MELANOMA SKIN CANCER DETECTION USING STACKED ENSEMBLE MODEL

PRS Practice Module Report
by

GROUP 10

Apar Garg (A0231539E)
Gopan Ravikumar Girija (A0231541U)
Sarah Elita Shi Yuan Wong (A0231507N)

SUBMITTED TO

INSTITUTE OF SYSTEMS SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

in

Partial fulfilment of the requirements for

Graduate Certificate in Pattern Recognition Systems

NOVEMBER 2021

TABLE OF CONTENTS

TABLE OF CONTENTS ii

LIST OF TABLES v

LIST OF FIGURES vi

LIST OF ABBREVIATIONS vii

1. INTRODUCTION 1

1.1. Background 1

1.2. Business Case 1

1.3. Solution 2

2. TOOLS & TECHNIQUES 3

2.1. Machine Learning Framework 3

2.1.1. TensorFlow & Keras 3

2.1.2. Scikit-Learn 3

2.2. Image Processing 4

2.2.1. OpenCV 4

2.2.2. Skimage 4

2.3. Visualization 4

2.3.1. Matplotlib 4

2.4. Front-end Development Framework 5

2.4.1. Streamlit 5

3. Datasets 5

4. Proposed Methodology 6

4.1. Skin Lesion Segmentation 7

4.1.1. Network Architecture 7

4.1.2. Training 7

4.1.3. Data Augmentation 9
4.2. Base Learners 9
4.2.1. Center Base Learner 9
4.2.2. Border Base Learner 11
4.2.3. Asymmetry Base Learner 13
4.2.4. Whole Image Base Learner 15
4.2.5. Blue-White Veil Base Learner 16
4.3. Arbitrator 17
4.3.1. Final Arbitrator (Neural Network) 17
4.3.2. Logistic Regression Arbitrator 19
4.3.3. Support Vector Machine (SVM) Arbitrator 19
5. Results 19
5.1. Skin Lesion Segmentation 20
5.2. Individual Base Learner Performance 20
5.3. Arbitrator performance 21
6. User Interface 23
6.1. UI Description 23
6.1.1. Features 23
6.1.2. User Guide 24
6.1.3. To run on Local Machine (Windows) 24
6.2. UI Mockup 25
7. Conclusion and Future Work 26
7.1. Conclusion 26
7.2. Future Work 27
REFERENCES 28

APPENDIX A: Classification reports for Neural Network, Logistic Regression, SVM arbitrators 30

APPENDIX B: Melanoma classification model trained on patient-level contextual information..... 31

LIST OF TABLES

Table Page

Table 1: U-net Segmentation network performance in the test dataset 20
Table 2: Overall accuracy, precision, recall, and f1 score achieved in test data 20
Table 3: Precision, recall, and f1 score for melanoma class achieved in test data 21
Table 4: Overall precision, recall, and f1 score for blue-white veil base learner 21
Table 5: Accuracy, Precision (Melanoma), Recall (Melanoma), Precision (Overall), Recall (Overall) for arbitrators 22

LIST OF FIGURES

Figure Page

Figure 1: Diagnosis procedure for melanoma 1
Figure 2: Proposed solution 2
Figure 3 (left to right): ISIC_0024393 (non-melanoma) and ISIC_0026909 (melanoma) 5
Figure 4: Overall Proposed Architecture 6
Figure 5: U-net Segmentation Network Architecture 7
Figure 6: Jaccard Loss Function 8
Figure 7: Training and validation loss for U-net segmentation Network 8
Figure 8: ISIC_0026506 (melanoma lesion) segmentation mask, original image, centre cropped image (from left to right) 10
Figure 9: Accuracy and loss curves for centre base learner 11
Figure 10: ISIC_0026506 (melanoma lesion) segmentation mask, border mask, original image, border cropped image (from left to right) 12
Figure 11: Accuracy and loss curves for border base learner 12
Figure 12: ISIC_0026506 (melanoma lesion) segmentation mask, original image, step-1 image, asymmetry processed image (from left to right) 14
Figure 13: Accuracy and loss curves for asymmetry base learner 14
Figure 14: Accuracy and loss curves for original image base learner 15
Figure 15: Accuracy and loss curves for Blue-White-Veil base learner 16
Figure 16: Loss curve for Final Arbitrator 18
Figure 17: Accuracy curve for Final Arbitrator 18
Figure 18: ROC Curve comparison across different arbitrators 22
Figure 19: Precision-Recall curves of arbitrators 23
Figure 20: Landing page of UI 25
Figure 21: UI showing results of the prediction 26

LIST OF ABBREVIATIONS

API	Application Programming Interface
AUC	Area Under Curve
GUI	Graphical User Interface
FN	False Negative
FP	False Positive
ML	Machine Learning
ReLU	Rectified Linear Unit
ROC	Receiver Operator Characteristic
SGD	Stochastic Gradient Descent

1. INTRODUCTION

1.1. Background

There are over 200 different forms of skin cancer. Melanoma is the deadliest of them all. Although it is responsible for only about 1% of all skin cancers diagnosed in the U.S., it makes up most skin cancer deaths [1].

