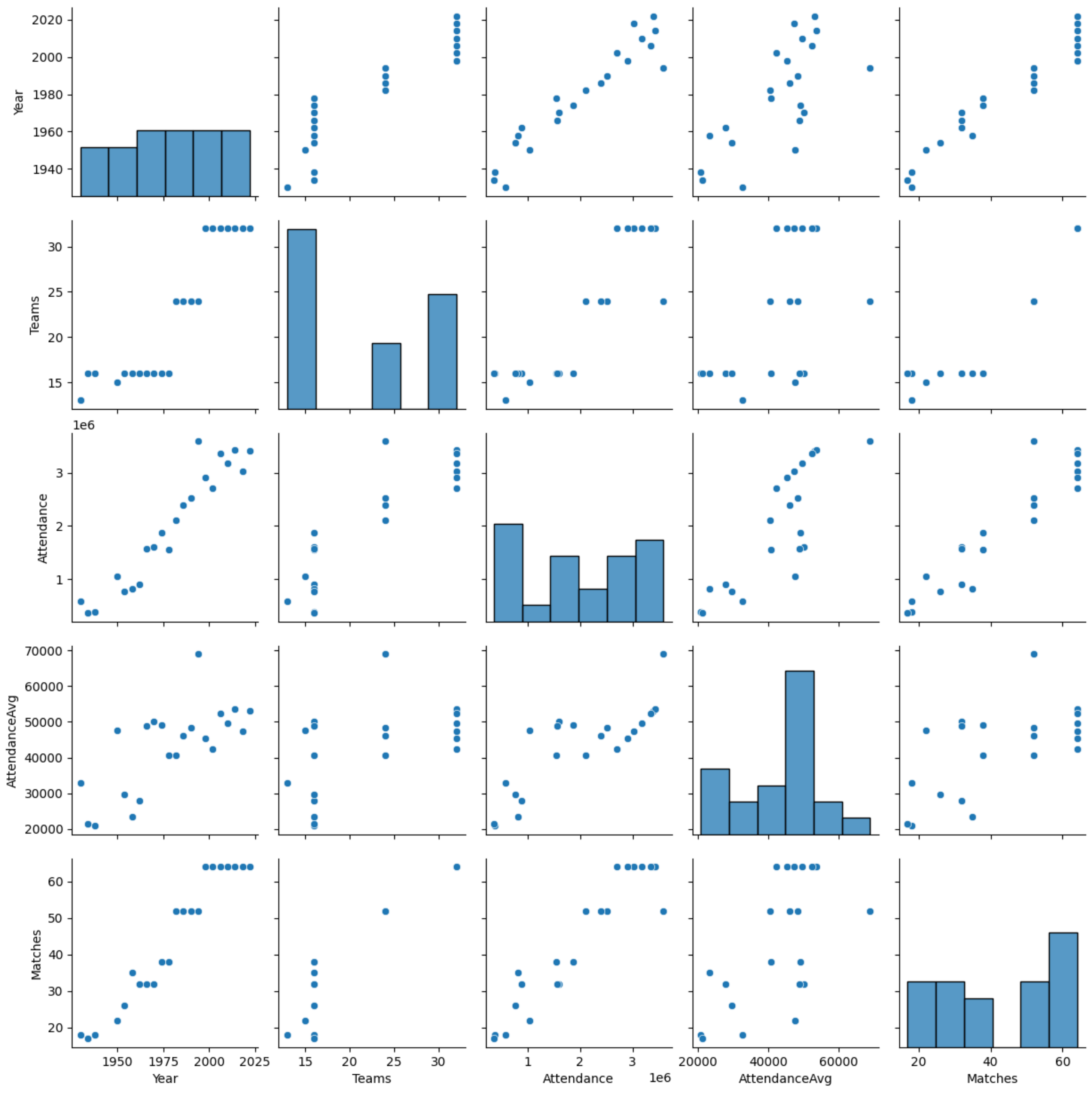
```
 Year           -0.171851
Teams            0.322539
Attendance      -0.051976
AttendanceAvg   -0.294979
Matches         -0.142693
dtype: float64
```

Skewness after transformation:

```
 Year           -0.190974
Teams            0.184794
Attendance      -0.751878
AttendanceAvg   -0.864754
Matches         -0.541151
dtype: float64
```



HISTOGRAMS



Outliers detected:

```
Year             0
Teams            0
Attendance       0
AttendanceAvg    0
Matches          0
dtype: int64
```

