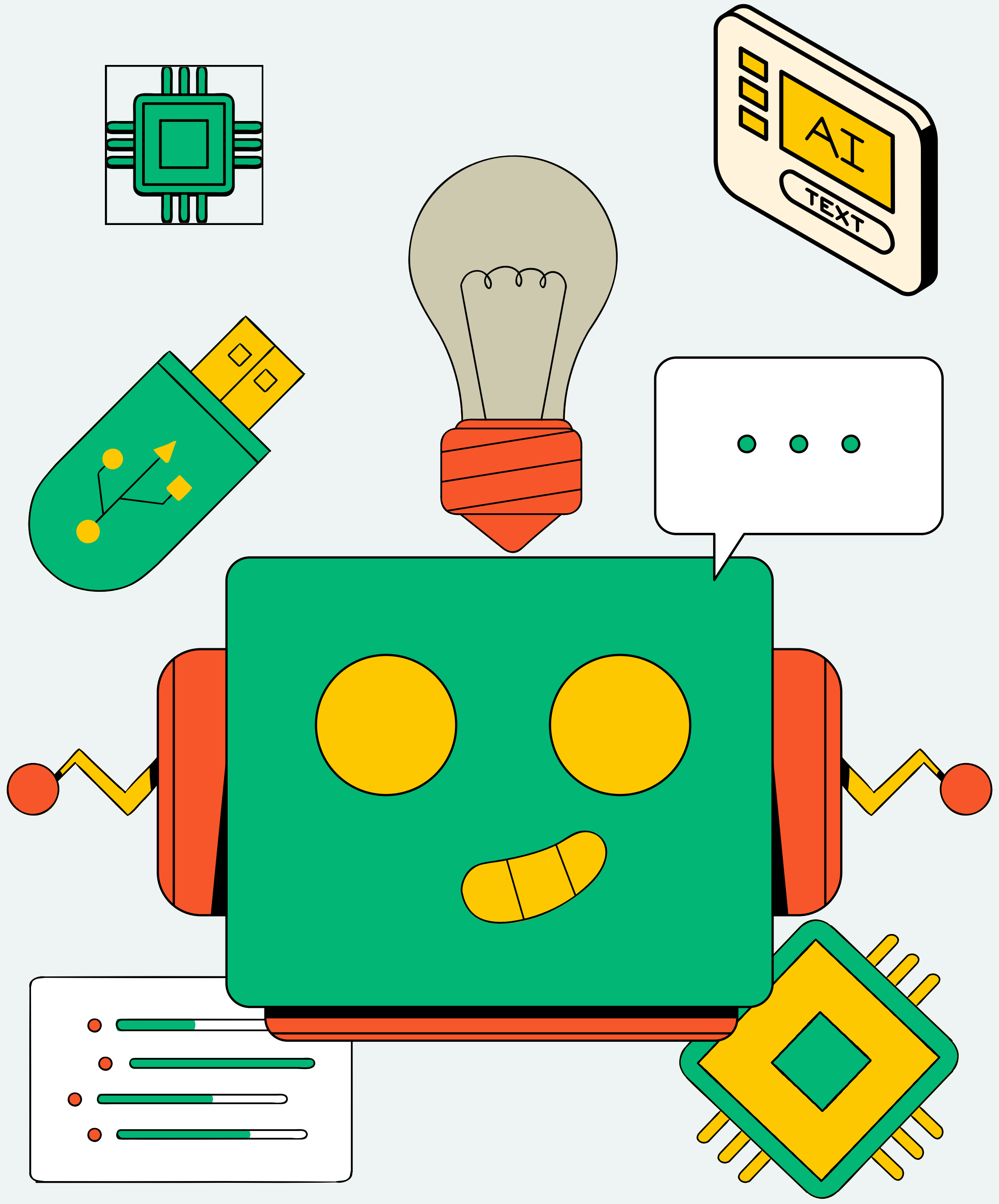
FACIAL EXPRESSION RECOGNITION

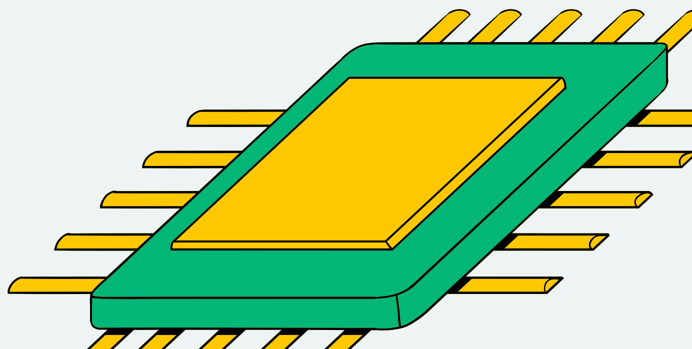


SML PRESENTATION

PRESENTED BY:

ABHAY DAGAR (2022014)

ROHAN BASUGADE (2022416)



PRESENTATION OUTLINE

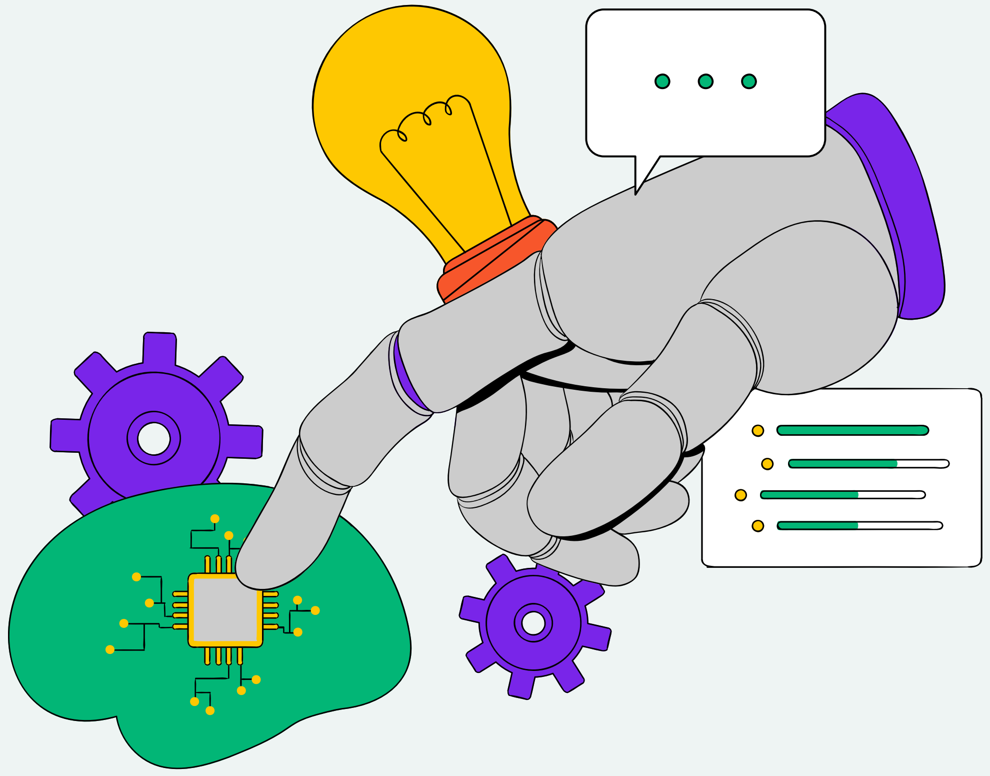
- Introduction
- Importance And Application
- Dataset Detail
- Models We Made?
- Decision Tree, Random Forest, Adaboost
- Graphs plotted for Analysis
- Confusion Matrix, F1 Score, ROC Curve, Accuracy of Models, Classwise Accuracy
- CNN Model
- Validation Accuracy & Loss
- Conclusion And Inference
- Models Testing and Prediction?
- Photos, Video, Webcame
- User Interface (Streamlit)

INTRODUCTION

Facial Expression Recognition (FER) is a computer vision task aimed at identifying and categorizing emotional expressions depicted on a human face.

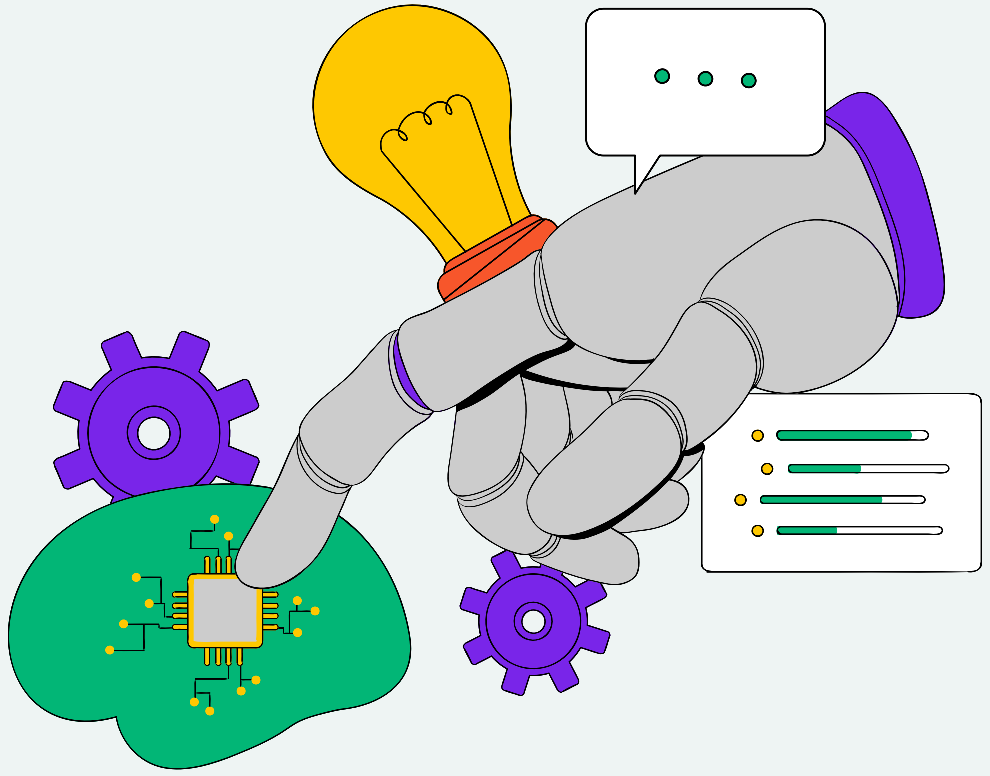
The goal is to automate the process of determining emotions in real-time by analyzing various facial features such as eyebrows, eyes, mouth, and other relevant regions. These features are then mapped to a set of emotions, including anger, fear, surprise, sadness, and happiness.

IMPORTANCE AND APPLICATIONS



1. **Human-Computer Interaction:** FER enables more natural and intuitive interactions between humans and machines. For instance, emotion-aware interfaces can adapt their behavior based on the user's emotional state.
2. **Behavioral Analysis:** Researchers and psychologists use FER to study human behavior, emotional responses, and social interactions.
3. **Virtual Reality & Augmented Reality:** FER enhances immersive experiences by adapting virtual environments based on the user's emotions.
4. **Mental Health Monitoring:** FER can aid in monitoring mental health conditions by detecting changes in emotional states.

OVERVIEW OF OUR PROJECT



Recent research in FER has focused on employing diverse machine learning and deep learning techniques to achieve higher accuracy and real-time processing capabilities

Our project implemented and evaluated various models, including Decision Trees, Random Forests, AdaBoost, and CNNs. Notably, the CNN model achieved an accuracy exceeding 92%.

In summary, FER continues to evolve, with ongoing efforts aimed at enhancing accuracy and robustness.

DATASET DETAIL

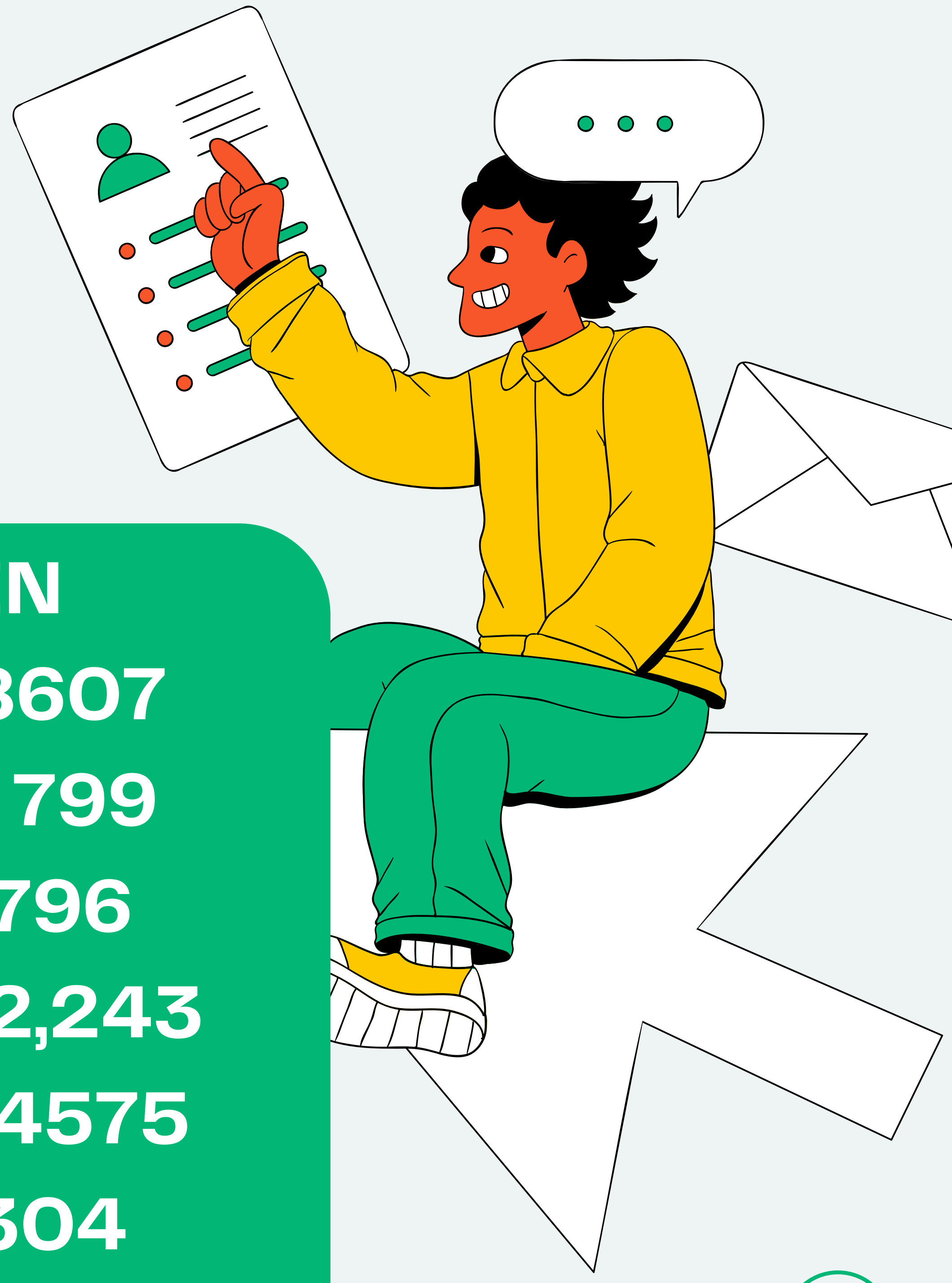
Fer2013 Dataset & FED Dataset: This dataset contains facial images categorized into seven emotion classes: Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral. It was prepared by Pierre-Luc Carrier and Aaron Courville.

Dataset Size: 50,262(Val: 16557,Train: 50262)

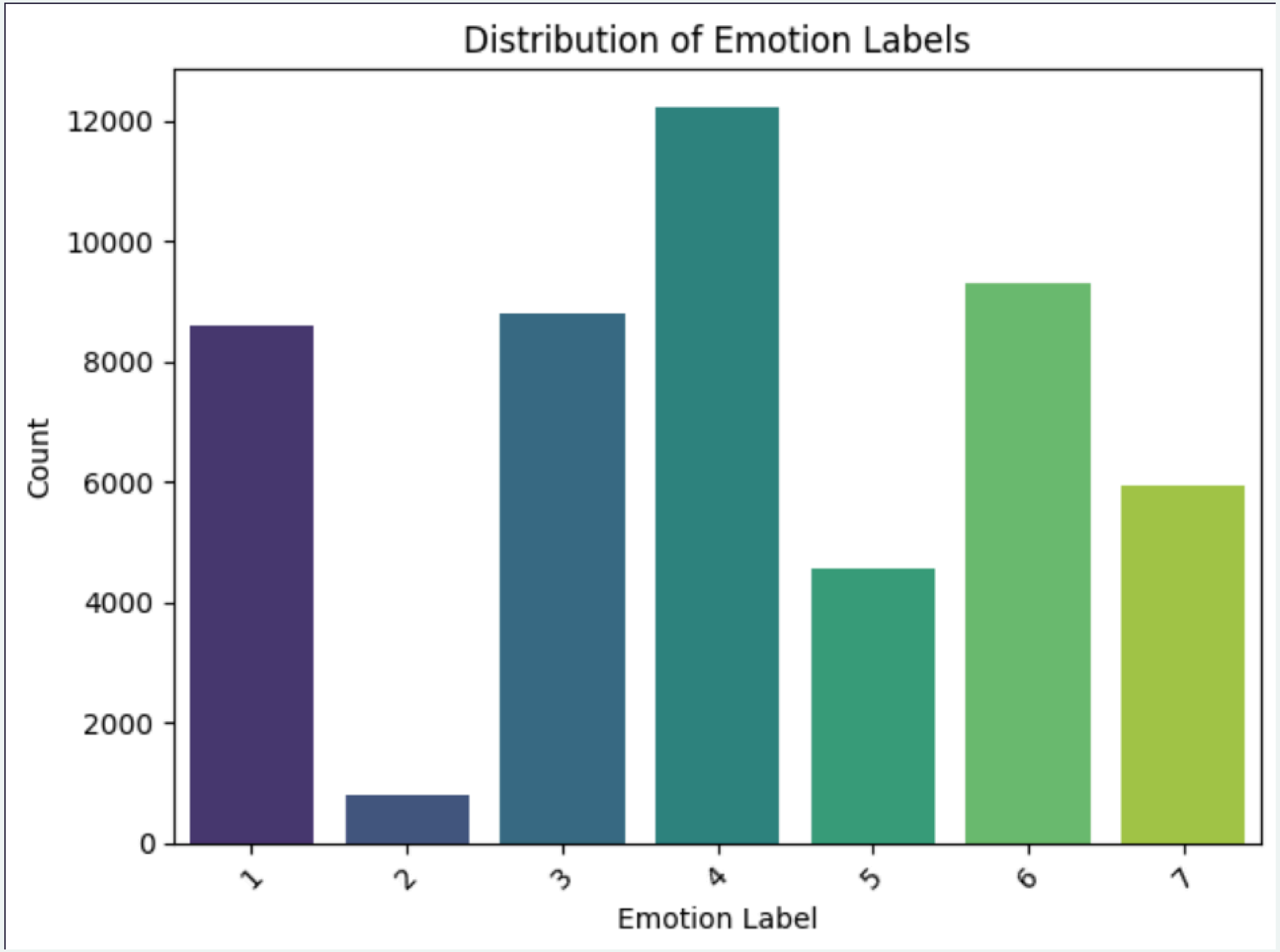
Handling Imbalance Data

The small size of the Disgust image class (799 out of 50,262) posed a challenge.
Three options were considered:
(i) Concatenating Disgust with Angry data.
(ii) Using Synthetic Minority Over-sampling Technique (SMOTE) for class balancing.
(iii)Leaving the data as is.
After testing, the decision was made to leave the data unchanged due to the precision of deep neural networks and the desire to maintain dataset sparsity.