The diagnosis procedure for melanoma is illustrated below:

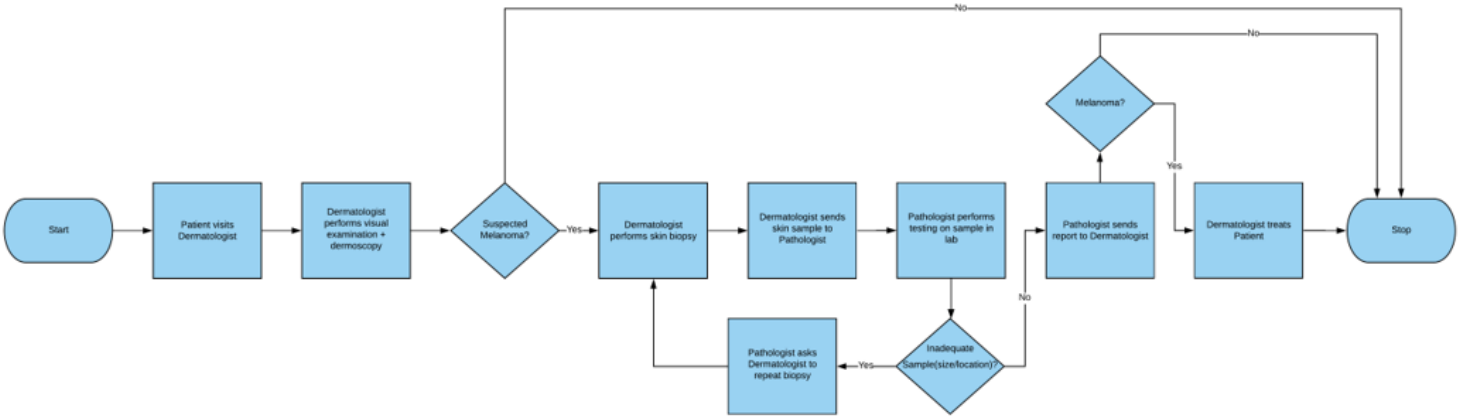


Figure 1: Diagnosis procedure for melanoma

The diagnostic procedure for melanoma starts with clinical screening, followed by dermoscopic analysis and histopathological examination. Early diagnosis and treatment of melanoma give patients a significantly higher chance of survival.

1.2. Business Case

There are many gaps in the diagnostic procedure for melanoma:

1. False Negatives (FN) – The patient has melanoma, but the dermatologist concludes otherwise based on the dermoscopy he has performed.
    - Dermatologist only sends suspected cases for lab-testing. Since the sensitivity of dermoscopy can range from 60% to 100% [2], some unsuspected cases, which are in

fact melanoma, will be left out for lab-testing. This may cause late diagnosis and treatment.

2. False Positives (FP) – The patient does not have melanoma, but the dermatologist suspects it as melanoma and performs biopsy as a further check.
    - Skin biopsy is an invasive (removal of a small part of the skin) process, which can leave a scar [3].
    - The typical cost of a skin biopsy without insurance is 120 – 450 USD. Lab evaluation fees may add extra fees from 50 – 350 USD [3].

1.3. Solution

This project aims to build an automated classification system based on deep learning techniques to predict the presence of melanoma skin cancer using skin lesions images. The solution will aid dermatologists in detecting melanoma during the early screening stage, thereby, reducing False Negatives (FN) and False Positives (FP).

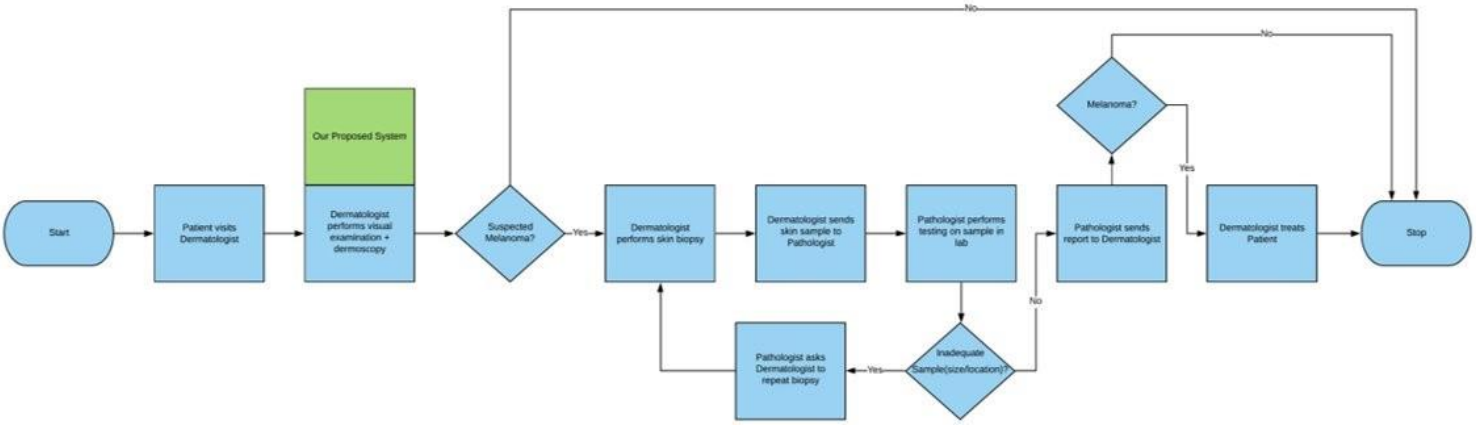


Figure 2: Proposed solution

To make it accessible to the dermatologist, the system will be deployed on an easy-to-use website. The dermatologist will upload the patient demographic information along with an image of the skin lesion. With the image, the model will analyse the data and return the results within a minute. The patient demographic information along with the prediction results from the model will be used

to generate a detailed report. The report can be stored locally on the machine and emailed for future reference.