Linear Regression Performance:
MAE: 0.0007383028606437137
MSE: 6.748664961551653e-07
R2 Score: 0.9999971942578311

Decision Tree Performance:
MAE: 0.25904245803118686
MSE: 0.16364454833850475
R2 Score: 0.31965149759351696

Random Forest Performance:
MAE: 0.09090128146466903
MSE: 0.018440681546609626
R2 Score: 0.9233332842372562

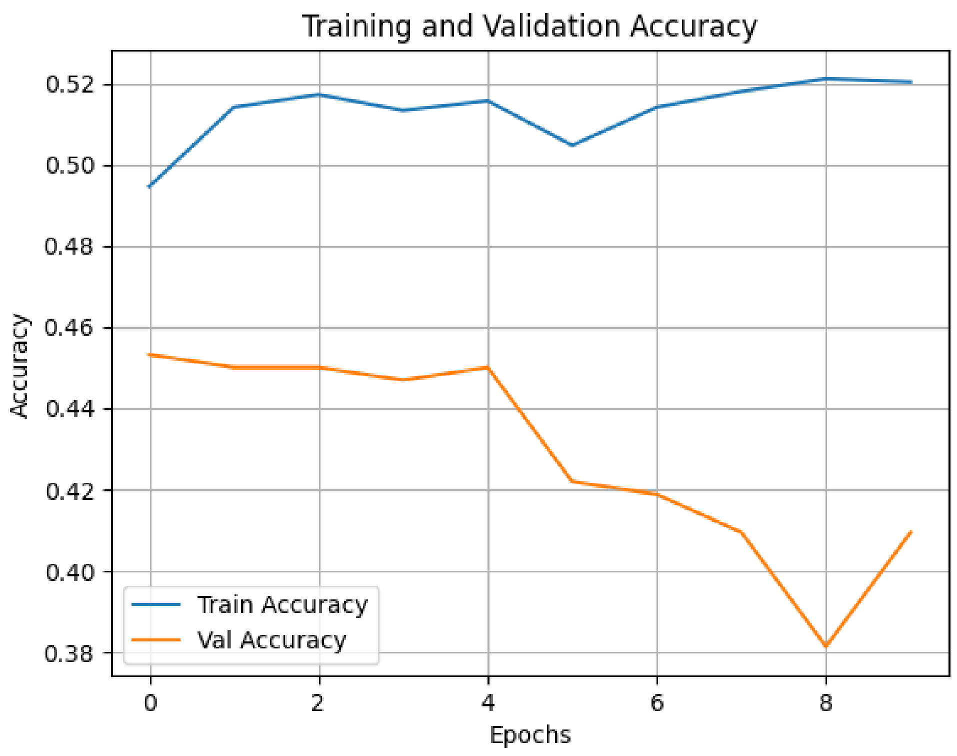
The dataset exhibited **moderate skewness** in features like price, area, and bathrooms, indicating slight asymmetry in their distributions. **Kurtosis** values suggest that the features mostly have near-normal or slightly flatter distributions.