TRAIN
Angry: 8607
Dsgust: 799
Fear: 8796
Happy: 12,243
Neutral: 4575
Sad: 9304
Surprise: 5938

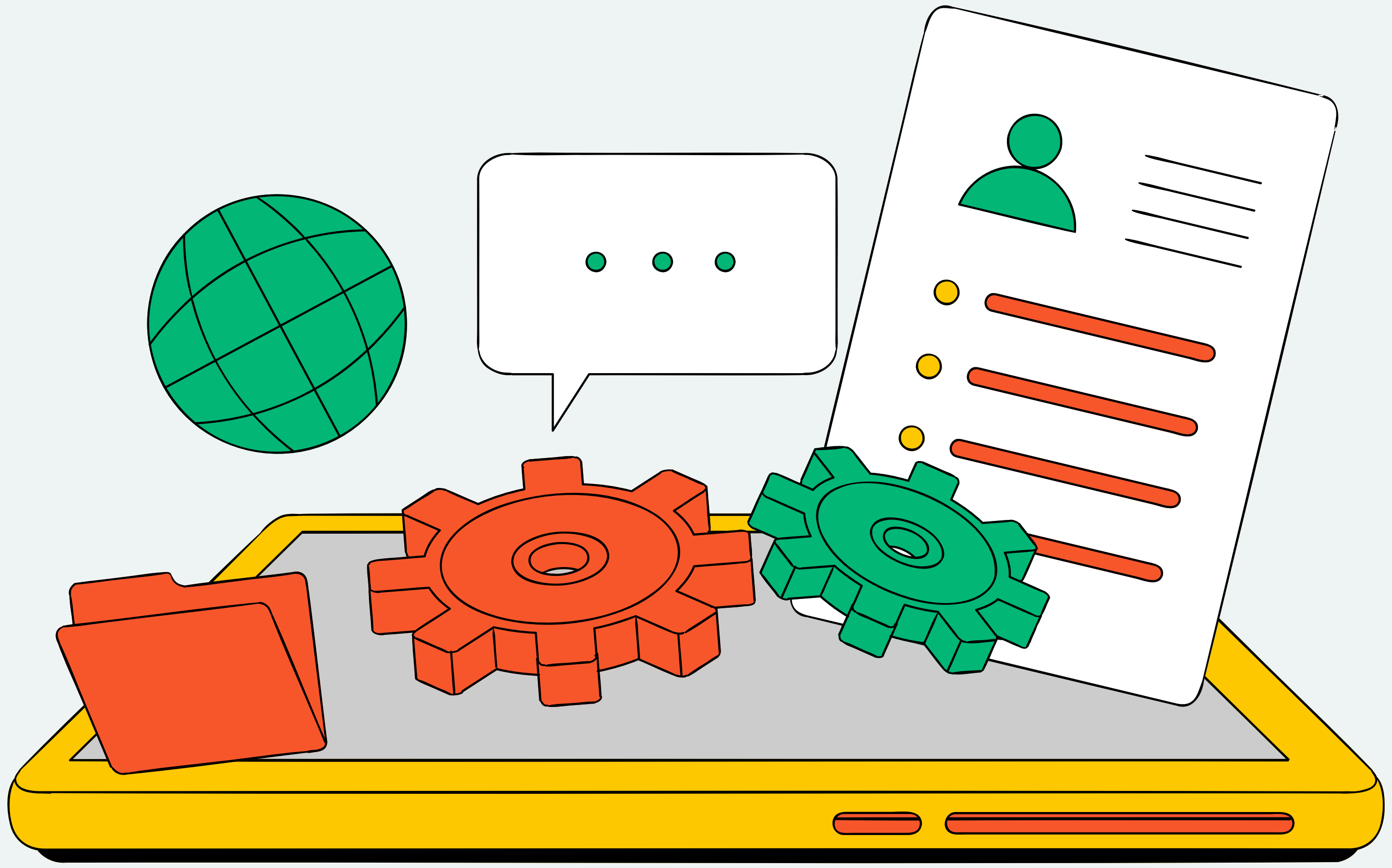


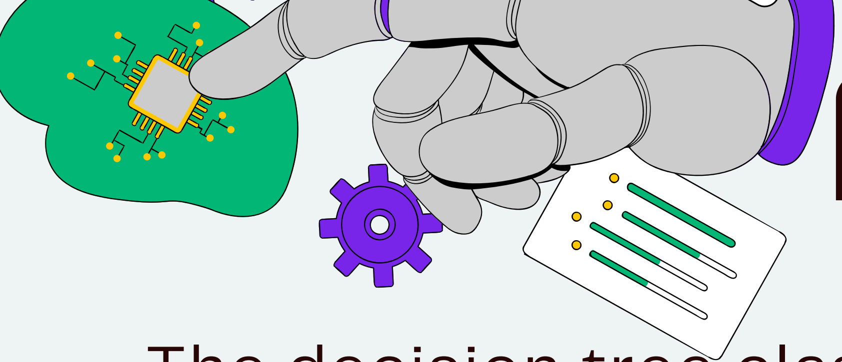
Training Data



MODELS THAT WE MADE ARE:

- Decision Tree
- Random Forest
- AdaBoost
- CNN(Multi Layer Preceptron)

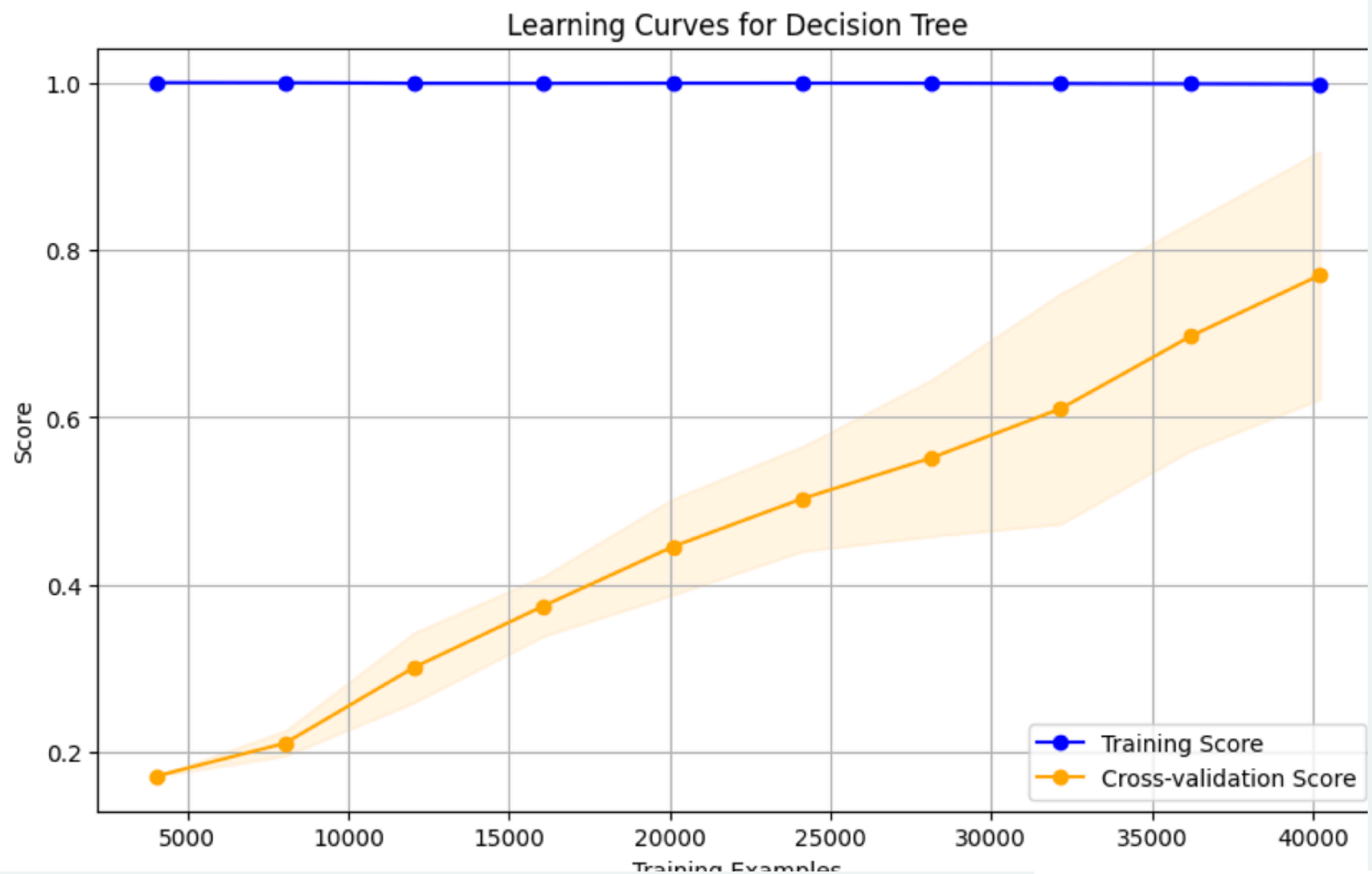




MODEL 01 : DECISION TREE

The decision tree classifier is a powerful non-linear supervised learning algorithm commonly used for classification tasks. Its architecture consists of decision nodes and leaf nodes, forming a tree-like structure. Here's how it works:

1. **Decision Nodes:**
    - Each decision node represents a feature value test (e.g., "Is the intensity of the eyebrow raised more than a threshold?").
    - Based on the outcome of this test, the decision tree branches into different paths.
2. **Leaf Nodes:**
    - Each leaf node represents a class label (e.g., "Happy," "Sad," etc.).
    - When a sample reaches a leaf node, it is assigned the corresponding class label.
3. **Training Process:**
    - During training, the decision tree algorithm recursively splits the feature space into subsets.
    - It optimizes the splits to maximize information gain (or minimize impurity) at each decision node.
    - The goal is to create decision boundaries that effectively separate different classes.
4. **Workflow:**
    - Let's break down the workflow for integrating the decision tree classifier into facial expression recognition:
        - Images: Raw facial images serve as input data.
        - Create Batches: To improve processing efficiency, the images are divided into batches.
        - Extract Features: Relevant features (such as intensity of facial landmarks, texture patterns, etc.) are extracted from the images.
        - Train Classifier: The decision tree classifier is trained using the extracted features.
        - Predict Validation Labels: Once trained, the classifier predicts the labels (emotions) of validation data.



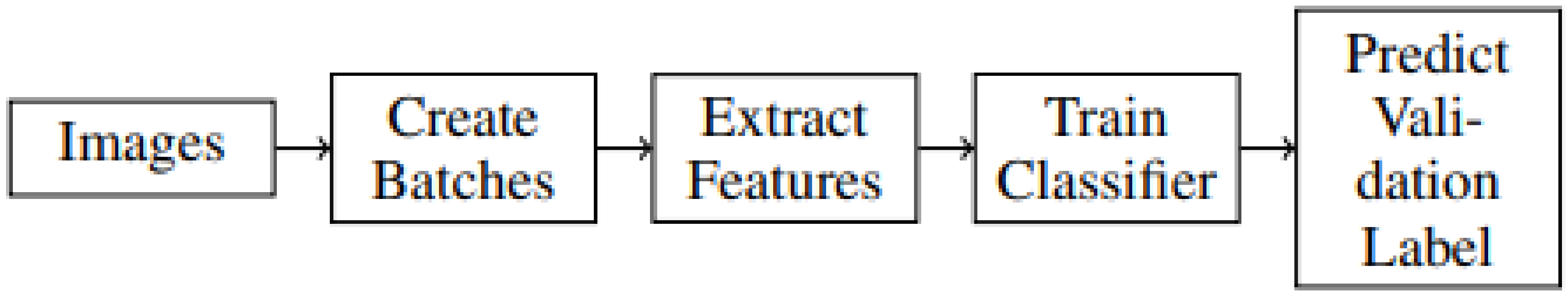
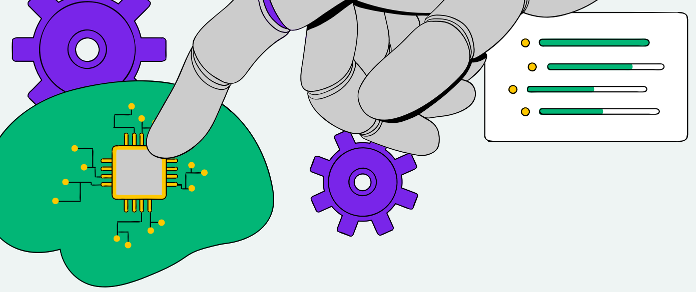


Fig. 3. Workflow Diagram



MODEL 2: RANDOM FOREST

The Random Forest classifier is an ensemble learning method that combines multiple decision trees to make predictions. Here's how it works:

1. **Decision Trees:**
   - Each decision tree in the Random Forest is trained on a random subset of the training data.
   - Unlike a single decision tree, which can be prone to overfitting, Random Forests create diversity by using different subsets of data for each tree.
2. **Bootstrap Aggregation (Bagging):**
   - Random Forests use bootstrap sampling (sampling with replacement) to create multiple subsets of the training data.
   - Each subset is used to train a separate decision tree.
   - This ensures diversity among the trees, as they learn from slightly different data points.
3. **Random Feature Selection:**
   - At each node of a decision tree, a random subset of features is considered for splitting.
   - This randomness reduces the correlation between trees.
   - It prevents all trees from making similar decisions based on the same features.

4. **Voting:**

During classification, each tree in the Random Forest independently predicts the class for a given input.

The class that receives the most votes (mode of the classes predicted by individual trees) becomes the final prediction.

5. **Principal Component Analysis (PCA):**

The Random Forest model in the provided code snippet uses principal components extracted from the input data using PCA.

PCA reduces the dimensionality of the data while preserving important information.

Each decision tree in the Random Forest is trained on a random subset of these principal components.

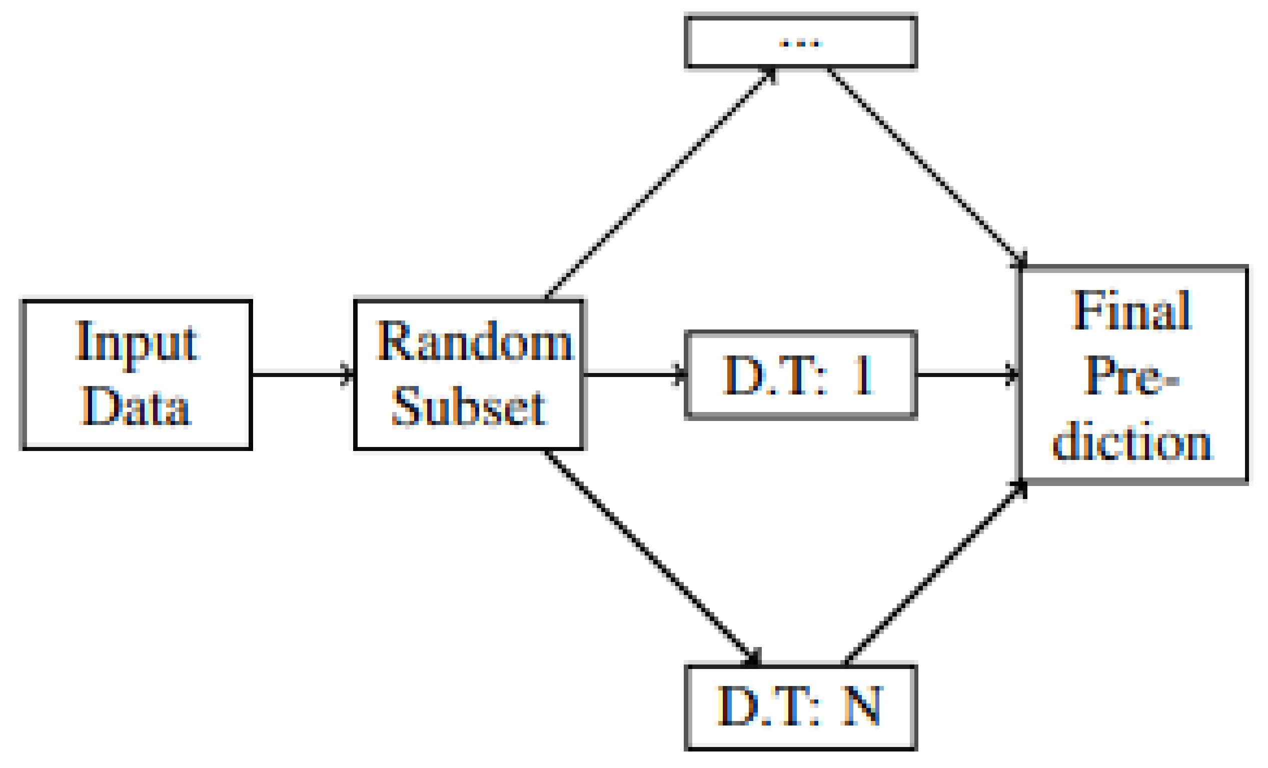
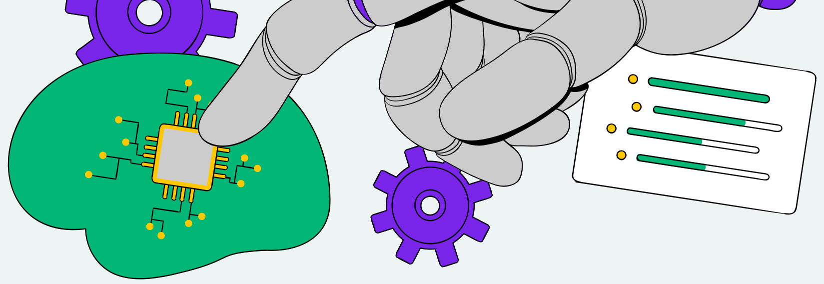


Fig. 7. Architecture of a Random Forest Classifier



MODEL 03 : ADABOOST

The AdaBoost classifier is an ensemble learning method that combines multiple weak classifiers to create a strong classifier. Here's how it works:

1. Weak Classifiers:
    - Weak classifiers are simple models (e.g., decision stumps) that perform slightly better than random guessing.
    - They are the building blocks of the AdaBoost ensemble.
2. Sample Weighting:
    - During training, misclassified examples are assigned higher weights.
    - This prioritizes their correct classification in subsequent iterations.
3. Sequential Training:
    - Classifiers are trained sequentially.
    - Each subsequent classifier focuses more on examples that were misclassified by the previous ones.
    - This adaptive approach improves overall performance.
4. Weighted Voting:
    - During classification, each weak classifier's prediction is weighted based on its accuracy.
    - The final prediction is determined by a weighted majority vote across all weak classifiers.
5. Integration with Principal Component Analysis (PCA):
    - The AdaBoost classifier in the provided code snippet is trained on principal components extracted from the input data using PCA.
    - PCA reduces dimensionality while preserving essential information.