2. TOOLS & TECHNIQUES

2.1. Machine Learning Framework

2.1.1. TensorFlow & Keras

TensorFlow is an end-to-end open-source platform for machine learning. It contains an extensive, versatile ecosystem of tools, libraries, and community resources that allows users to easily build and deploy ML-powered applications [4].

Keras is the high-level API of TensorFlow 2 written in Python. It was developed with a focus on enabling fast experimentation with deep neural networks. It provides essential abstractions and building blocks for developing and shipping machine learning solutions which makes it simpler to use than TensorFlow [5].

In this project, we have used Keras to build the convolutional neural network models for the base learners and final arbitrator. Some Keras APIs that we used are:

- ImageDataGenerator for image data augmentation.
- DenseNet121 for loading pre-trained model architecture and weights.
- Layers (e.g., Dense, GlobalAveragePooling2D, Input, Dropout, Conv2D, BatchNormalization) for creating custom layers in the neural network.
- Model to instantiate the models.
- Optimizers (e.g., SGD, Adam) to adjust the learning parameters of the models.

2.1.2. Scikit-Learn

Scikit-learn is an open-source machine learning library built on NumPy, SciPy, and matplotlib. It supports supervised and unsupervised learning and provides various tools for model fitting, data pre-processing, model selection and evaluation, and many other utilities [6].

In this project, we have used Scikit-Learn to evaluate the performance of each model and tune the model parameters accordingly. We have also experimented with the Logistic Regression model and Support Vector Machine as an arbitrator to combine the base model outputs. Some Scikit-Learn APIs that we used for model evaluation are accuracy_score, confusion_matrix, precision_recall_curve, and classification_report.

2.2. Image Processing

2.2.1. OpenCV

Image processing is aimed to help the computer understand the content of an image. OpenCV is a library of programming functions mainly used for image processing. It provides de-facto standard API for computer vision applications [7]. This library was used to perform several image processing techniques like augmentation in our project.

2.2.2. Skimage

It is an image processing library that implements algorithms and utilities for use in research, education, and industry applications. It is released under the liberal "Modified BSD" open-source license, provides a well-documented API in the Python programming language, and is developed by an active, international team of collaborators [8].

2.3. Visualization

2.3.1. Matplotlib

Matplotlib is a portable 2D plotting and imaging package aimed primarily at the visualization of scientific, engineering, and financial data. matplotlib can be used interactively from the Python shell, called from python scripts, or embedded in a GUI application (GTK, Wx, Tk, Windows). Many popular hardcopy outputs are supported including JPEG, PNG, PostScript, and SVG. Features include the creation of multiple axes and figures per page, interactive navigation, many predefined line styles, and symbols, images, antialiasing, alpha blending, and financial plots, W3C compliant font management, and FreeType2 support, legends and tables, pseudocolour plots, mathematical text and more. It works with both NumPy array and Numeric [9].

2.4. Front-end Development Framework

2.4.1. Streamlit

It is an open-source Python framework to create and deploy custom web apps [10].

3. DATASETS

For training and testing our segmentation network, arbitrators, and some of the base learners, we used publicly available data from HAM10000 [11], a well-curated data set of dermoscopy images collected specifically for use in the machine learning context. The HAM10000 dataset consists of 10015 dermoscopy images of size 450 × 600. It consists of 1113 melanoma images and 8902 non-melanoma images. Some randomly selected sample images from the HAM10000 dataset are shown below.

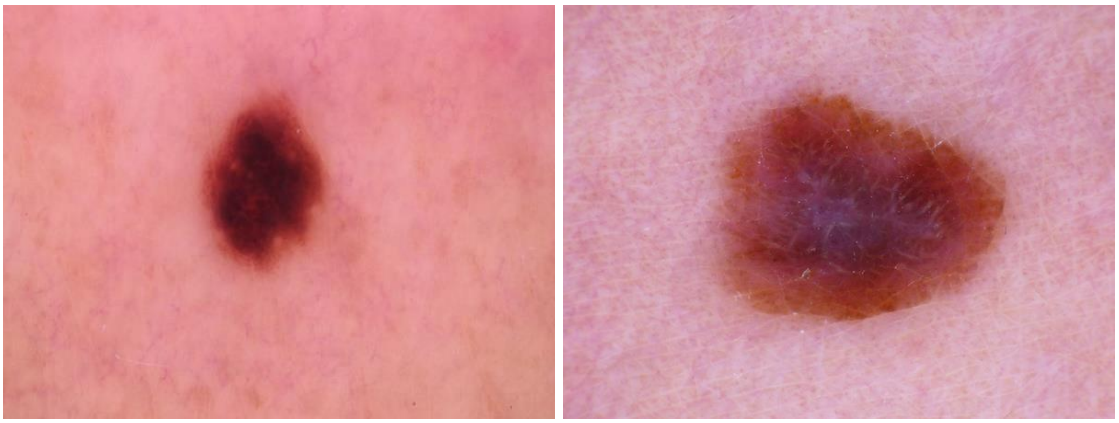


Figure 3 (left to right): ISIC_0024393 (non-melanoma) and ISIC_0026909 (melanoma)

All 4 base learners out of 5 were trained to classify the input as melanoma or non-melanoma. The fifth base learner was trained to identify the presence of or absence of diagnostic criteria: the blue-white veil. This network was trained using the derm7pt dataset [12] which has ground truth label annotation of various features including the blue-white veil. The derm7pt dataset consists of 1011 dermoscopy images of size 512x768. It consists of 195 images having blue-white veil features and 816 images without blue-white veil features.