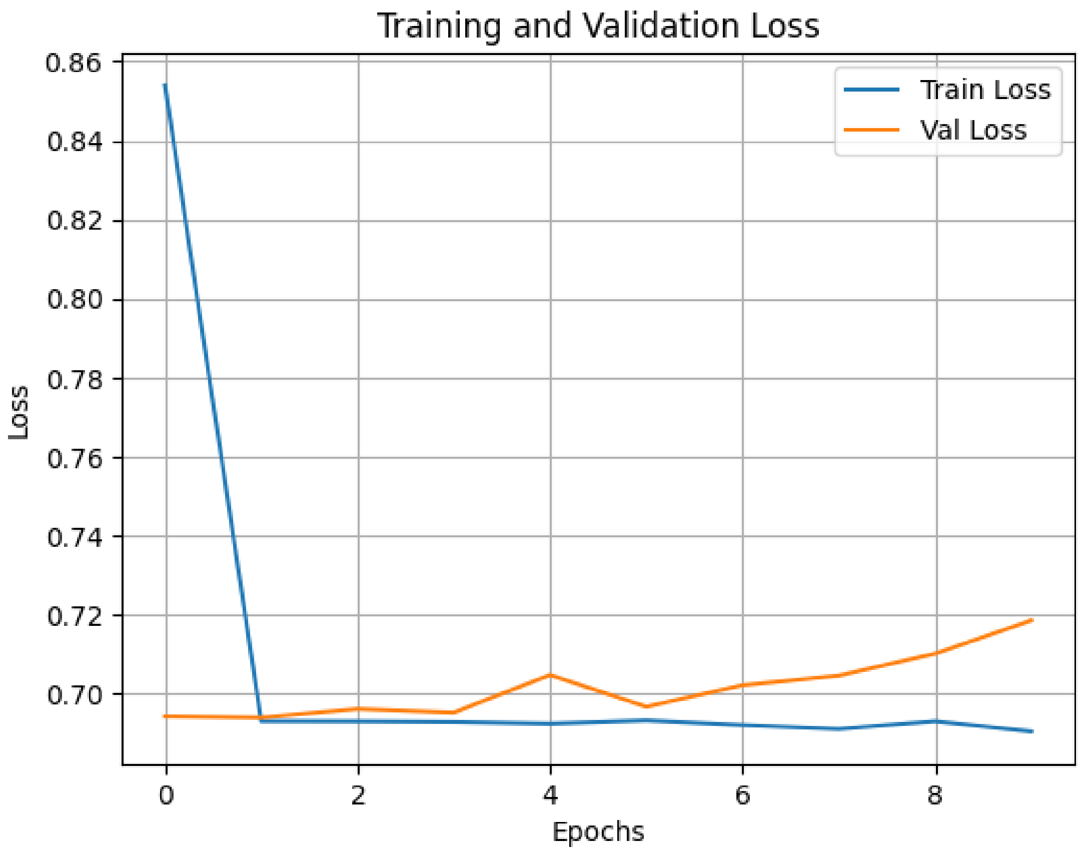
In terms of model performance:

- **Linear Regression** performed best overall with the **lowest RMSE (1.27M)** and **highest R^2 score (0.55)**, indicating it explained about 55% of the variance in house prices.
- **Random Forest** came next with a slightly higher RMSE and lower R^2 (0.44).
- **Decision Tree** performed the worst, with the **highest RMSE (1.77M)** and lowest R^2 (0.13), suggesting poor generalization.

PROJECT-2

```
Epoch 1/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 3s 30ms/step - accuracy: 0.4882 - loss: 1.1882 -
val_accuracy: 0.4531 - val_loss: 0.6943
Epoch 2/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 16ms/step - accuracy: 0.5123 - loss: 0.6931 -
val_accuracy: 0.4500 - val_loss: 0.6940
Epoch 3/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 16ms/step - accuracy: 0.5183 - loss: 0.6929 -
val_accuracy: 0.4500 - val_loss: 0.6962
Epoch 4/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 16ms/step - accuracy: 0.5156 - loss: 0.6928 -
val_accuracy: 0.4469 - val_loss: 0.6953
Epoch 5/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.4960 - loss: 0.6932 -
val_accuracy: 0.4500 - val_loss: 0.7048
Epoch 6/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.5005 - loss: 0.6940 -
val_accuracy: 0.4219 - val_loss: 0.6967
Epoch 7/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.5549 - loss: 0.6922 -
val_accuracy: 0.4187 - val_loss: 0.7022
Epoch 8/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.5360 - loss: 0.6900 -
val_accuracy: 0.4094 - val_loss: 0.7046
Epoch 9/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.5265 - loss: 0.6908 -
val_accuracy: 0.3812 - val_loss: 0.7101
Epoch 10/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 1s 16ms/step - accuracy: 0.5165 - loss: 0.6902 -
val_accuracy: 0.4094 - val_loss: 0.7185
```