WORKFLOW OF ADABOOST

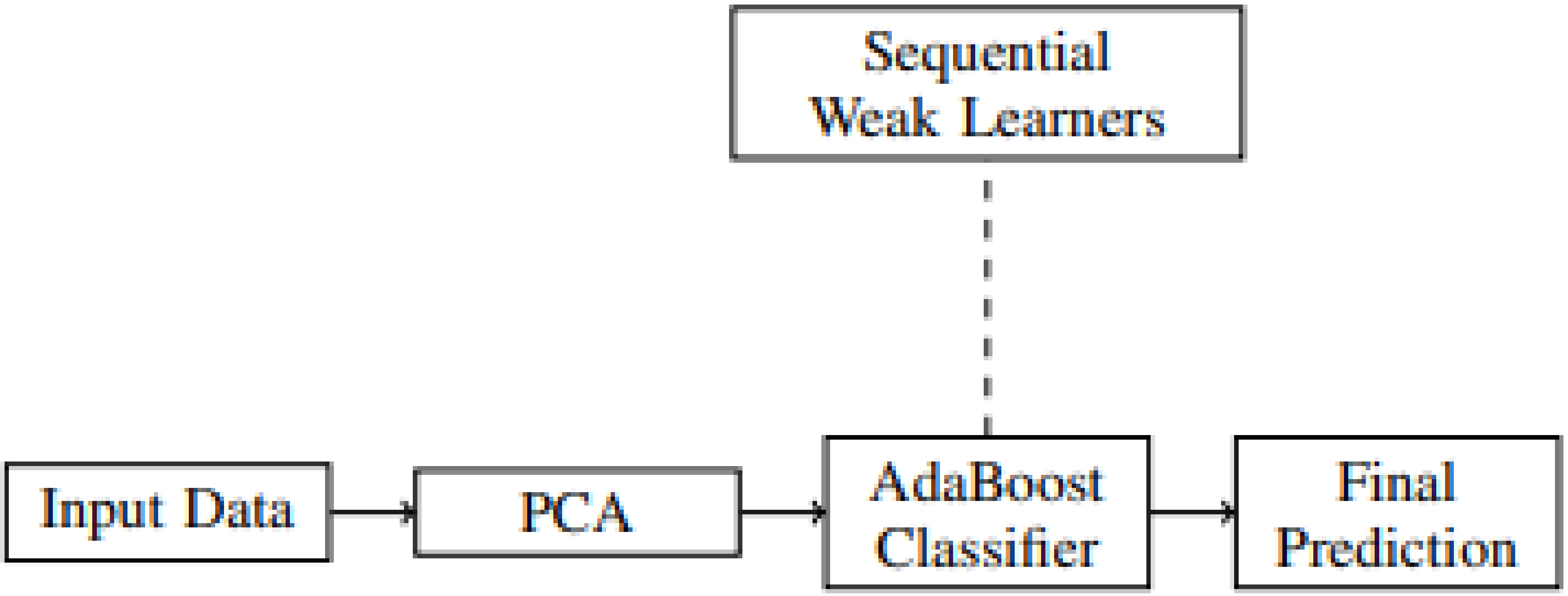


Fig. 10. Workflow Diagram of AdaBoost Classifier

GRAPHS PLOTTED FOR ANALYSIS

CONFUSION MATRIX

F-1 SCORE

ROC CURVE

MODEL ACCURACY

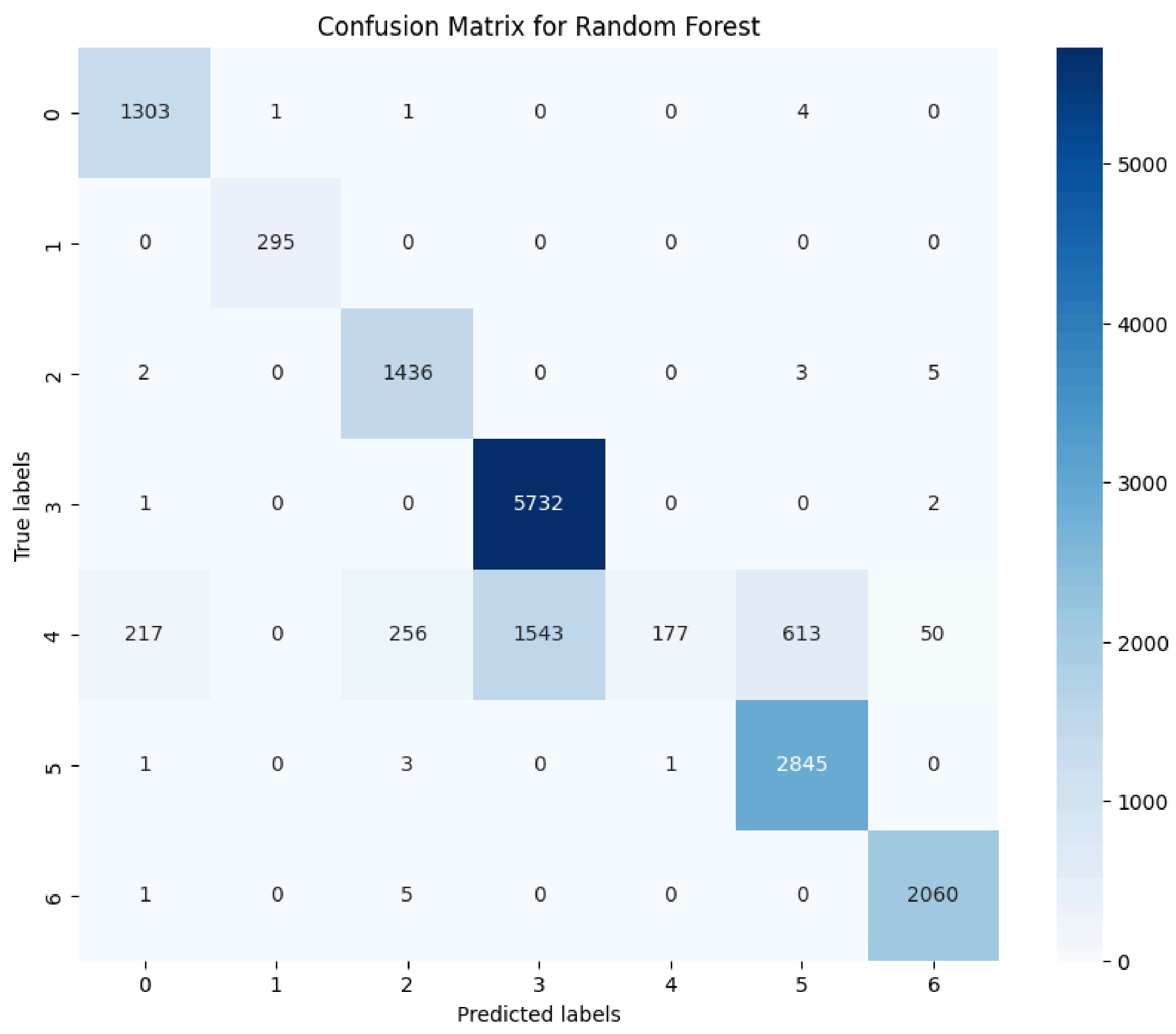
CLASSWISE ACCURACY

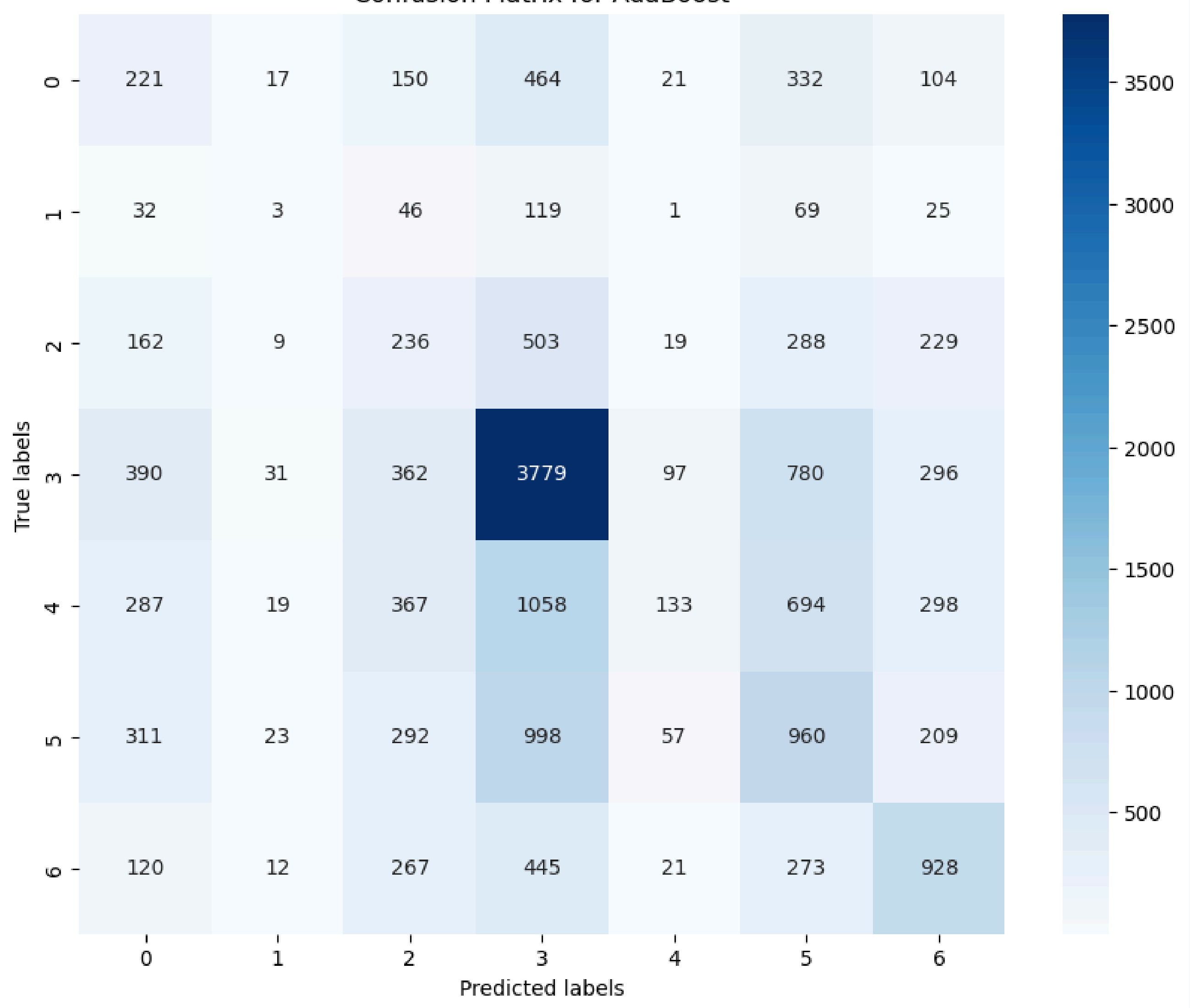
CONFUSION MATRIX

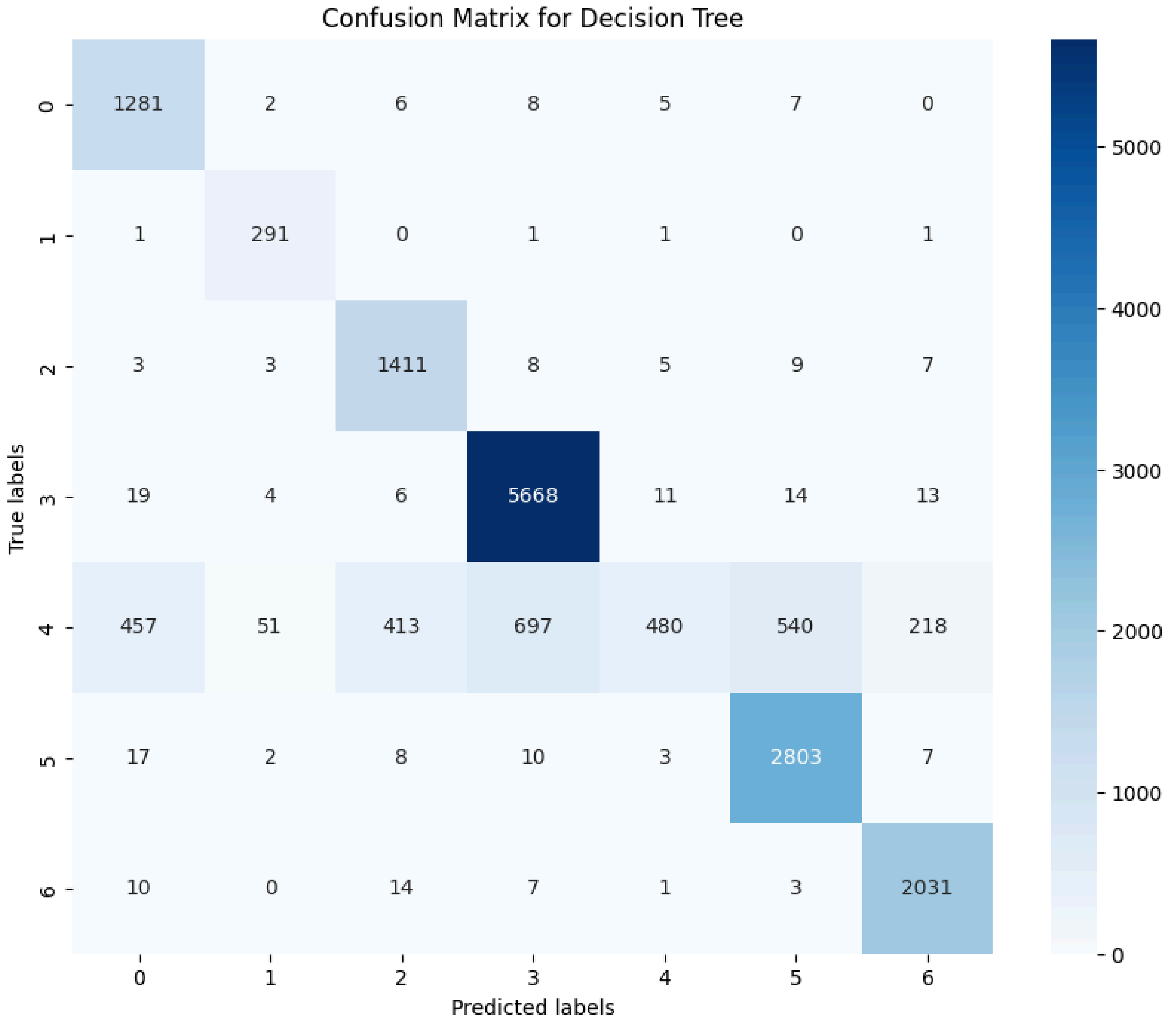
A confusion matrix heatmap is a visual representation that helps us evaluate the performance of a classification model. Here's how it works:

1. What Is a Confusion Matrix?
   - A confusion matrix summarizes the predictions made by a model in terms of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) outcomes.
   - It organizes these outcomes into a matrix, where rows represent the actual class labels, and columns represent the predicted class labels.
2. Elements of the Confusion Matrix:
   - True Positive (TP): Instances correctly predicted as positive (e.g., correctly identifying disease cases).
   - False Positive (FP): Instances incorrectly predicted as positive (e.g., false alarms).
   - True Negative (TN): Instances correctly predicted as negative (e.g., healthy individuals correctly identified).
   - False Negative (FN): Instances incorrectly predicted as negative (e.g., missing disease cases).
3. Heatmap Visualization:
   - The confusion matrix heatmap displays the counts or proportions of these outcomes.
   - Each cell in the heatmap corresponds to a combination of actual and predicted classes.
   - The intensity of color in each cell indicates the frequency or proportion of instances falling into that category.
4. Interpreting the Heatmap:
   - Brighter colors (higher values) indicate better performance.
   - Patterns emerge, allowing us to identify where the model excels (high TP and TN) and where it struggles (high FP and FN).