4. PROPOSED METHODOLOGY

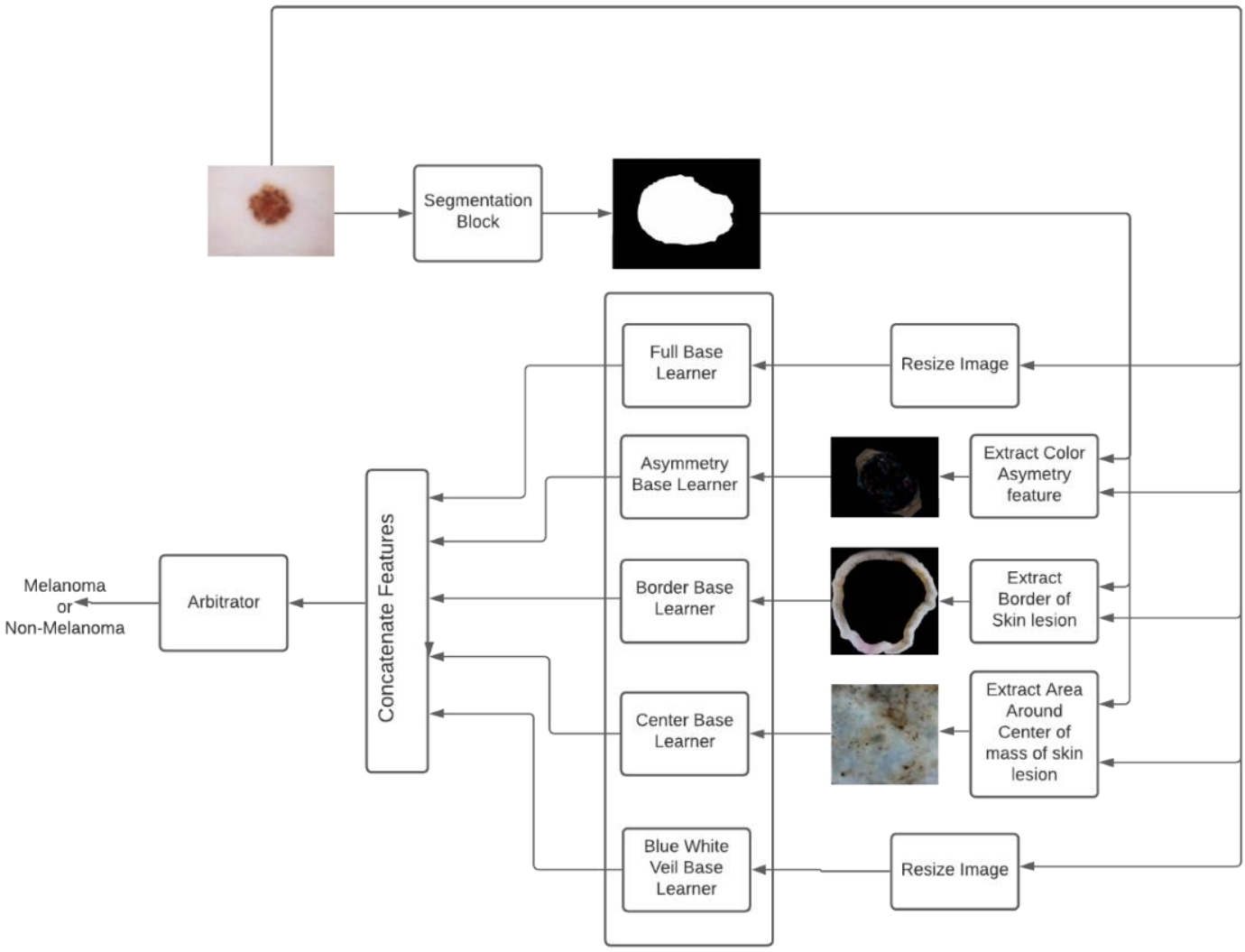


Figure 4: Overall Proposed Architecture

The overall proposed architecture of our system is illustrated in Figure 4. First, a segmentation network is used to separate the skin lesion region from the background. After that, a series of base learners are trained to identify melanoma skin lesions using key visual features that are important for classifying melanoma skin lesions from non-melanoma skin lesions. Finally, the output feature vector of the base learners is combined and passed as input to an arbitrator that makes the final classification. Detailed explanation for each module is provided in below sessions.