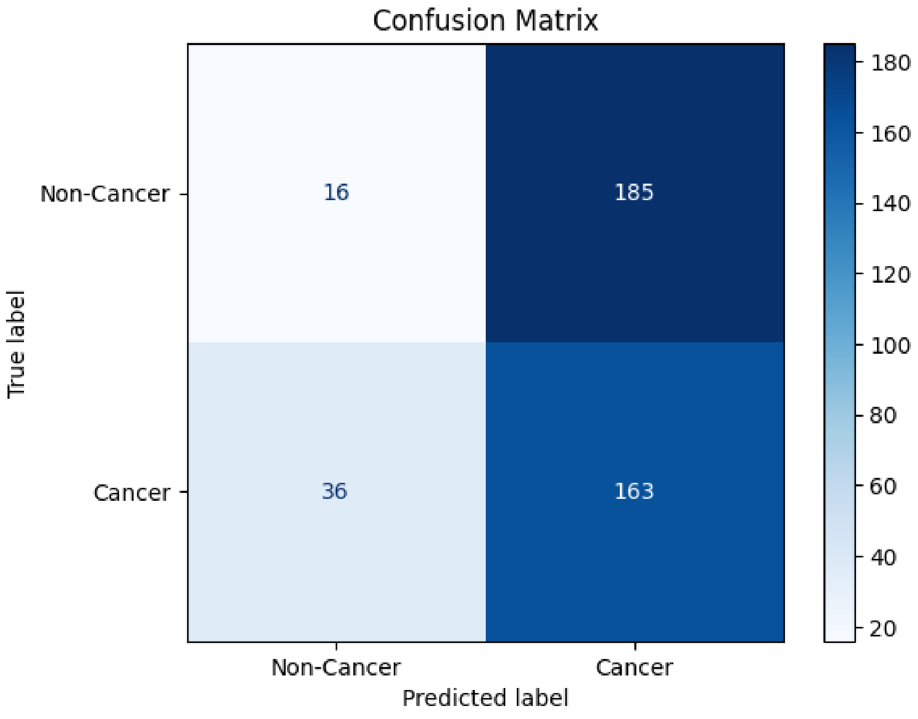


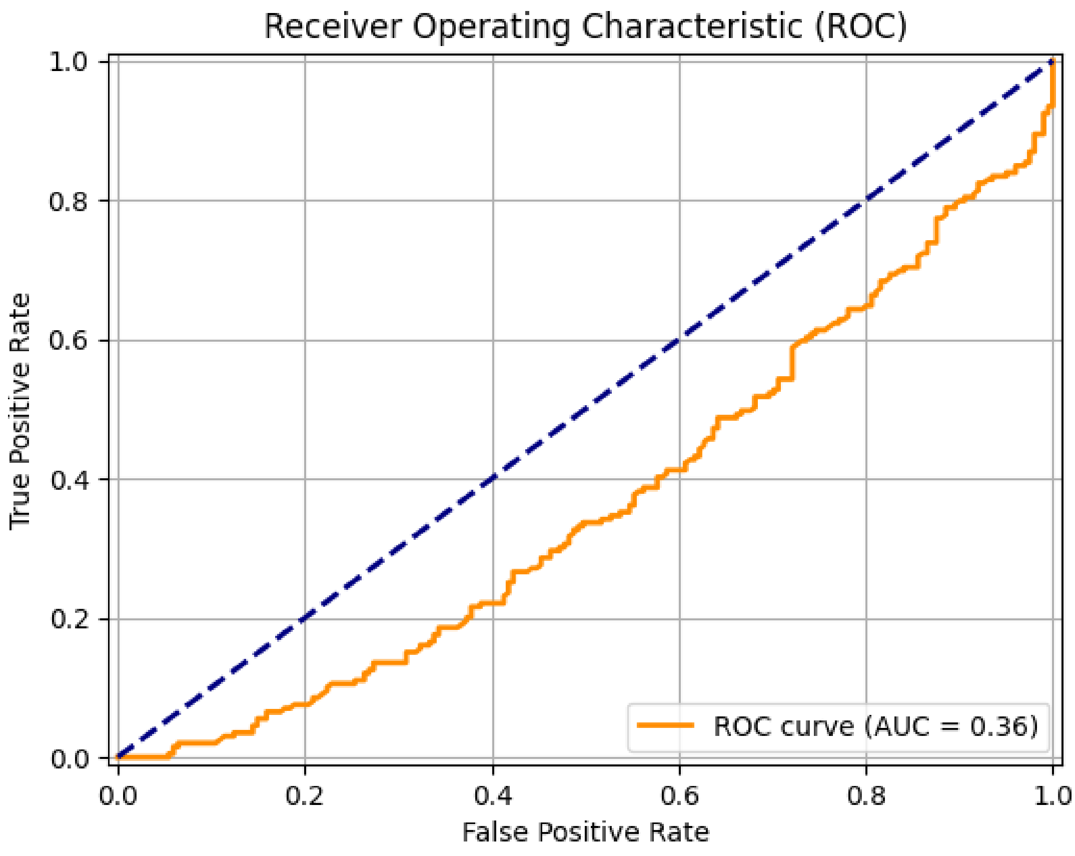
```
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 21ms/step - accuracy: 0.4402 - loss: 0.7186
Test Accuracy: 0.4475
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step
              precision    recall  f1-score   support

  Non-Cancer       0.31      0.08      0.13       201
      Cancer       0.47      0.82      0.60       199

    accuracy                           0.45       400
```

```
   macro avg       0.39      0.45      0.36       400
weighted avg       0.39      0.45      0.36       400
```