CONFUSION MATRIX OF MODELS



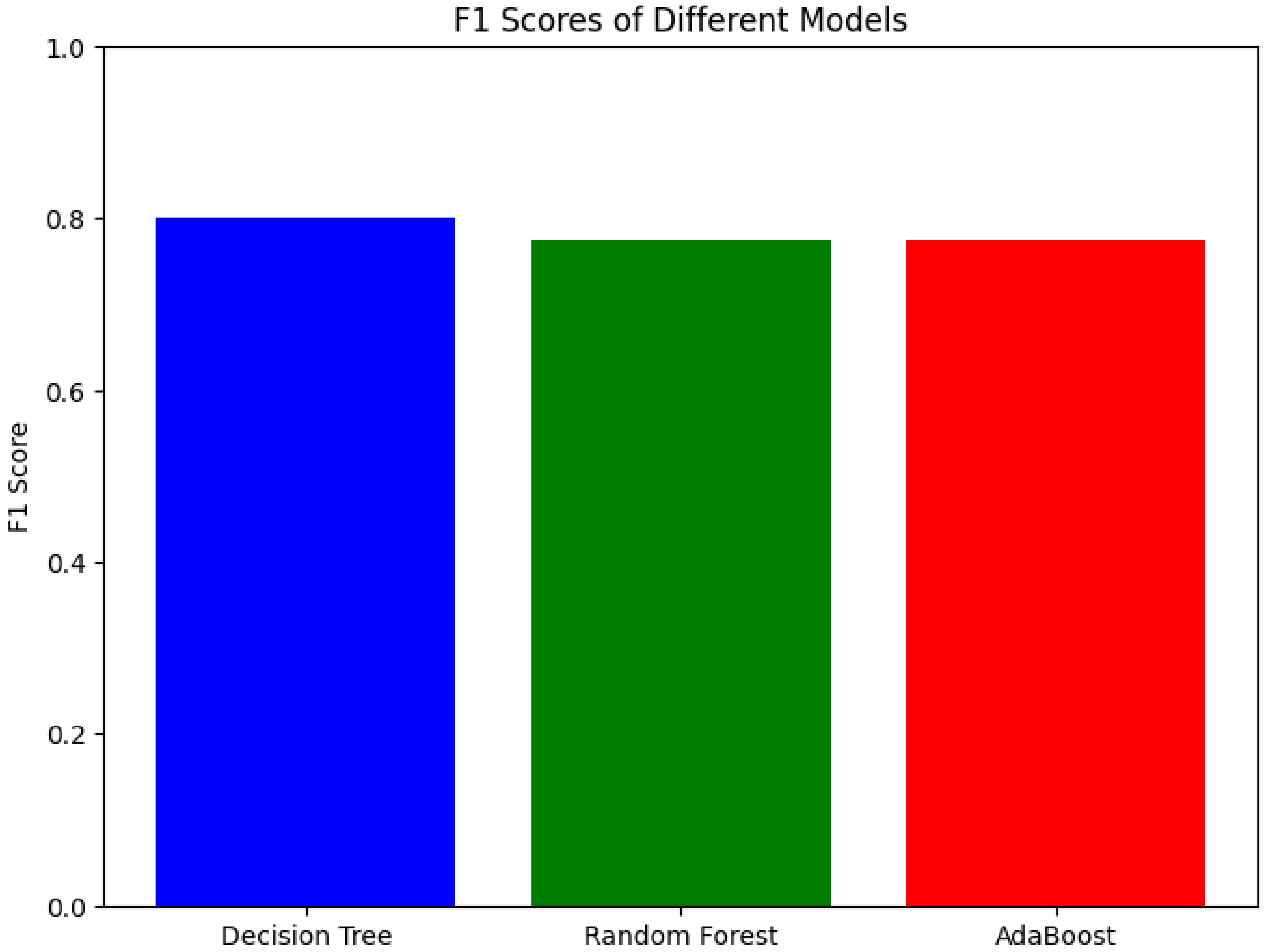




F-1 SCORE

The F1 score (also known as F-measure or balanced F-score) is a common classification machine learning metric. It provides a balanced view of a model's performance by considering both precision and recall for the minority positive class.

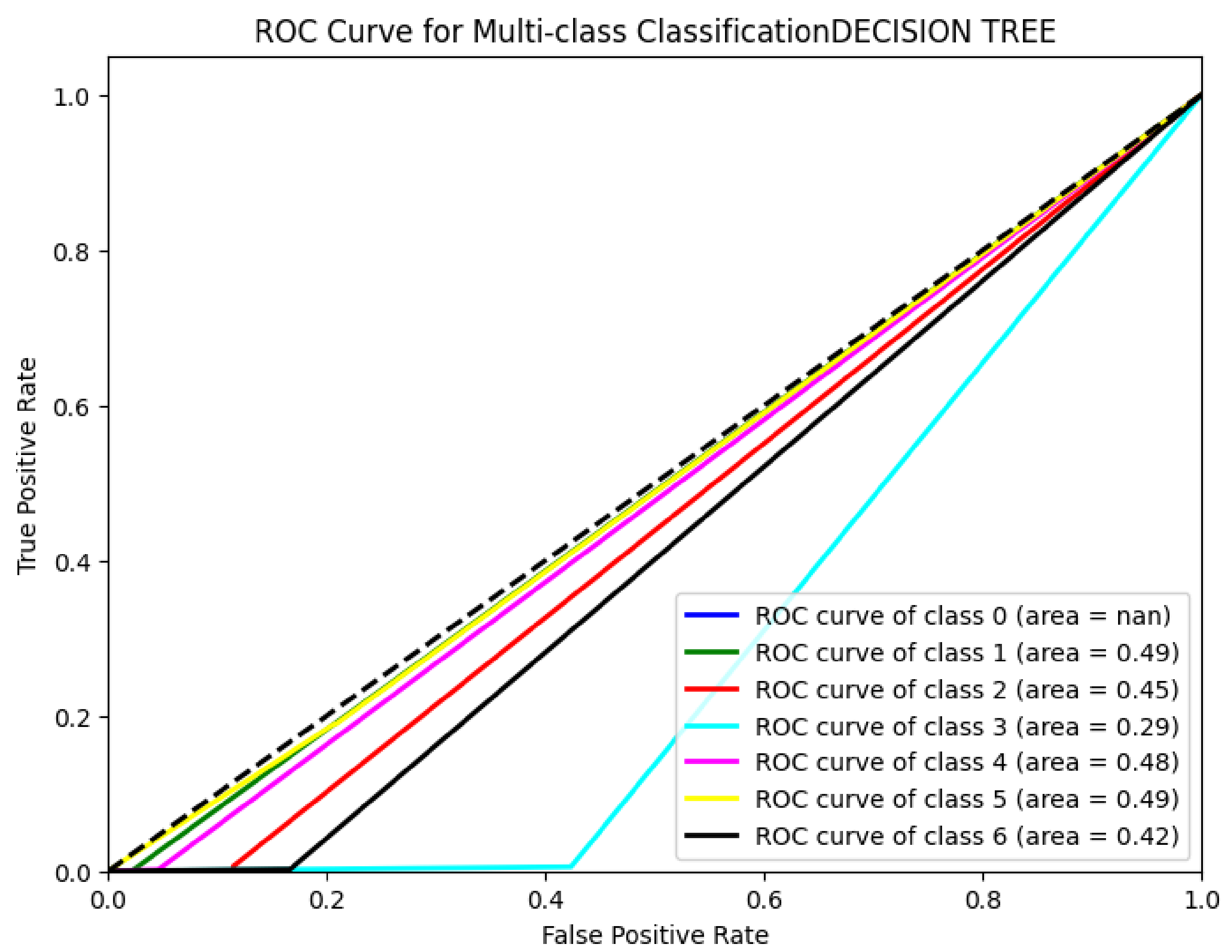
1. Definition:
   - The F1 score is the harmonic mean of precision and recall.
   - It symmetrically represents both precision (accuracy with positive predictions) and recall (ability to capture positive cases).
2. Mathematical Formula:
   - The F1 score can be expressed as:
   - F1=precision+recall2 · precision · recall
3. Interpretation:
   - The F1 score ranges from 0 to 1, where 1 represents the best performance.
   - Specifically, it measures the model's balanced ability to:
     - Capture positive cases (recall).
     - Be accurate with the cases it does capture (precision).
4. Interpreting F1 Score Values:
   - Depending on your use case and dataset:
     - F1 score > 0.9: Very good
     - F1 score between 0.8 and 0.9: Good
     - F1 score between 0.5 and 0.8: OK
     - F1 score < 0.5: Not good


F1 Scores of Different Models
F1 Score
1.0
0.8
0.6
0.4
0.2
0.0
Decision Tree
Random Forest
AdaBoost

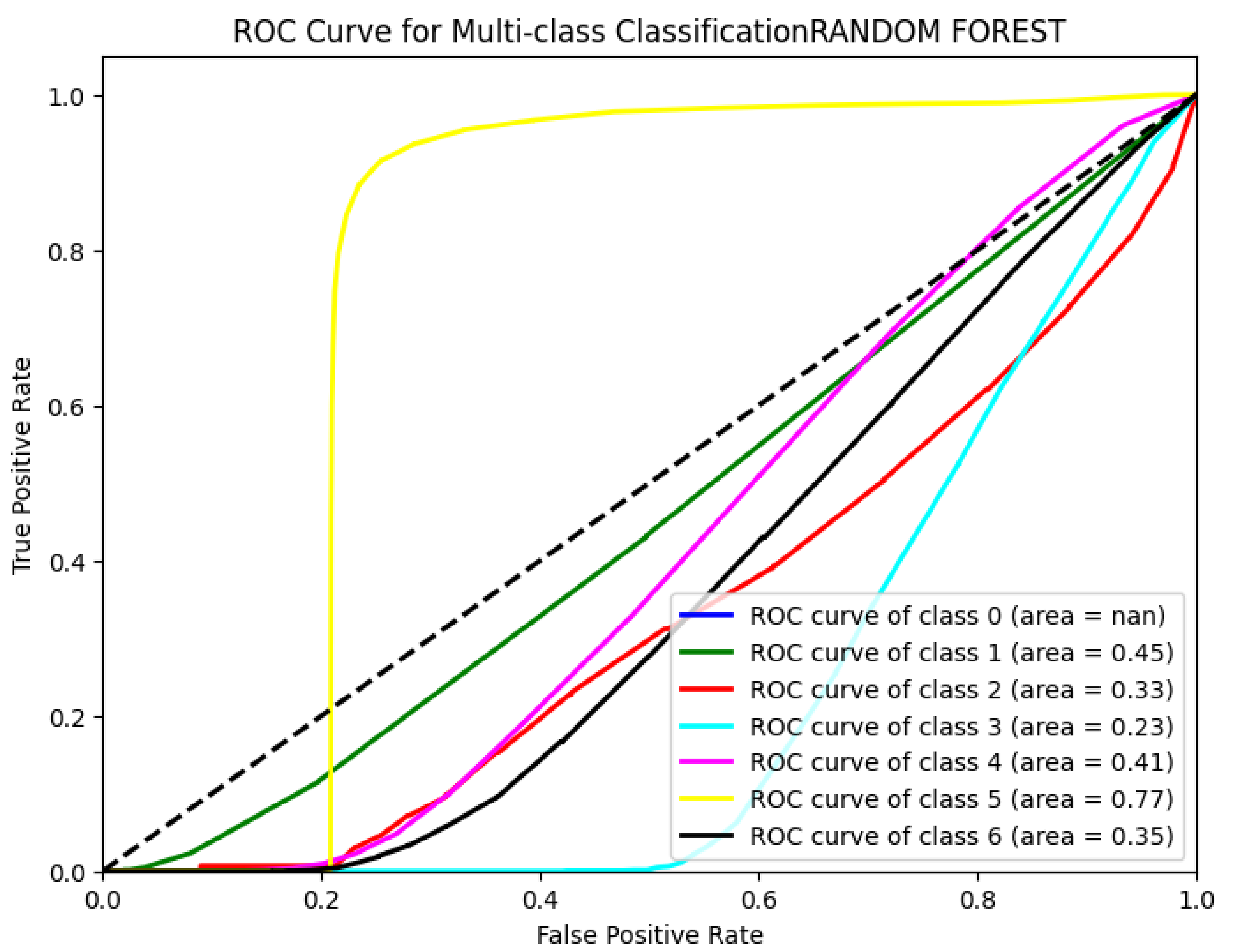
ROC CURVE

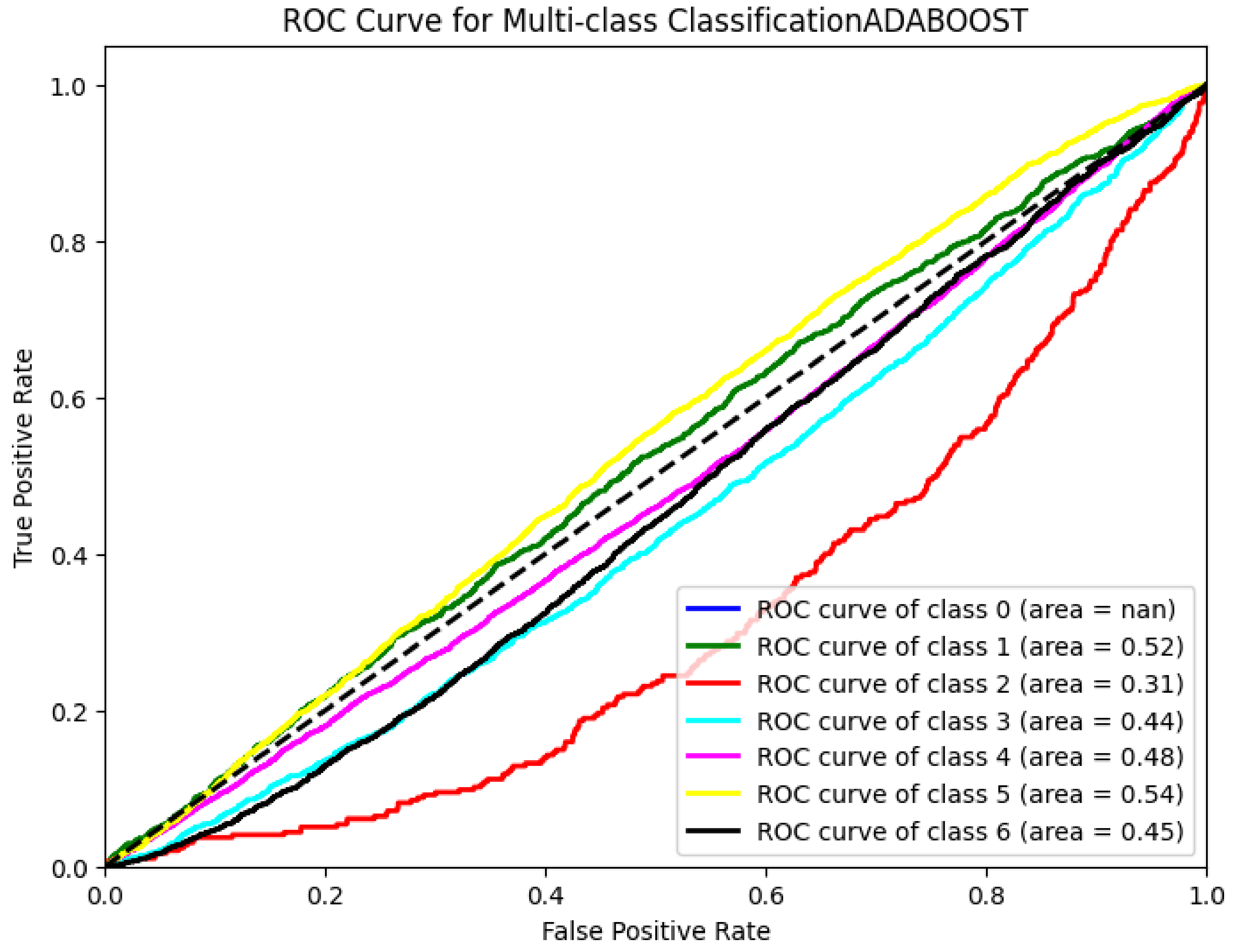
The ROC curve is a powerful tool for assessing the performance of a binary classification model. Here's how it works:

1. What Is the ROC Curve?
    - The ROC curve visualizes how well a model can distinguish between positive and negative classes.
    - It plots the True Positive Rate (TPR) against the False Positive Rate (FPR).
2. True Positive Rate (TPR):
    - Also known as recall or sensitivity.
    - Measures the proportion of actual positive cases correctly predicted by the model.
    - TPR = TP / (TP + FN)
3. False Positive Rate (FPR):
    - Measures the proportion of actual negative cases incorrectly predicted as positive.
    - FPR = FP / (FP + TN)
4. Interpreting the ROC Curve:
    - The curve represents different thresholds for classifying positive cases.
    - Each point on the curve corresponds to a specific threshold.
    - The ideal model achieves a TPR of 1 (perfect recall) and an FPR of 0 (no false positives).