4.1. Skin Lesion Segmentation

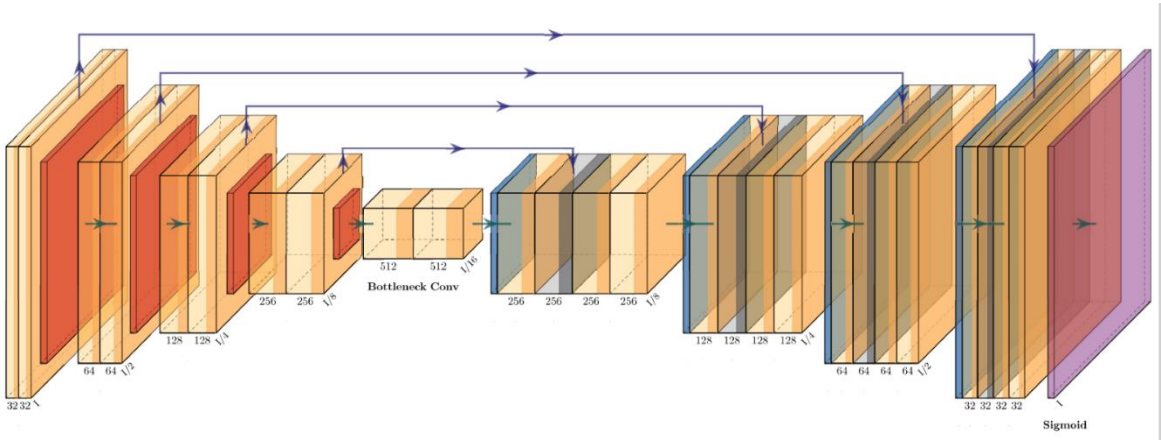


Figure 5: U-net Segmentation Network Architecture

Most of our base learners require the input images to be pre-processed in a specialized way and the foreground and background segmentation of the skin lesion regions is required for that. Here we achieved skin lesion segmentation using a custom U-net convolution network [13] with skip connections.

4.1.1. Network Architecture

The network architecture is illustrated in Fig 5. The network architecture has a contracting path (encoder) followed by an expansive path (decoder). The contracting path consists of repeated application of 2 3x3 convolutions each followed by a ReLU (Rectified Linear Units) activation function and a 2x2 max pooling function with stride 2 for downsampling. The expansive path consists of up-sampling layers using a 3x3 transpose convolution layer and the output of this is concatenated with a correspondingly cropped feature map from the contractive path. The final layer consists of a 1x1 convolution followed by a sigmoid activation function that is used to map the 32 component feature vectors to either foreground or background classes.

4.1.2. Training

The images and corresponding binary segmentation mask in the HAM10000 dataset [11] were used to train the model. The input images and ground truth binary masks were rescaled to 192x256

pixel resolutions before passing as input to the network. We used an Adam optimizer with a learning rate of 0.001 to train the model. The model was trained for 100 epochs or used early stopping monitoring the validation loss with patience 10 epochs. The corresponding training loss and validation loss achieved for the best model is shown in Fig 7.

The model was trained on a couple of loss functions (cross-entropy loss and Jaccard distance) out of which, the model trained on Jaccard distance gave better performance. The Jaccard distance, which measures dissimilarity between sample sets, is complementary to the Jaccard coefficient and is obtained by subtracting the Jaccard coefficient from 1. The Jaccard Index, also known as the Jaccard similarity coefficient, is a statistic used in understanding the similarities between sample sets. The measurement emphasizes the similarity between finite sample sets and is formally defined as the size of the intersection divided by the size of the union of the sample sets. The mathematical representation of the index is shown in Fig 6.

$$JaccardIndex = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

$$JaccardDistance = 1 - JaccardIndex$$

Figure 6: Jaccard Loss Function

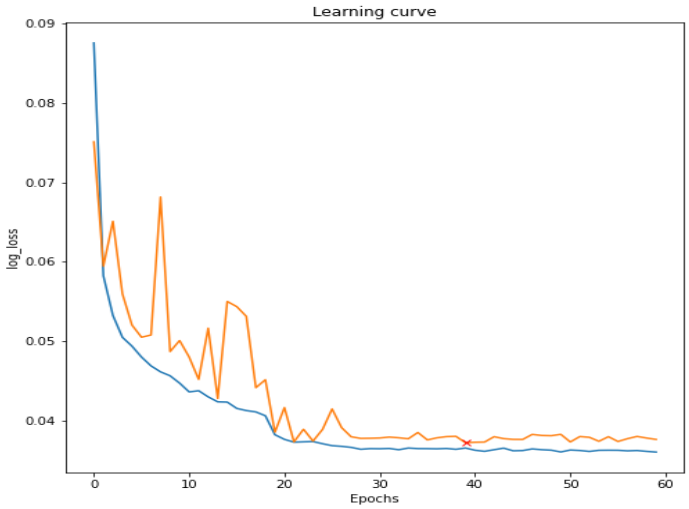


Figure 7: Training and validation loss for U-net segmentation Network

4.1.3. Data Augmentation

Data Augmentation is essential to improve the diversity of training data, especially when training a model with a small dataset. In the case of microscopical images, we primarily need shift and rotation invariance as well as robustness to deformations. To increase the size of training data and to enhance the diversity, we have performed 3 geometrical augmentations (Horizontal flip, Vertical flip, and Random rotation).

4.2. Base Learners

We designed implicit prefiltering methods to emphasize relevant features which expert dermatologists found helpful in decision-making during visual examination [14]. The image-based classifiers take as input either the original image, its subregion, or an image preprocessed in a specialized way. We used DenseNet121 as the feature extractors for base learner classification models that take in the preprocessed image rescaled to 224x224 as input.

4.2.1. Center Base Learner

4.2.1.1. Preprocessing step

Centre cropping was performed on the images to extract a 224-by-224-pixel sized region from the centre of each image. Using the centre portion of images allows the base model to capture fine details in the lesion which may otherwise be lost during image resizing.

The first step was to determine the coordinates for the Centre of Area of the lesion for each image. For this, we used the Python package scikit-image regionprops to obtain the region properties of the image's segmentation mask. This includes the centroids over both x and y axes. The centroids over the axes are taken as the centre coordinates of the lesion and the surrounding 224-by-224 pixels are cropped out of the original image. If the 224-by-224 pixel region fell out of the original image pixel range, the frame was shifted minimally to fit back into the original image. An example of centre cropping is shown below.