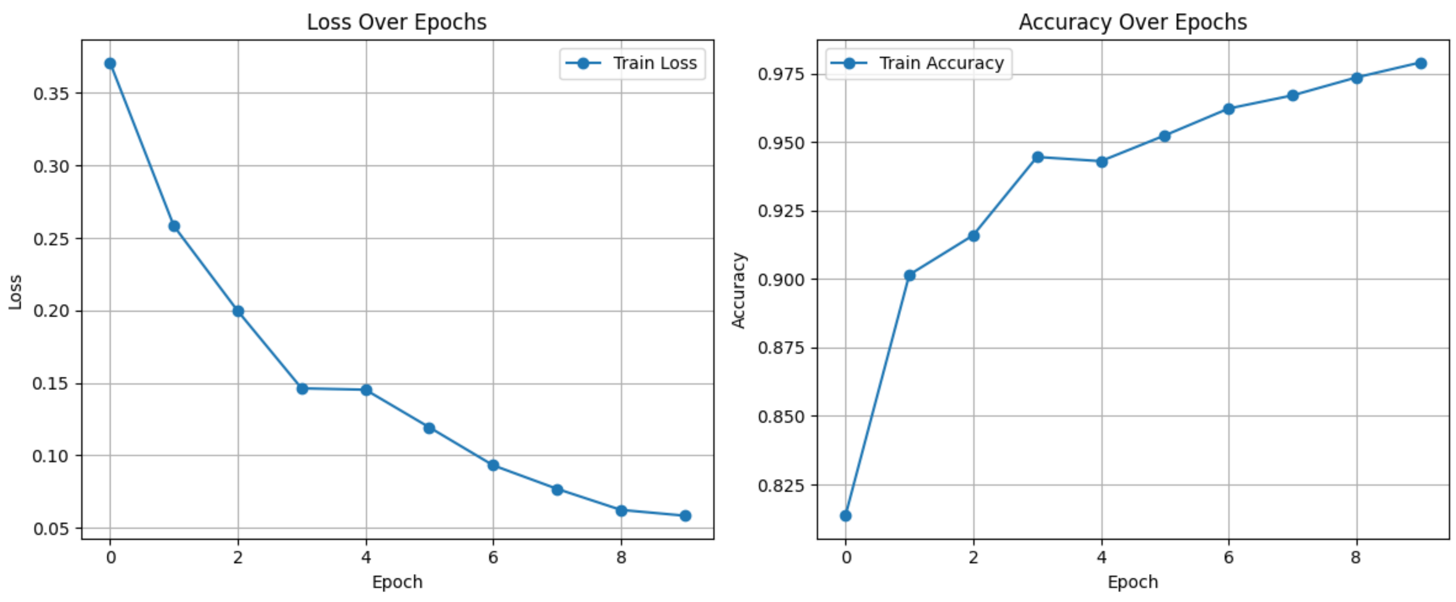


```
T-Test: t-statistic = -3.6596, p-value = 0.0003
Z-Test: z-score = -2.1000, p-value = 0.9821
ANOVA: F-statistic = 823.3633, p-value = 0.0000
```

–

PROJECT-3

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
Epoch 1: Loss=0.3710, Accuracy=0.8136
Epoch 2: Loss=0.2582, Accuracy=0.9016
Epoch 3: Loss=0.1996, Accuracy=0.9160
Epoch 4: Loss=0.1462, Accuracy=0.9446
Epoch 5: Loss=0.1453, Accuracy=0.9430
Epoch 6: Loss=0.1193, Accuracy=0.9524
Epoch 7: Loss=0.0931, Accuracy=0.9622
Epoch 8: Loss=0.0769, Accuracy=0.9670
Epoch 9: Loss=0.0624, Accuracy=0.9736
Epoch 10: Loss=0.0584, Accuracy=0.9790
```



Epoch	**Loss**	**Accuracy**
1	0.3710	81.36%
5	0.1453	94.30%
10	0.0584	97.90%

- By epoch 10, the low loss and high accuracy suggest the model has fit the training data very well.
- However, only training data is used here — this performance doesn’t guarantee good results on unseen data.
- Important next step: Evaluate on the test set to check for overfitting or generalization issues.