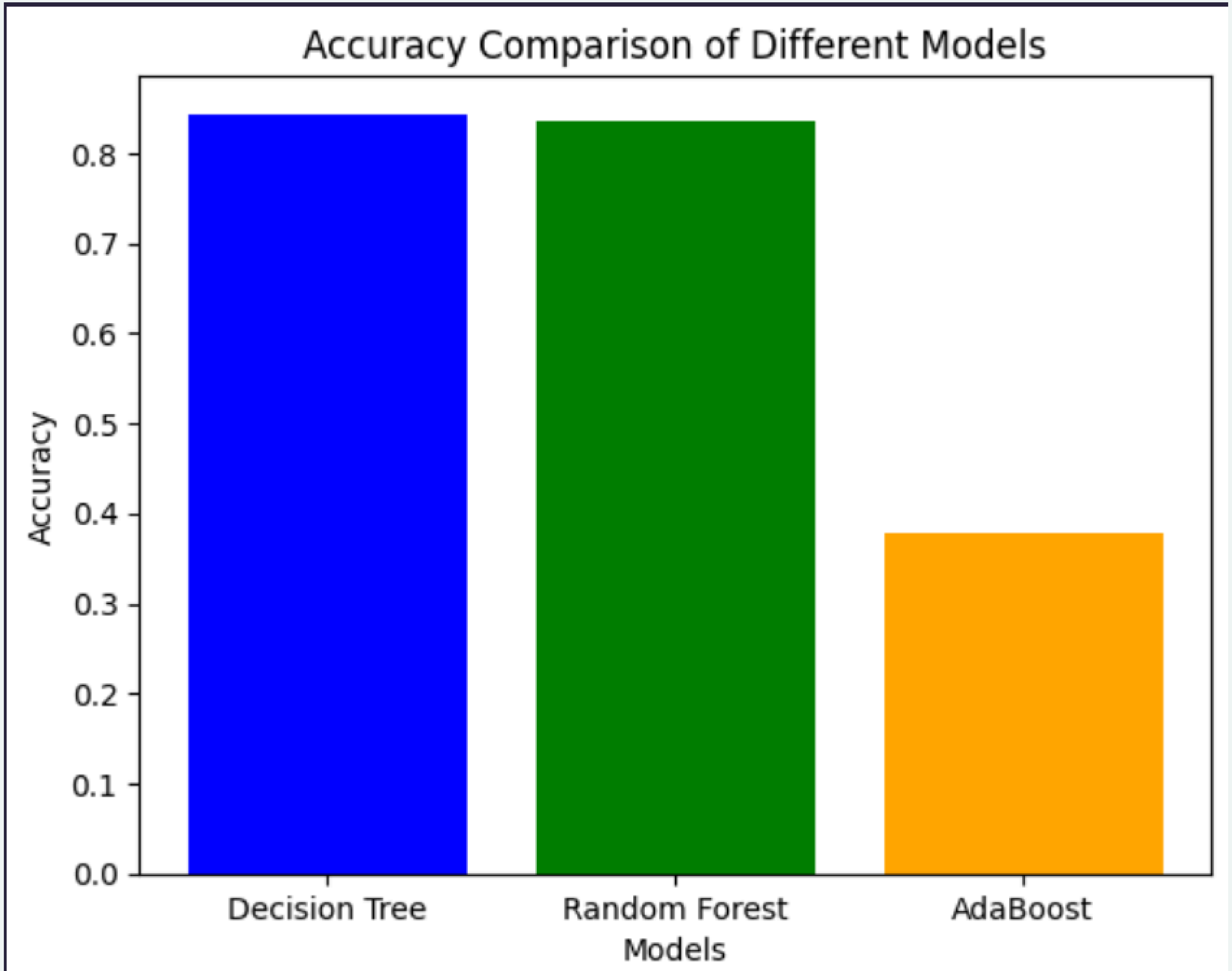

ROC Curve for Multi-class ClassificationDECISION TREE
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = nan)
ROC curve of class 1 (area = 0.49)
ROC curve of class 2 (area = 0.45)
ROC curve of class 3 (area = 0.29)
ROC curve of class 4 (area = 0.48)
ROC curve of class 5 (area = 0.49)
ROC curve of class 6 (area = 0.42)

ROC CURVES

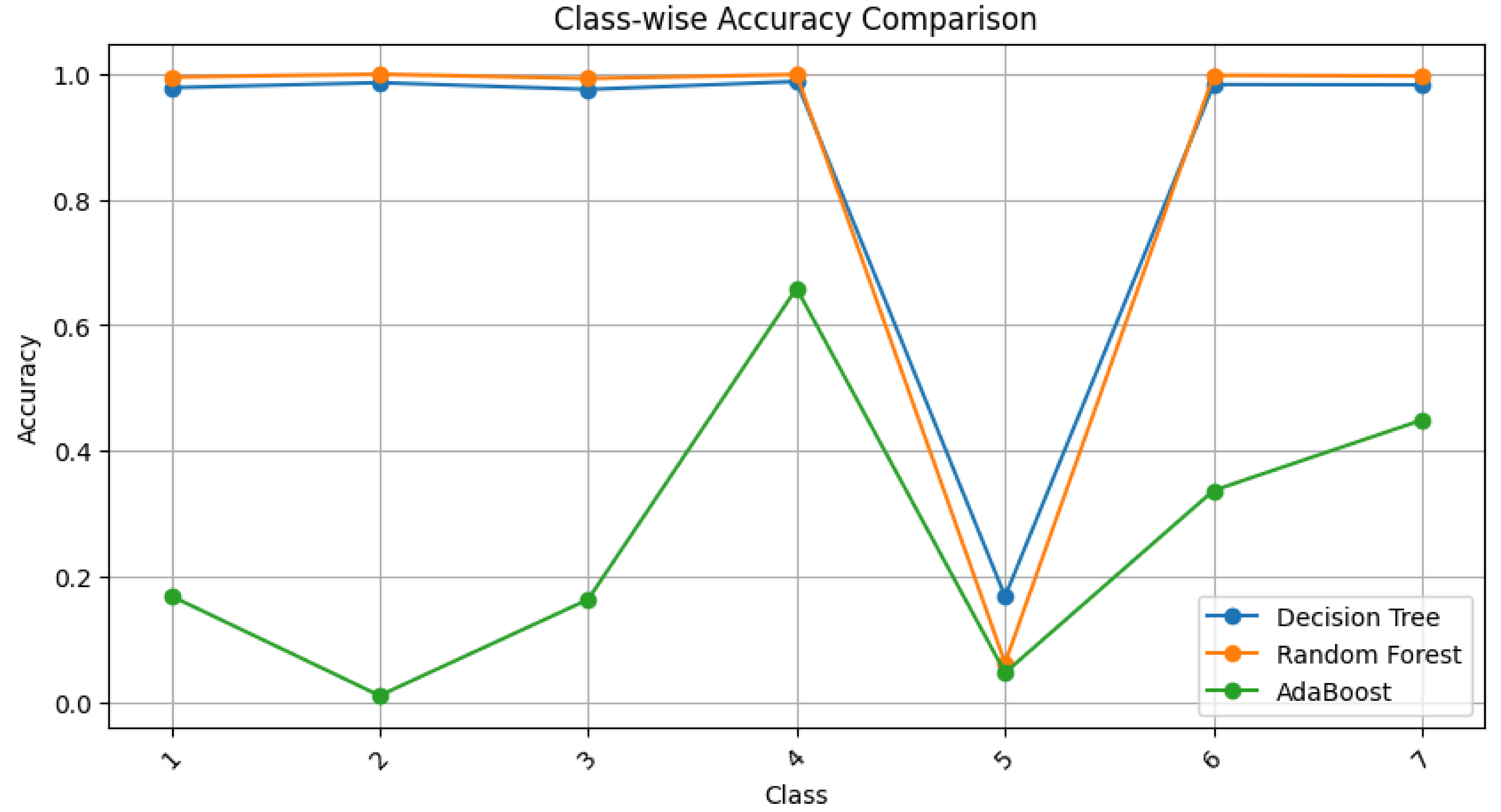

ROC Curve for Multi-class ClassificationRANDOM FOREST
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = nan)
ROC curve of class 1 (area = 0.45)
ROC curve of class 2 (area = 0.33)
ROC curve of class 3 (area = 0.23)
ROC curve of class 4 (area = 0.41)
ROC curve of class 5 (area = 0.77)
ROC curve of class 6 (area = 0.35)


ROC Curve for Multi-class ClassificationADABOOST
True Positive Rate
False Positive Rate
ROC curve of class 0 (area = nan)
ROC curve of class 1 (area = 0.52)
ROC curve of class 2 (area = 0.31)
ROC curve of class 3 (area = 0.44)
ROC curve of class 4 (area = 0.48)
ROC curve of class 5 (area = 0.54)
ROC curve of class 6 (area = 0.45)

ACCURACY OF MODELS



CLASSWISE ACCURACY OF MODELS



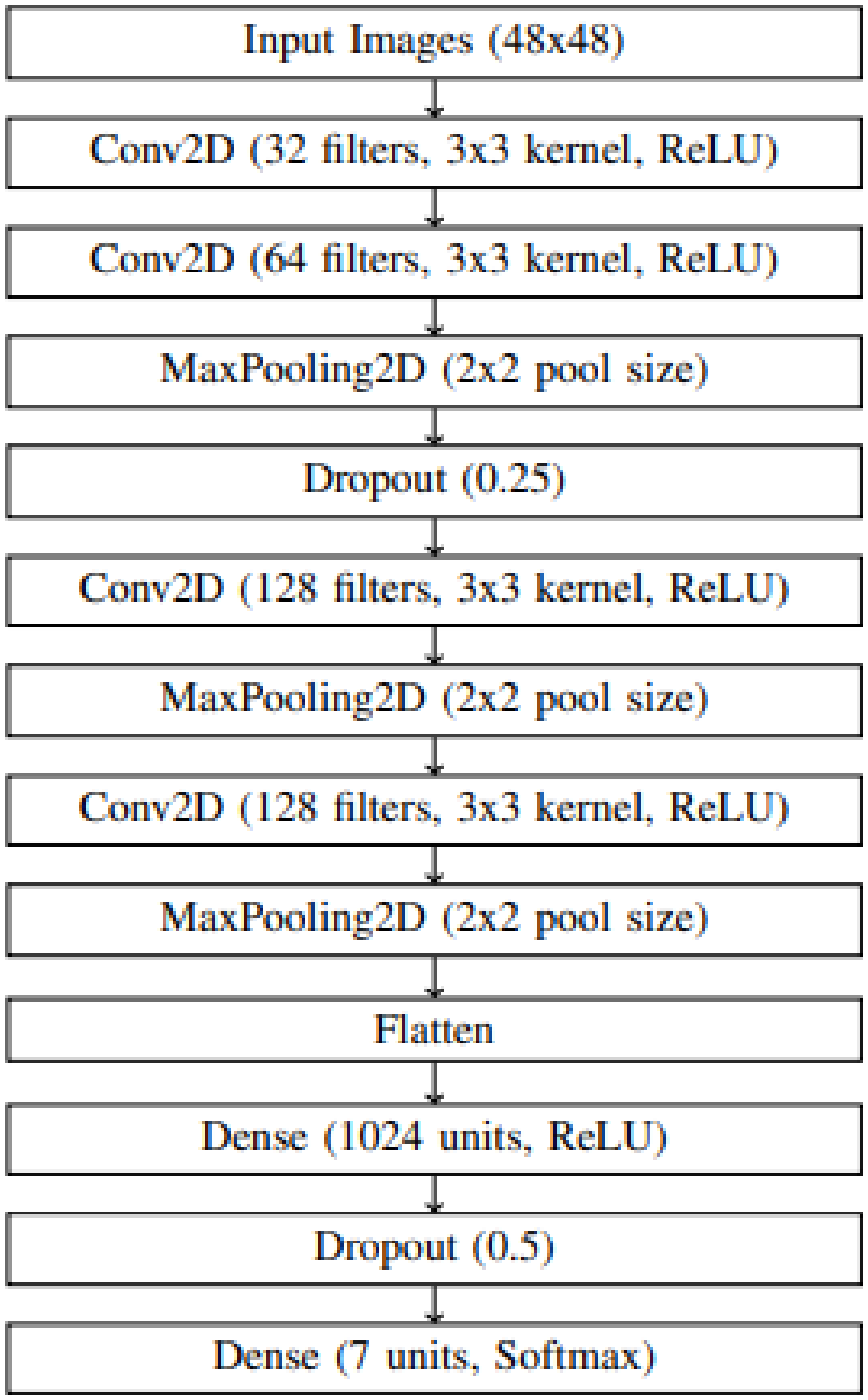
CONVOLUTION NEURAL NETWORK(CNN) MODEL

The CNN model for emotion classification is designed to process grayscale images of size 48x48 pixels. Here are the key components:

1. Input Layer:
   - Receives grayscale images with a single channel (48x48 pixels).
2. Convolutional Layers (Conv2D):
   - The model starts with two convolutional layers.
   - Each layer applies a Rectified Linear Unit (ReLU) activation function.
   - First layer: 32 filters.
   - Second layer: 64 filters.
3. Pooling Layer (MaxPooling2D):
   - After each convolutional layer, a max-pooling layer reduces spatial dimensions while retaining important features.
4. Dropout Layer:
   - Two dropout layers prevent overfitting by randomly deactivating a fraction of neurons during training (dropout rate of 0.25).
5. Additional Convolutional Layers:
   - Two more convolutional layers follow, each with max-pooling and dropout layers.
   - These layers extract higher-level features.

6. Flatten Layer:
    - Flattens the feature maps into a one-dimensional vector.
7. Dense Layers (Fully Connected):
    - First dense layer: 1024 units with ReLU activation.
    - Second dense layer (output layer): 7 units with softmax activation for class probabilities.
8. Model Compilation:
    - Compiled using categorical cross-entropy loss.
    - Optimized with Adam optimizer (learning rate of 0.0001 and decay of 1e-6).
    - Accuracy metric used for evaluation.
9. Training:
    - The model is trained for 100 epochs using training data generated by a data generator with a batch size of 64.

MODEL METHODOLOGY



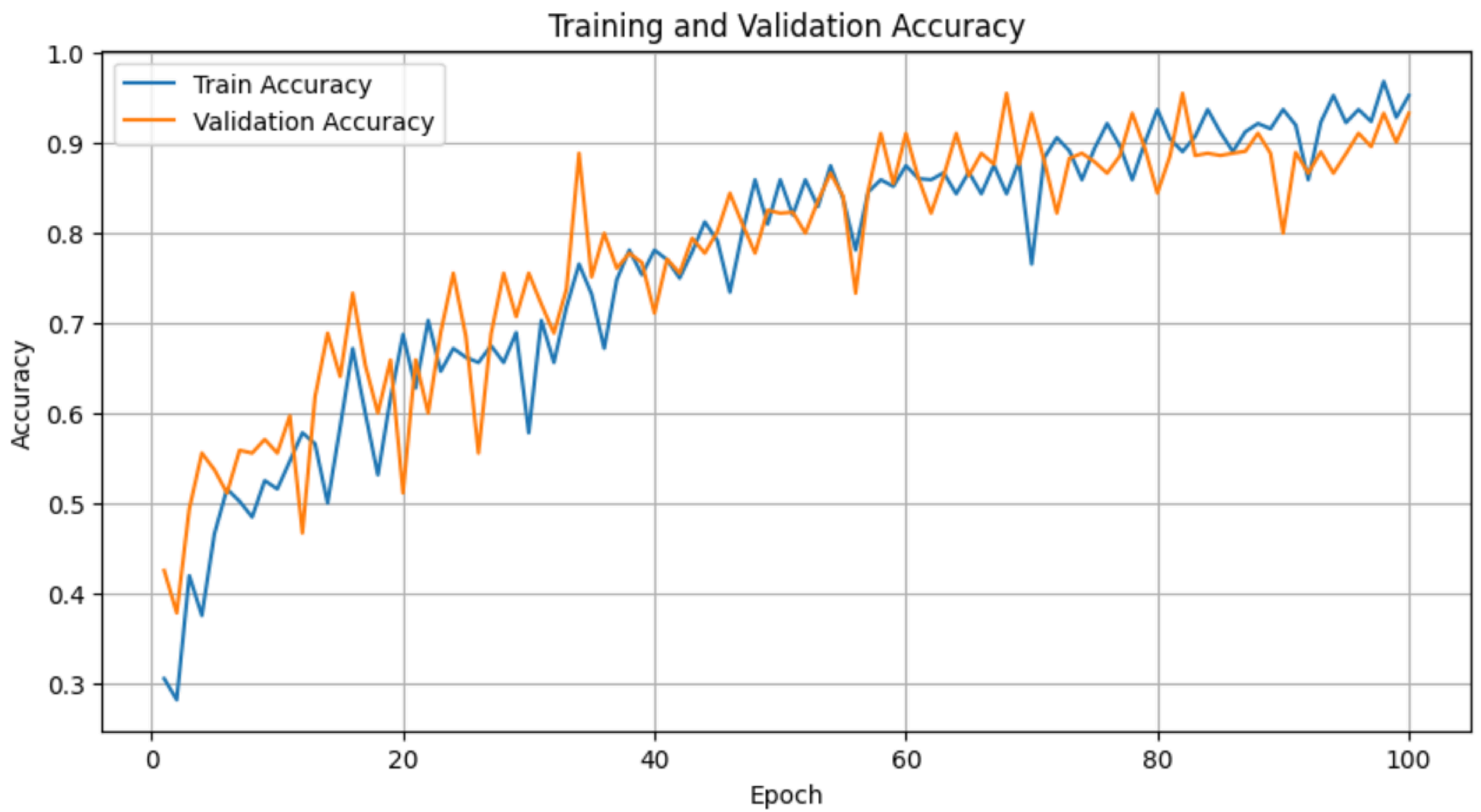
CNN TRAINED MODEL INFO

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 46, 46, 32)	320
conv2d_1 (Conv2D)	(None, 44, 44, 64)	18,496
max_pooling2d (MaxPooling2D)	(None, 22, 22, 64)	0
dropout (Dropout)	(None, 22, 22, 64)	0
conv2d_2 (Conv2D)	(None, 20, 20, 128)	73,856
max_pooling2d_1 (MaxPooling2D)	(None, 10, 10, 128)	0
conv2d_3 (Conv2D)	(None, 8, 8, 128)	147,584
max_pooling2d_2 (MaxPooling2D)	(None, 4, 4, 128)	0
dropout_1 (Dropout)	(None, 4, 4, 128)	0
flatten (Flatten)	(None, 2048)	0
dense (Dense)	(None, 1024)	2,098,176
dropout_2 (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 7)	7,175