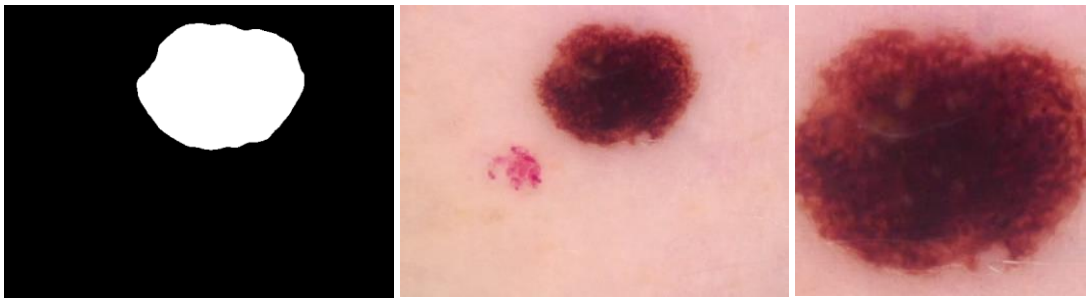


Figure 8: ISIC_0026506 (melanoma lesion) segmentation mask, original image, centre cropped image (from left to right)

4.2.1.2. Network training

The images and corresponding binary segmentation mask in the HAM10000 dataset [11] were used to train the model. The input images were cropped with resolution and passed as input to DenseNet121 without any rescaling. We used an Adam optimizer with a learning rate of 0.001 to train the model. The model was trained for 100 epochs or used early stopping monitoring the validation loss with patience 20 epochs. The corresponding training and validation, loss, and accuracy curve for the best model are shown in Fig 9.

Since the dataset size was small, to prevent overfitting we used ImageNet weights as starting weights and added additional bias and weight regularizers for feature extractors. We also used the dropout layer to regularize the classifier. In addition to that, to overcome the heavy class imbalance between melanoma and non-melanoma classes we tried many combinations of weights and loss functions like Focal loss, Binary cross-entropy, and categorical cross-entropy. Out of these combinations, Binary cross-entropy with a class weight of 6 for melanoma and 1 for non-melanoma with sigmoid activation in the output layer gave the best results.

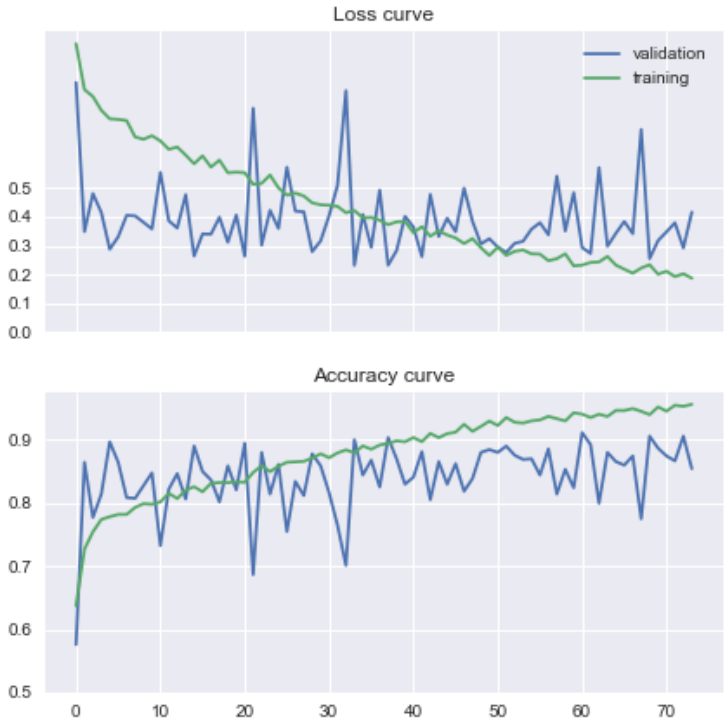


Figure 9: Accuracy and loss curves for centre base learner

4.2.1.3. Data augmentation

To improve the diversity of data, we have used several geometrical data augmentation techniques like horizontal flip, vertical flip, random rotation, and zoom.

4.2.2. Border Base Learner

4.2.2.1. Preprocessing step

Border cropping was performed on the images to extract a 25-pixel thick region along the borders of the lesions. The lesion border is one of the criteria (ABCDE) that dermatologists use to deduce whether a lesion is a melanoma since the border of a melanoma lesion is usually jagged and irregular [15]. Using the border region of the images allows the base model to focus on border details.

The first step was to obtain the lesion border mask based on the segmentation mask. This was done by first getting a 1-pixel thick lesion perimeter from the edges in the segmentation mask, then extending the perimeter by 5 pixels outwards and 20 pixels inwards. After which, the border mask is superimposed on the original image to obtain the border cropped image as shown below.

Figure 10: ISIC_0026506 (melanoma lesion) segmentation mask, border mask, original image, border cropped image (from left to right)

4.2.2.2. Network training

The Border Base Learner was trained based on the same model parameters as the Center Base Learner. The accuracy and loss curves from training this model are shown in Figure 11 below.