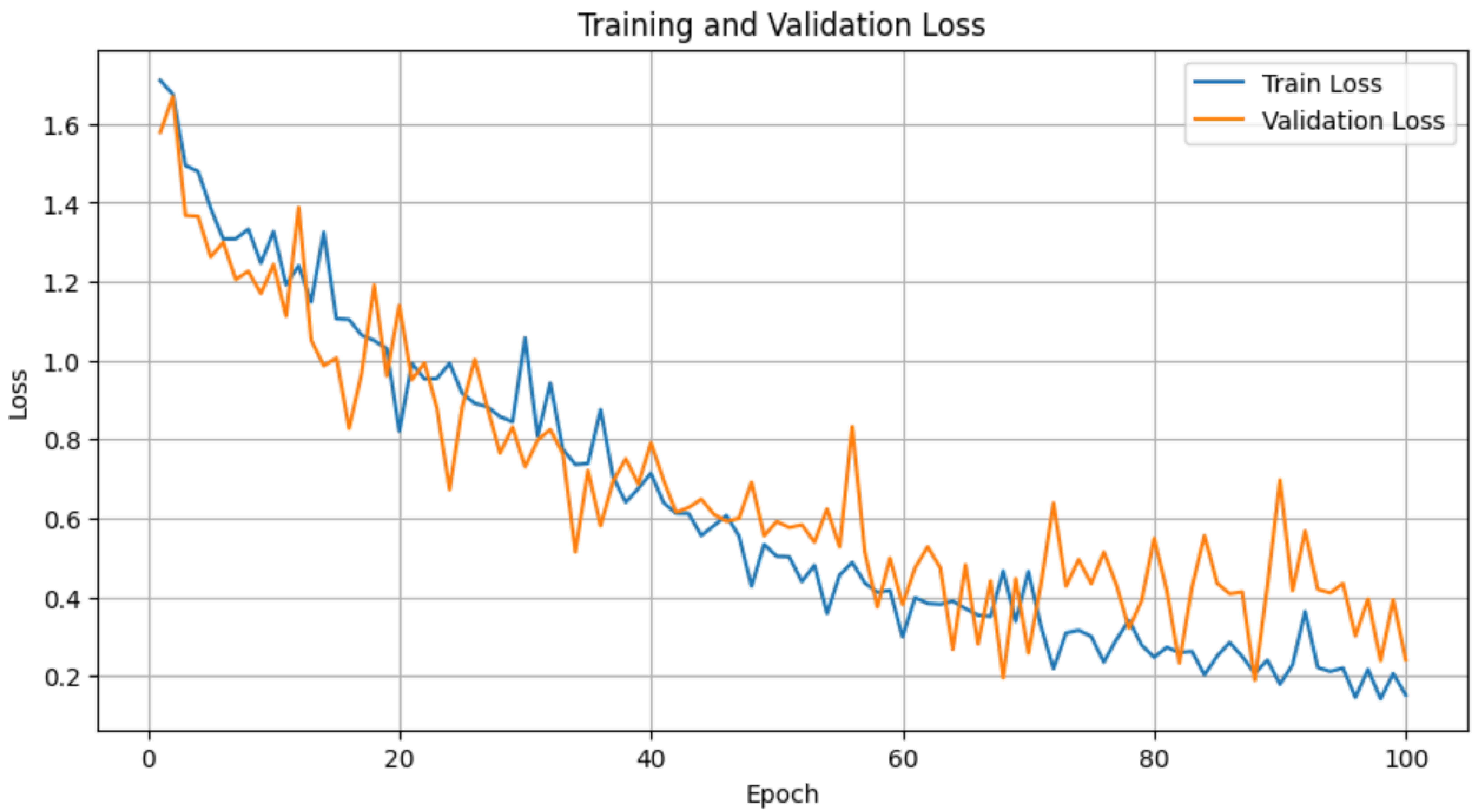
CNN MODEL ACCURACY & LOSS

Epoch	Train Accuracy	Validation Accuracy	Train Loss	Validation Loss
1	0.304773092	0.4250848	1.709854126	1.578348756
2	0.28125	0.377777785	1.674691439	1.669211149
3	0.419279665	0.493580431	1.494337082	1.367712975
4	0.375	0.55555582	1.479401827	1.36563766
5	0.466034502	0.536882281	1.386543036	1.262283206
6	0.515625	0.51111114	1.308107853	1.29954195
7	0.501932323	0.558624029	1.30832541	1.205394268
8	0.484375	0.55555582	1.332825422	1.226353168
9	0.524981081	0.570736408	1.246869564	1.169461846
10	0.515625	0.55555582	1.327034712	1.243700027
11	0.546655238	0.597020328	1.192016602	1.112165809
12	0.578125	0.466666669	1.240269661	1.38802588
13	0.565879107	0.618095934	1.148187876	1.051739454
14	0.5	0.688888907	1.325219631	0.986754119
15	0.584525287	0.640746117	1.105885267	1.006663084
16	0.671875	0.733333349	1.104039907	0.827749789
17	0.599764943	0.653221905	1.064158678	0.966632366
18	0.53125	0.600000024	1.050678015	1.191040158
19	0.616000652	0.658975303	1.030366778	0.960377991
20	0.6875	0.51111114	0.820915759	1.139821529
21	0.627793908	0.659217536	0.992973149	0.950944662
22	0.703125	0.600000024	0.953301609	0.994026184
23	0.646479964	0.689801335	0.954563379	0.878046632
24	0.671875	0.75555557	0.993115187	0.672986448
25	0.661978543	0.685259223	0.916791916	0.882054329
26	0.65625	0.55555582	0.891499341	1.003108025
27	0.674847603	0.68786335	0.882612348	0.880008757
28	0.65625	0.75555557	0.857685685	0.765174091
29	0.689509571	0.707243204	0.844859302	0.831291676
30	0.578125	0.75555557	1.05712676	0.730400026
31	0.703016043	0.721051335	0.80820334	0.798812985
32	0.65625	0.688888907	0.942984819	0.824988484
33	0.718016624	0.737948179	0.774552464	0.763585687
34	0.765625	0.888888896	0.736437798	0.514849961
35	0.732280195	0.751332343	0.739037156	0.721456587
36	0.671875	0.800000012	0.87552309	0.580851853
37	0.748217046	0.761082828	0.703720987	0.694693923
38	0.78125	0.777777791	0.641007602	0.751057625
39	0.753735185	0.767260194	0.675596535	0.686258614
40	0.78125	0.711111128	0.714010417	0.791776717
41	0.770449042	0.771196723	0.639924407	0.697221935
42	0.75	0.75555557	0.61255312	0.615292549
43	0.779254138	0.794331372	0.612069845	0.627309918
44	0.8125	0.777777791	0.556472838	0.648533404
45	0.791784525	0.801659405	0.581163526	0.61035496
46	0.734375	0.844444454	0.607616901	0.59235996
47	0.801406443	0.809895813	0.555095196	0.600041807
48	0.859375	0.777777791	0.427790403	0.691547096
49	0.81011194	0.825460255	0.53369689	0.555965424
50	0.859375	0.822222233	0.504022419	0.591795206
51	0.82027173	0.823280036	0.502518058	0.576679647
52	0.859375	0.800000012	0.439795017	0.583554208
53	0.829495192	0.835816383	0.480950892	0.539642036
54	0.875	0.866666675	0.359078348	0.623778999
55	0.838380039	0.841085255	0.456157923	0.527764976
56	0.78125	0.733333349	0.488246232	0.832450867
57	0.8458305	0.847262621	0.436525762	0.51537782
58	0.859375	0.911111116	0.412648082	0.375300854
59	0.852065802	0.855075121	0.417551249	0.499516696
60	0.875	0.911111116	0.299813151	0.381594688
61	0.860671759	0.860767901	0.398917794	0.474935293
62	0.859375	0.822222233	0.384474307	0.528068483
63	0.867126167	0.863493204	0.382121682	0.474591374
64	0.84375	0.911111116	0.390084416	0.267913789
65	0.867962897	0.863856614	0.371101856	0.482068956
66	0.84375	0.888888896	0.354836762	0.281323612
67	0.875453234	0.876574636	0.351428807	0.441433936
68	0.84375	0.955555558	0.466914117	0.195651814
69	0.880035043	0.876211226	0.338691682	0.447104752
70	0.765625	0.933333337	0.465519607	0.258452803
71	0.883939624	0.880147755	0.326738656	0.433215111
72	0.90625	0.822222233	0.219312027	0.639149785
73	0.891549468	0.883115292	0.309543341	0.42786935
74	0.859375	0.888888896	0.316453457	0.495753765
75	0.894258738	0.879299879	0.30094555	0.434490055
76	0.921875	0.866666675	0.235777259	0.514509082
77	0.897366405	0.885598361	0.292971134	0.431032836
78	0.859375	0.933333337	0.341484636	0.321213245
79	0.900852621	0.894561529	0.279102653	0.390987724
80	0.9375	0.844444454	0.247710735	0.549581766
81	0.904836833	0.885719478	0.27321133	0.418071598
82	0.890625	0.955555558	0.259927809	0.233234659
83	0.907486379	0.886325121	0.262831181	0.425196052
84	0.9375	0.888888896	0.203423709	0.5566715
85	0.912347078	0.886385679	0.250550389	0.436149299
86	0.890625	0.888888896	0.285589099	0.408750713
87	0.912845135	0.891109467	0.249787107	0.413591772
88	0.921875	0.911111116	0.207536981	0.189537466
89	0.916112185	0.888808131	0.240598455	0.429256171
90	0.9375	0.800000012	0.179587662	0.697007537
91	0.920176089	0.889716566	0.229009897	0.417151511
92	0.859375	0.866666675	0.364116609	0.568259656
93	0.923383415	0.890443325	0.22235395	0.420200288
94	0.953125	0.866666675	0.211925045	0.411051929
95	0.922925234	0.888323665	0.220903412	0.435141712
96	0.9375	0.911111116	0.146255016	0.301826596
97	0.924200177	0.896257281	0.217272878	0.395769656
98	0.96875	0.933333337	0.142355531	0.238885418
99	0.92874217	0.901102245	0.206511527	0.393803477
100	0.953125	0.933333337	0.152340397	0.241556421

VALIDATION ACCURACY



VALIDATION LOSS



CONCLUSION AND INFERENCE

In this project, we implemented a system that can predict emotions from facial expressions in human images. An additional component that predicts facial expressions in videos and in real-time using webcams was also developed. . Loss V/S Epochs We explored various techniques to achieve the best results. Initially, we experimented with basic models:

1) Decision Tree
2) Random Forest
3) AdaBoost

Finally, we proposed a CNN model that performed well with our dataset and provided good results in real-time prediction. The maximum validation accuracy achieved for facial emotion recognition was approximately 95.8% with our model. We chose to use CNN instead of conventional fully connected Multi-layer perceptron-based (MLP) networks for several reasons:

1) The number of weights in MLP grows very rapidly.
2) MLP is not transition invariant.
3) Spatial information is lost when we flatten the input.
4) CNNs preserve spatial information. Overall, CNNs work well with image classification problems, and one can explore their architecture further to increase accuracy

MODELS TESTING AND PREDICTION

1.PHOTOS

2. VIDEO

3. LIVE WEBCAM

MODEL TESTING AND OUTPUT PREDICTION

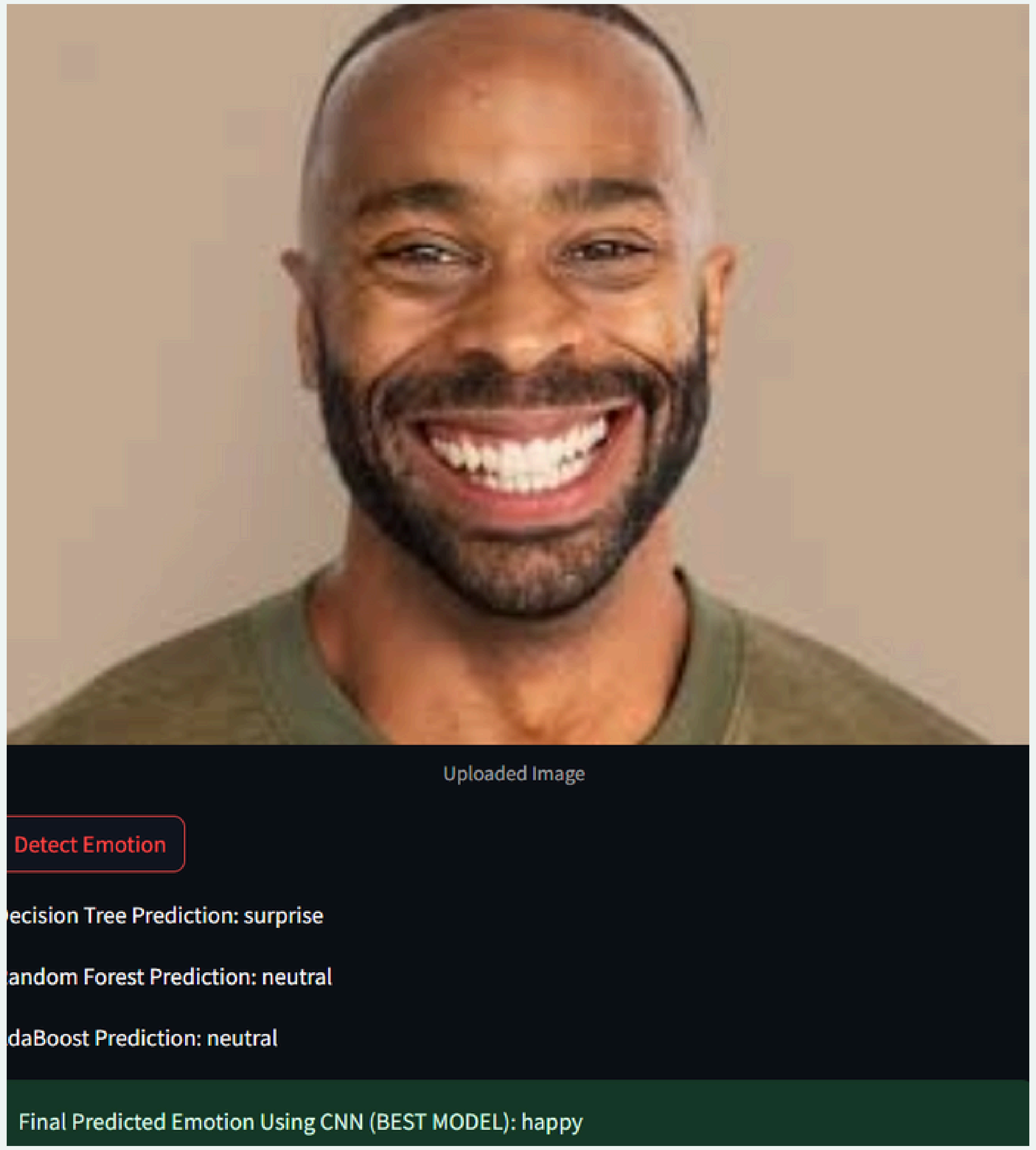
Our facial expression recognition system is versatile and capable of predicting emotions from various sources. Here are the three primary modes of prediction:

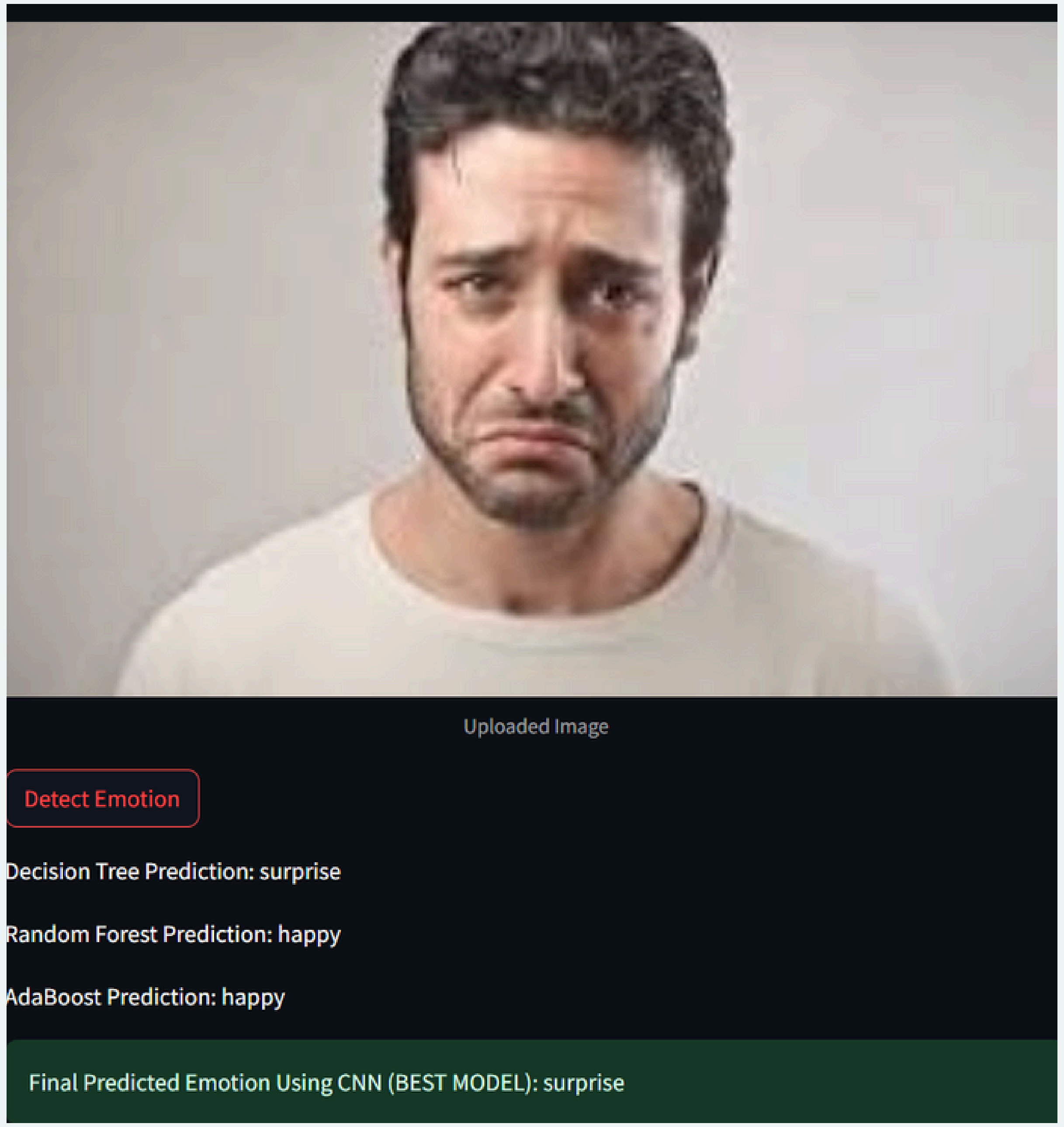
1. Image Prediction:
    - Users can upload static images containing faces.
    - The system analyzes these images and predicts the corresponding emotions (e.g., happy, sad, surprised).

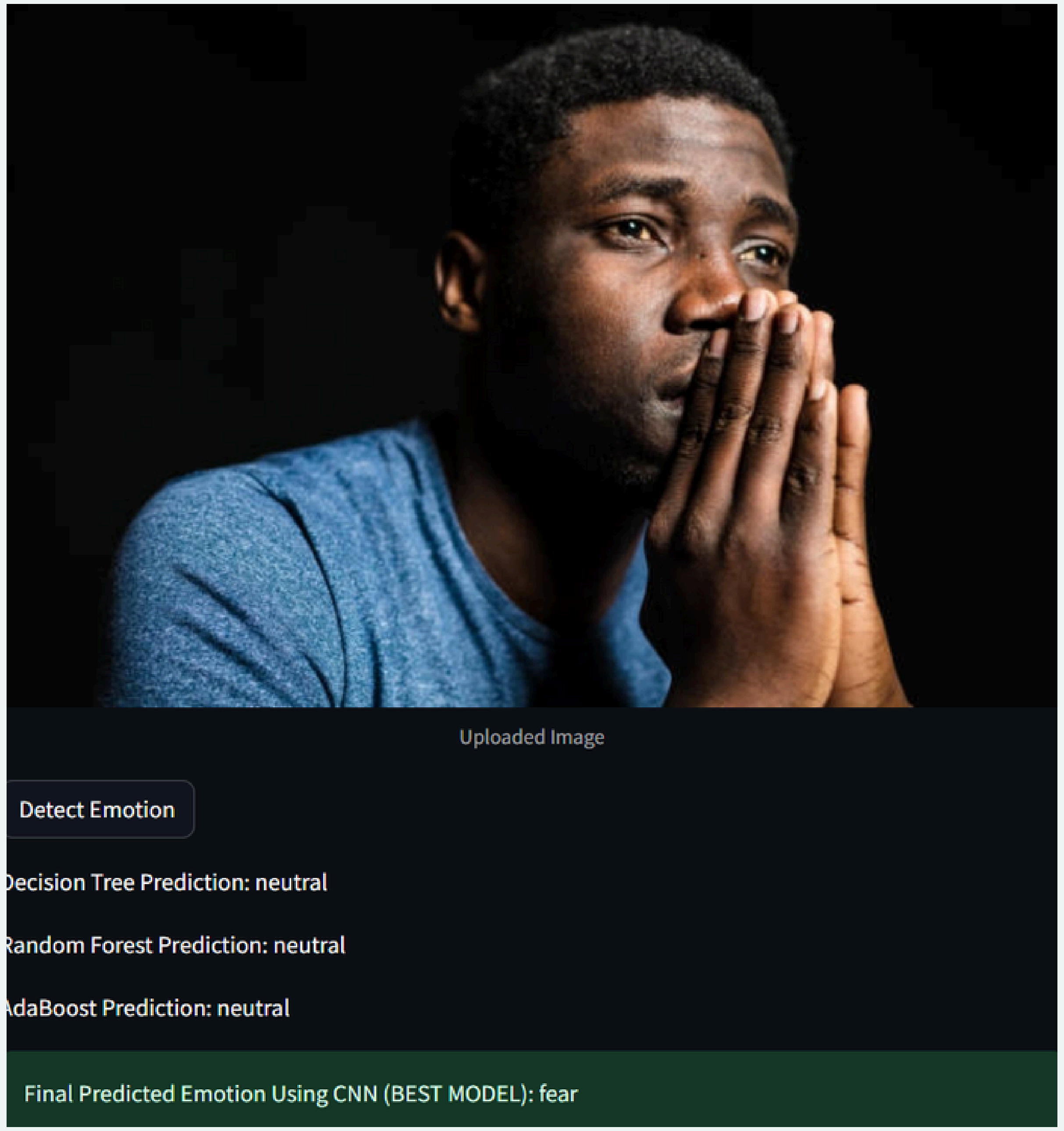
2. Video Prediction:
    - Users can provide video clips or sequences.
    - The system processes each frame and predicts emotions over time.
    - This mode is useful for analyzing emotional changes during conversations, presentations, or events.

3. Webcam Prediction:
    - Users can use their webcam in real-time.
    - The system captures live video frames and predicts emotions on the fly.
    - It's an interactive way to explore emotions during video calls or streaming.

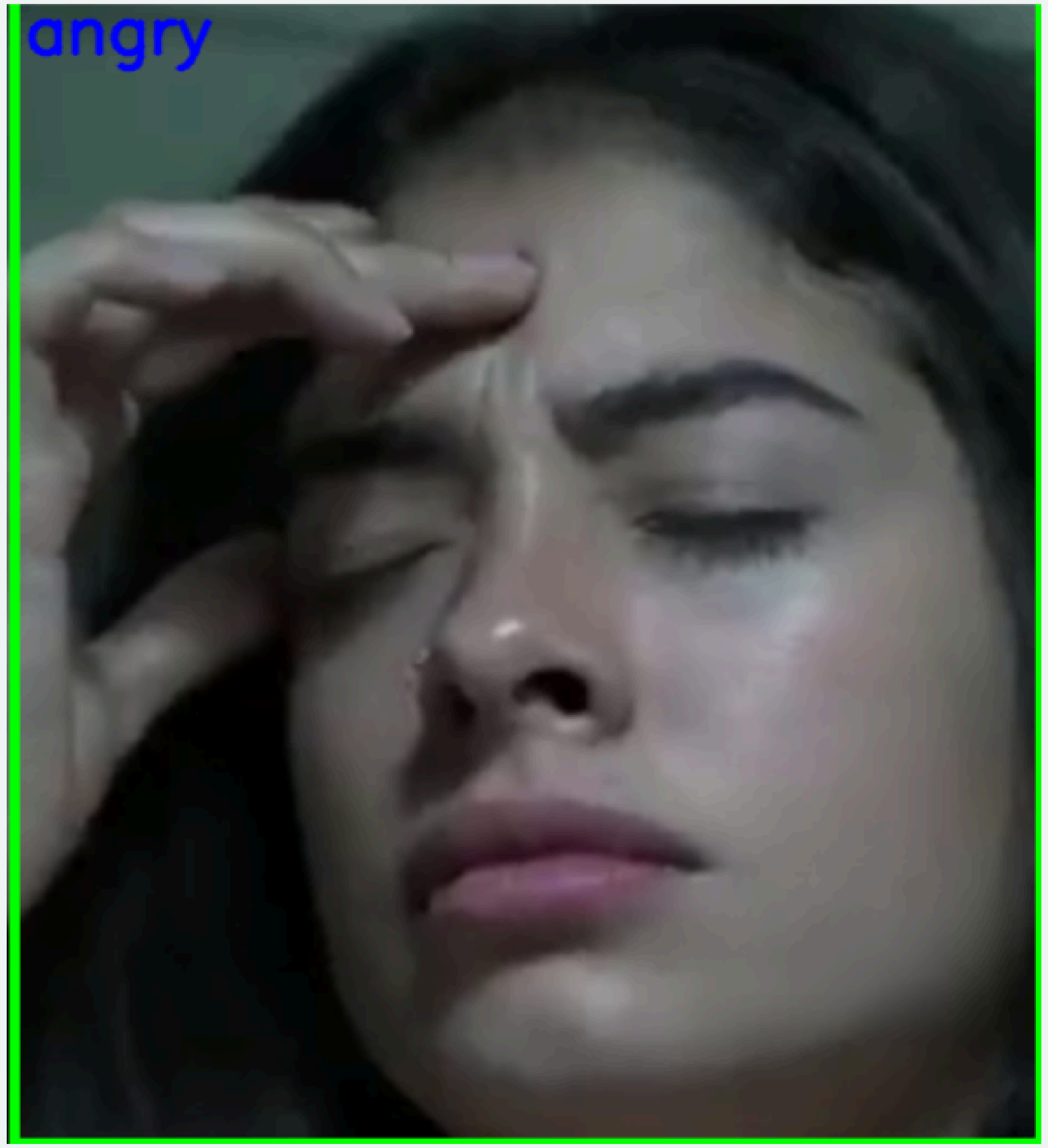
PHOTOS


Uploaded Image
Detect Emotion
ecision Tree Prediction: surprise
andom Forest Prediction: neutral
daBoost Prediction: neutral
Final Predicted Emotion Using CNN (BEST MODEL): happy

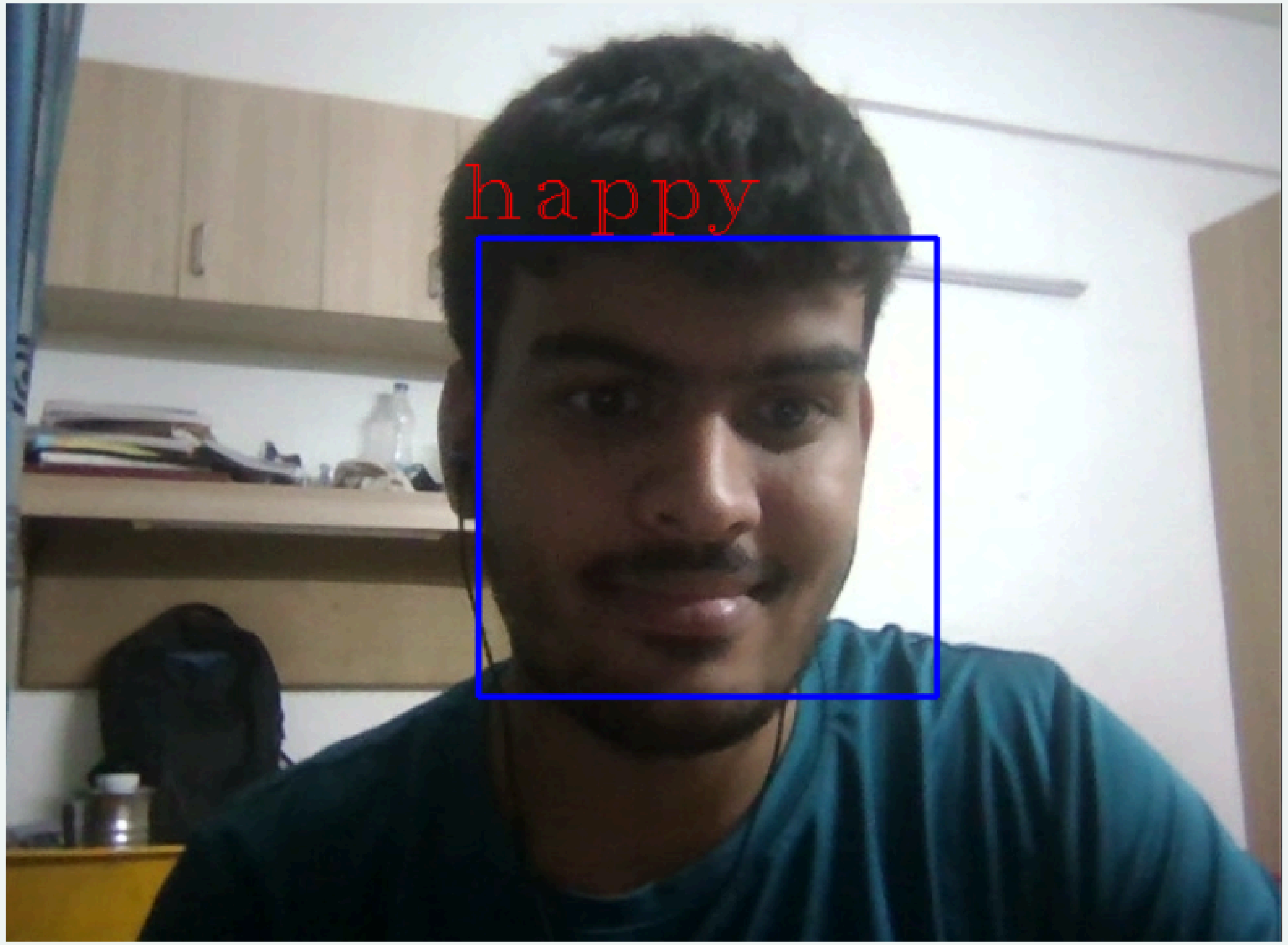

Uploaded Image
Detect Emotion
Decision Tree Prediction: surprise
Random Forest Prediction: happy
AdaBoost Prediction: happy
Final Predicted Emotion Using CNN (BEST MODEL): surprise

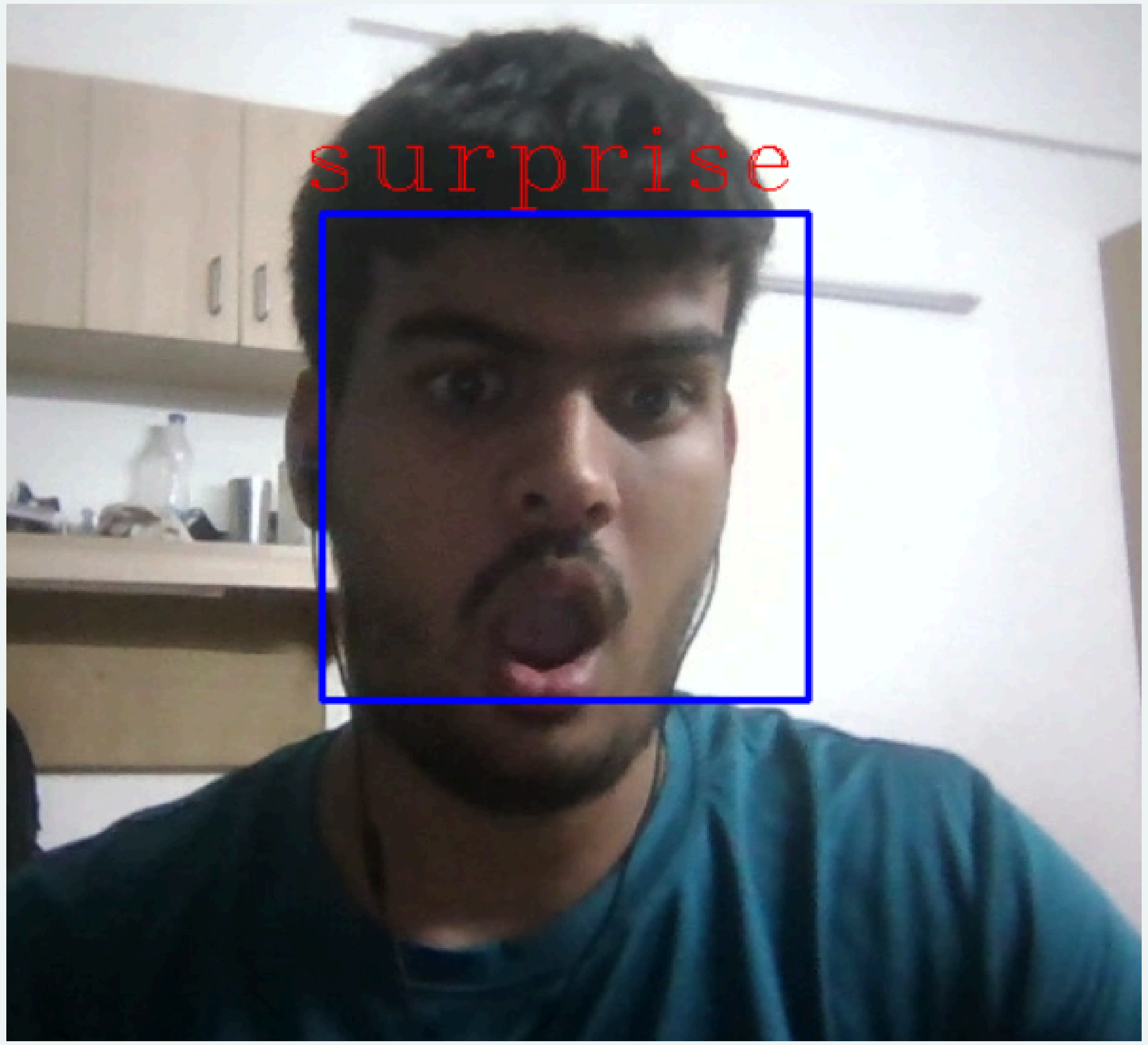

Uploaded Image
Detect Emotion
Decision Tree Prediction: neutral
Random Forest Prediction: neutral
AdaBoost Prediction: neutral
Final Predicted Emotion Using CNN (BEST MODEL): fear