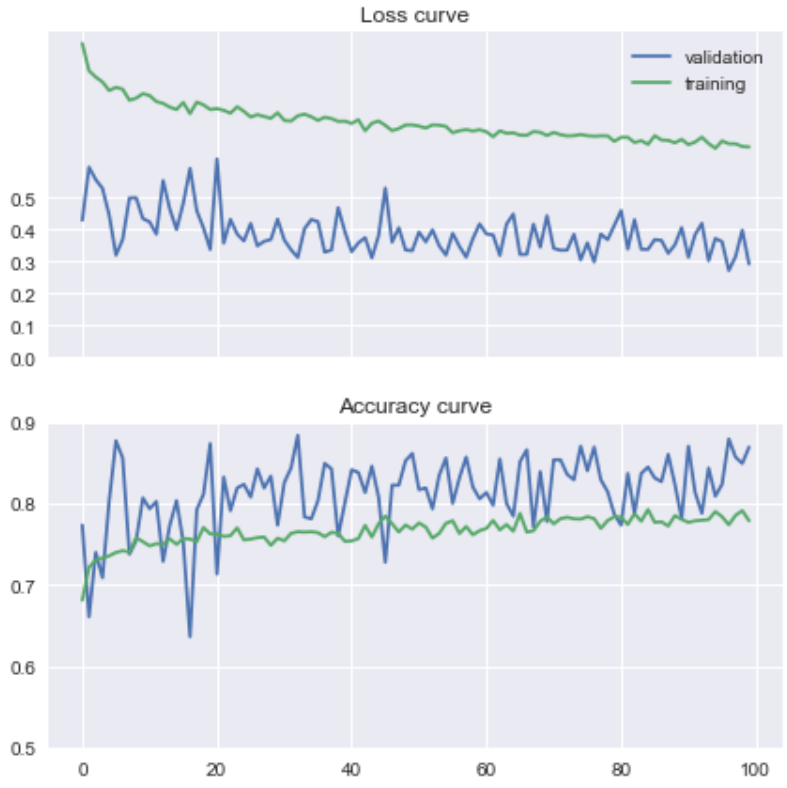


Figure 11: Accuracy and loss curves for border base learner

4.2.2.3. Data augmentation

To improve the diversity of data, we have used several geometrical data augmentation techniques like horizontal flip, vertical flip, random rotation, and zoom.

4.2.3. Asymmetry Base Learner

4.2.3.1. Preprocessing step

Asymmetry pre-processing was performed on the images to highlight the asymmetries in the lesion. Asymmetry is one of the criteria (ABCDE) that dermatologists use to deduce whether a lesion is considered melanoma. Hence, asymmetry pre-processing will also allow the base model to focus on asymmetry details of a lesion to predict melanoma.

The first step was to replace all non-lesion pixels in the image with the average colour of the lesion perimeter. This is to ensure that only the asymmetry of the lesion will be taken into consideration, and not colour variations of skin outside the lesion. This was done by first obtaining the 1-pixel thick lesion perimeter from the edges in the segmentation mask, then calculating the average RGB colours of the pixels along that perimeter in the original image. All non-lesion pixels are then replaced with that average colour (as shown in the third image below).

The second step was to get the asymmetry of the step-1 lesion image. Asymmetry is taken as the average of the 3 differences in pixel values of reflections of the lesion over the major axis, minor axis, and both major and minor axes. To do this, we first find out the y-axis and x-axis centroids and the orientation angle (of the major axis) of the lesion based on the segmentation mask. This was done using skimage regionprops. Next, we form linear equations (in the format y = m*x + c) of the major and minor axes by manipulating the centroids and orientation information. These equations of the major and minor axes are the basis for obtaining pixel differences of the reflections over axes.

For example, to get the asymmetry over the major axis for a particular pixel, we first get the x and y distances of that pixel to the major axis. Using the x and y distances, we locate the coordinate of the reflected pixel and get its colour difference. However, if the reflected coordinate falls out of the pixel range, then the difference is taken as 0. This is repeated for every pixel in the step-1 image. This process for obtaining pixel differences is similar for reflections over the major, minor, and both axes.

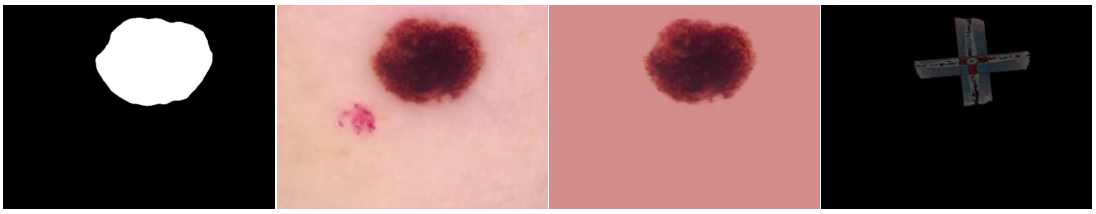


Figure 12: ISIC_0026506 (melanoma lesion) segmentation mask, original image, step-1 image, asymmetry processed image (from left to right)

4.2.3.2. Network training

The Asymmetry Base Learner was trained based on the same model parameters as the Center Base Learner. The accuracy and loss curves from training this model are shown in Figure 13 below.

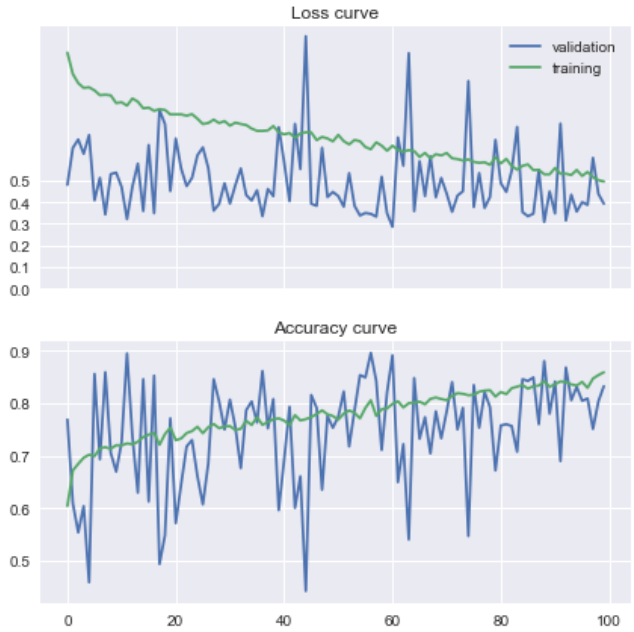


Figure 13: Accuracy and loss curves for asymmetry base learner

4.2.3.3. Data augmentation

To improve the diversity of data, we have used several geometrical data augmentation techniques like horizontal flip, vertical flip, random rotation, and zoom.