VIDEO



LIVE WEBCAM

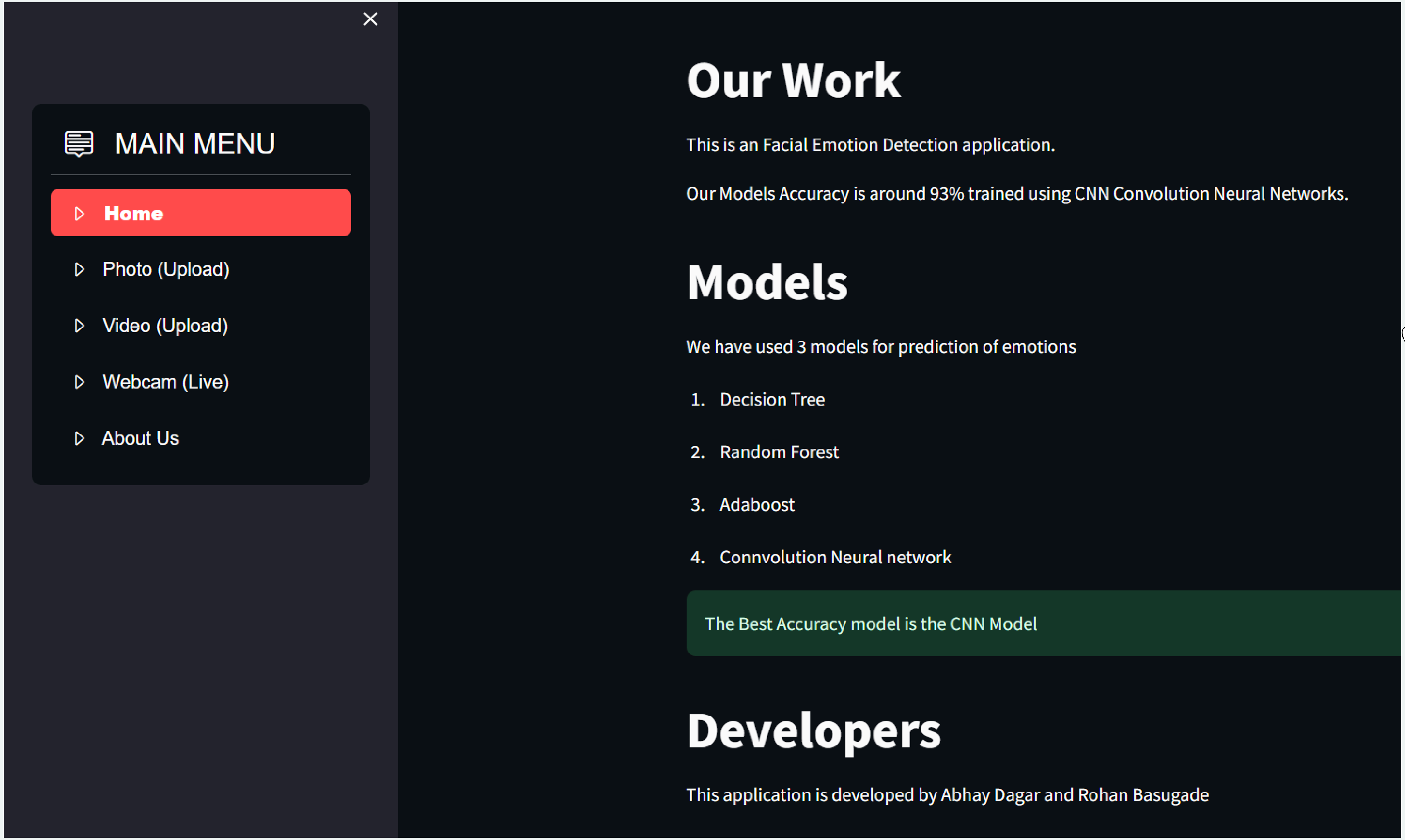




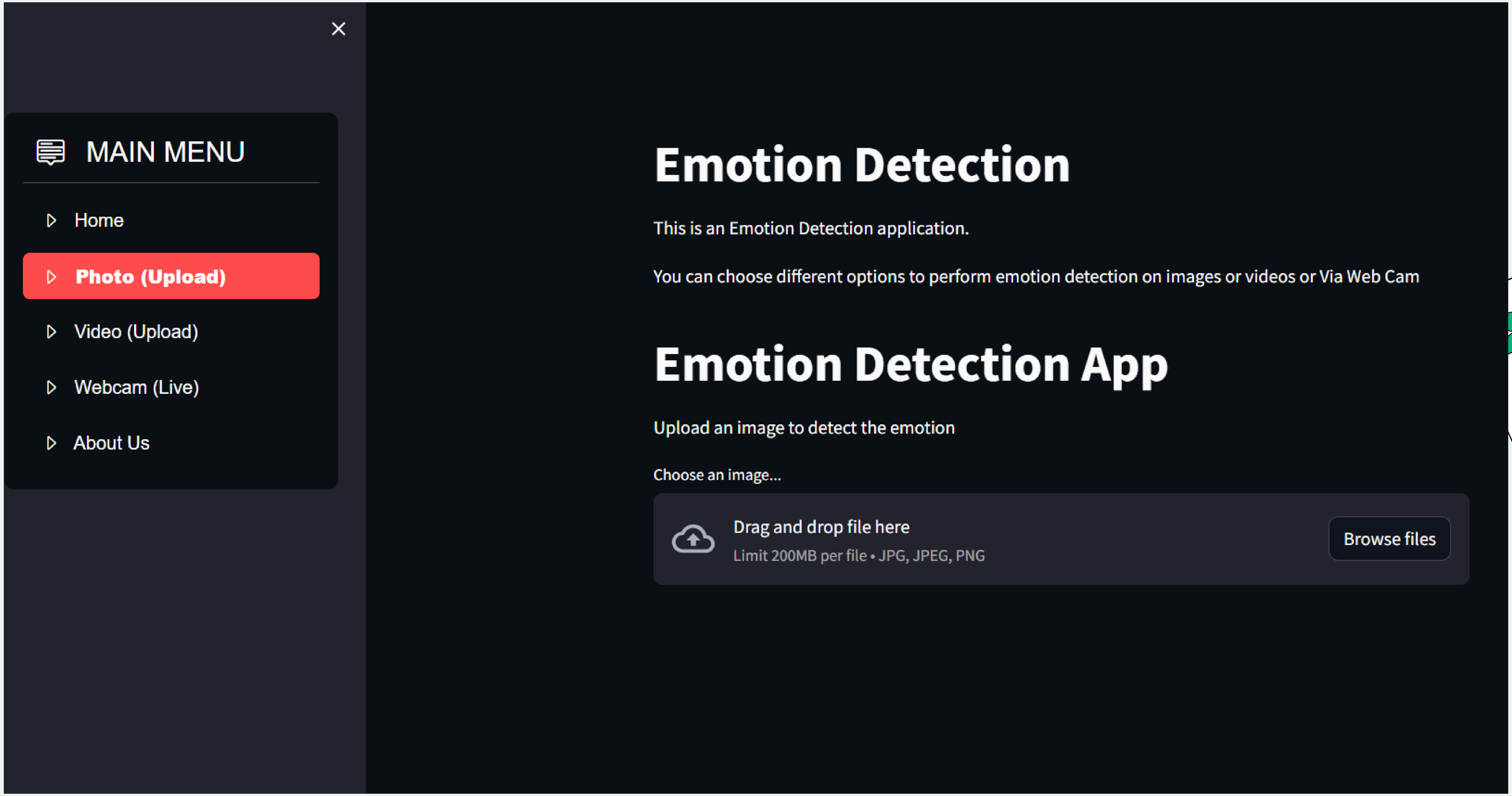
USER-FRIENDLY INTERFACE WITH STREAMLIT?

1. TO ENHANCE USER EXPERIENCE, WE'VE DEVELOPED AN INTERACTIVE FRONT END USING STREAMLIT.
2. STREAMLIT PROVIDES AN INTUITIVE INTERFACE FOR USERS TO UPLOAD IMAGES, INPUT VIDEOS, OR ACCESS THEIR WEBCAM.
3. IT SEAMLESSLY INTEGRATES WITH OUR FACIAL EXPRESSION RECOGNITION SYSTEM, MAKING IT ACCESSIBLE AND USER-FRIENDLY.

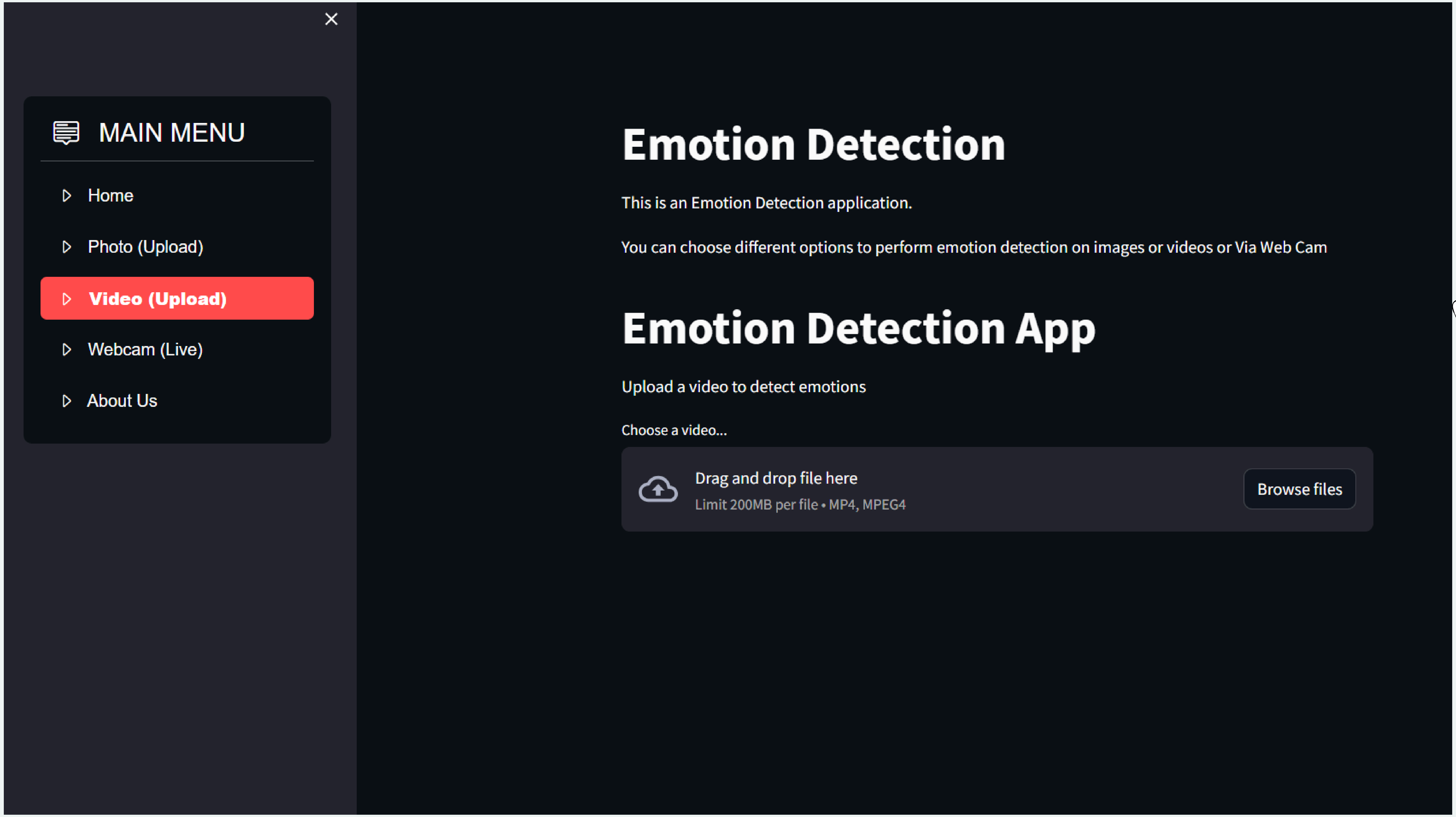
INTERFACE (HOME)



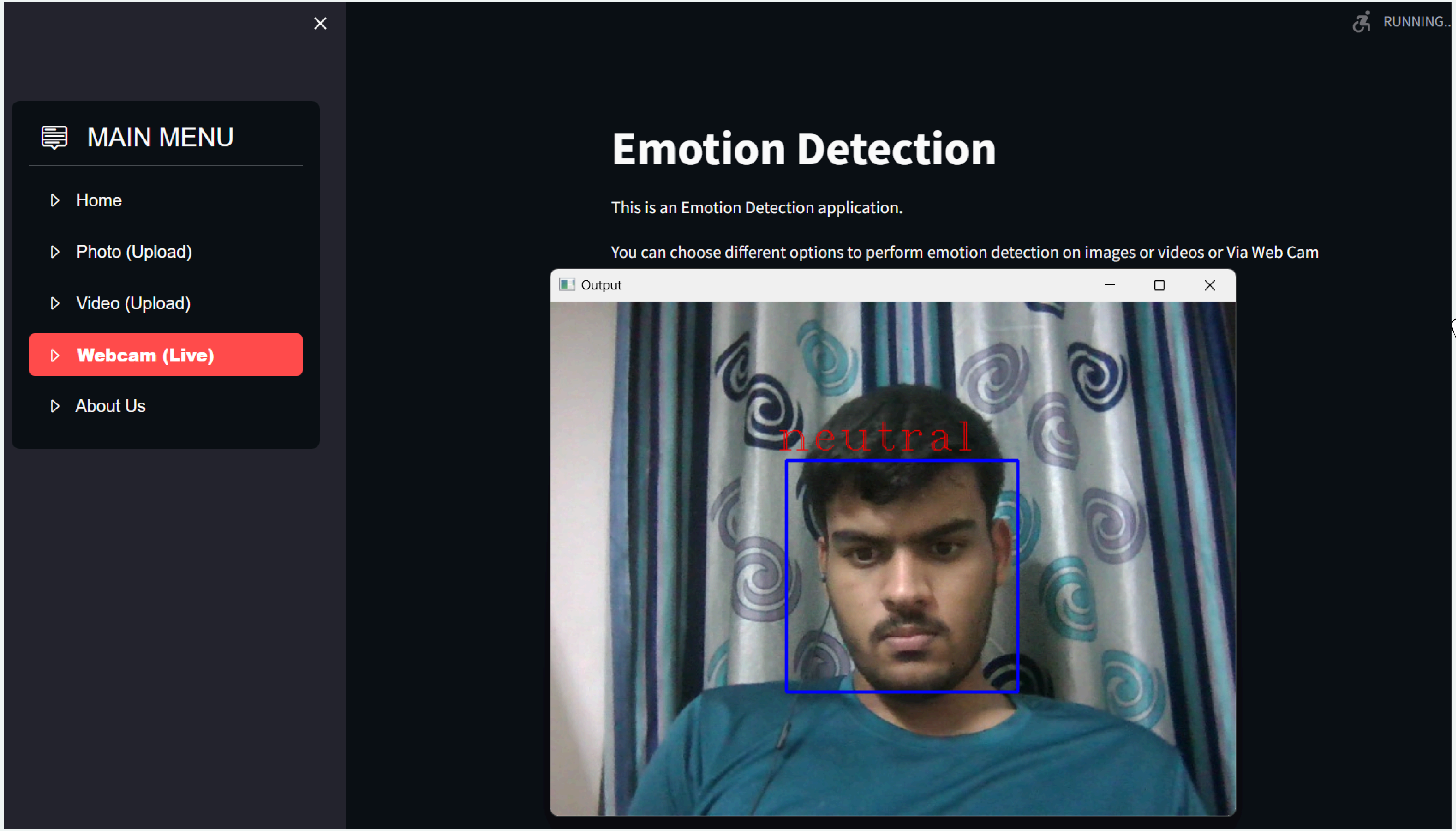
INTERFACE (PHOTO)



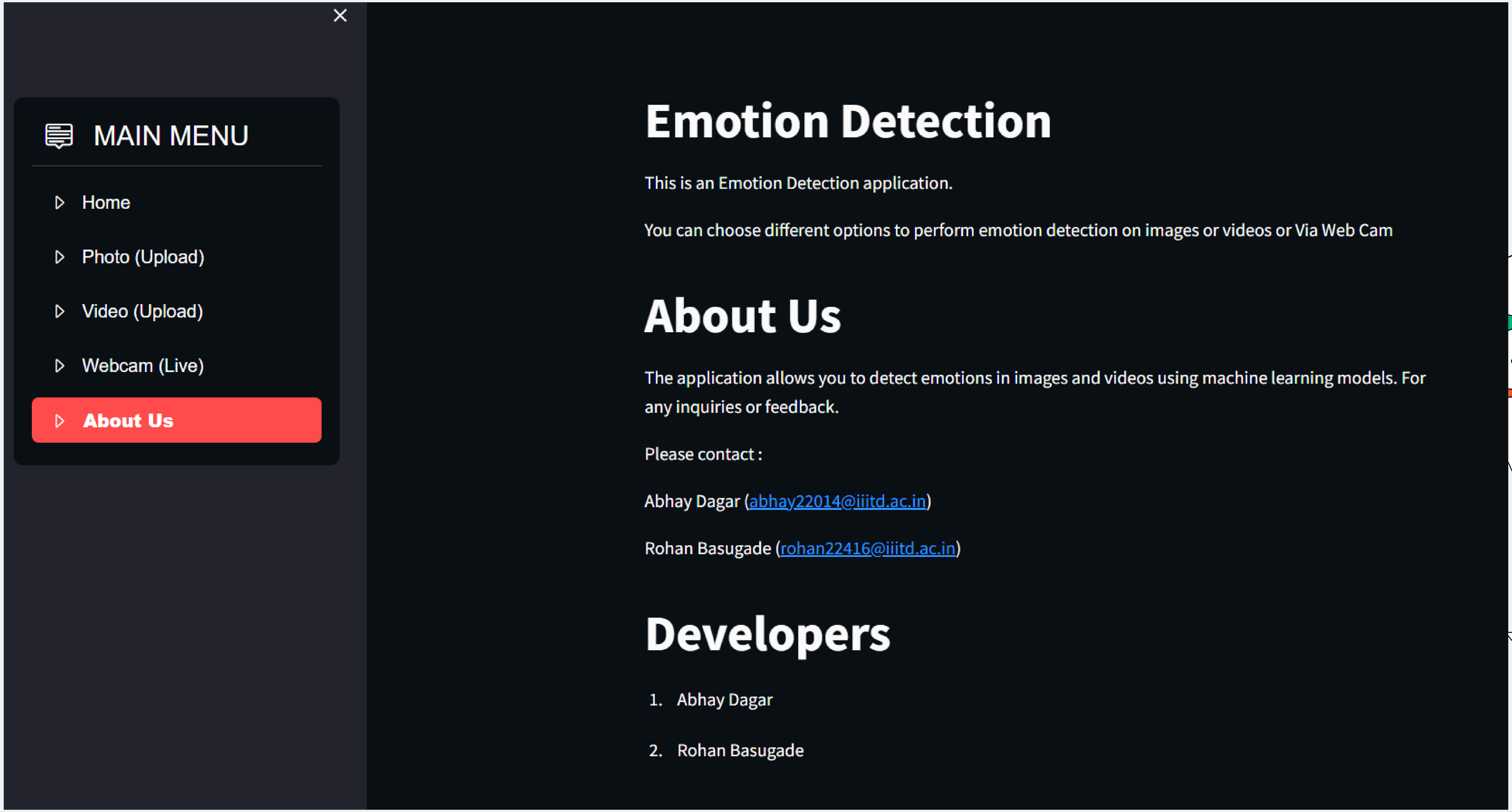
INTERFACE (VIDEO)



INTERFACE (LIVE WEBCAM)



INTERFACE(ABOUT US)



RESOURCE PAGE

[1] https://www.kaggle.com/datasets/msambare/fer2013

[2] https://www.kaggle.com/datasets/aadityasinghal/facial-expressiondataset

[3] M. A. Ozdemir, B. Elagoz, A. Alaybeyoglu, R. Sadighzadeh and A. Akan, "Real Time Emotion Recognition from Facial Expressions Using CNN Architecture,"

[4] Gaurav Sharma, Real Time Facial Expression Recognition, article on Medium.

[5] Introduction to CNN, GeeksforGeeks. Available at: https://www.geeksforgeeks.org/introduction-convolution-neuralnetwork/

[6] Keras CNN, JavaTPoint. Available at: https://www.javatpoint.com/keras-convolutional-neural-network