4.2.4. Whole Image Base Learner

4.2.4.1. Preprocessing step

No pre-processing was done on the images for Whole Image Base Learner. The full resolution images were only resized to 224-by-224 pixels by the ImageDataGenerator to be fed into the model.

4.2.4.2. Network training

The Whole Image Base Learner was trained based on the same model parameters as the Center Base Learner. The accuracy and loss curves from training this model are shown in Figure 14 below.

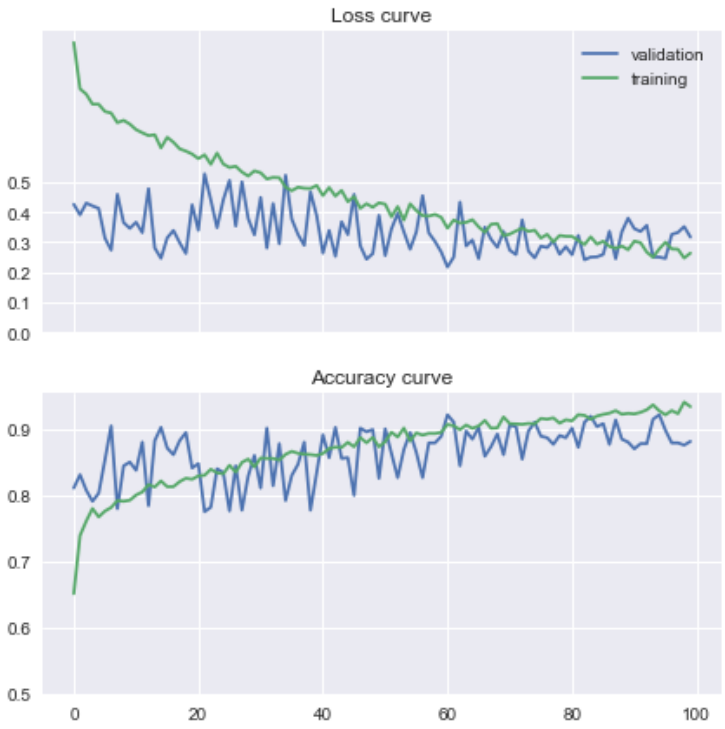


Figure 14: Accuracy and loss curves for original image base learner

4.2.4.3. Data augmentation

To improve the diversity of data, we have used several geometrical data augmentation techniques like horizontal flip, vertical flip, random rotation, and zoom.

4.2.5. Blue-White Veil Base Learner

This base learner was trained differently from the other 4 base learners. This was trained to identify the presence and absence of a specific diagnostic criterion known as a blue-white veil. This criterion is considered by dermatologists as one of the most useful in diagnosing melanoma: while only 51% of melanoma samples carry this property, its presence is indicative at 97% specificity [16].

4.2.5.1. Network training

The Blue-White Veil Base Learner was trained based on the same model parameters as the Center Base Learner, except the loss function. Instead of weighted binary cross entropy we used focal loss with sigmoid activation at output layer gave that gave best results. The accuracy and loss curves from training this model are shown in Figure 15 below.

Figure 15: Accuracy and loss curves for Blue-White-Veil base learner

4.2.5.2. Data augmentation

To improve the diversity of data, we have used several geometrical data augmentation techniques like horizontal flip, vertical flip, random rotation, and zoom.

4.3. Arbitrator

4.3.1. Final Arbitrator (Neural Network)

4.3.1.1. Network Architecture

The output of the previous dense layer of 5 base learners is concatenated together to form the input feature vector of size 640 (128 x 5). This feature vector is passed into a shallow neural network arbitrator that will perform the final classification (melanoma/non-melanoma). After several experimentations with many combinations of layers and corresponding nodes in each layer. An arbitrator with 5 hidden layers with (640, 256, 128,64,32) neurons gave optimal results.

4.3.1.2. Network Training

All the base learner weights were loaded and the output of the second last dense layer of each base learner was combined to create input to train the arbitrator network. Before starting the training all layers of the base learner networks were frozen, so the weights of base learners were not updated while training the arbitrator. The pre-processed images were passed as input to each base learner and the concatenated output of base learners was passed as input to the arbitrator.

We used an Adam optimizer with a learning rate of 0.0001 to train the model. The model was trained for 100 epochs or used early stopping monitoring the validation loss with patience 50 epochs. The corresponding training and validation, loss, and accuracy curve for the best model are shown in Figure 16 and Figure 17, respectively. To prevent overfitting, we used a dropout layer with a parameter value of 0.5 after each hidden layer. In addition to that, we experimented with different loss functions like Focal loss, weighted binary cross-entropy, and weighted categorical cross-entropy. Out of these, weighted binary cross-entropy with the class weight of 5 for melanoma class and 1 for non-melanoma class gave optimal results.

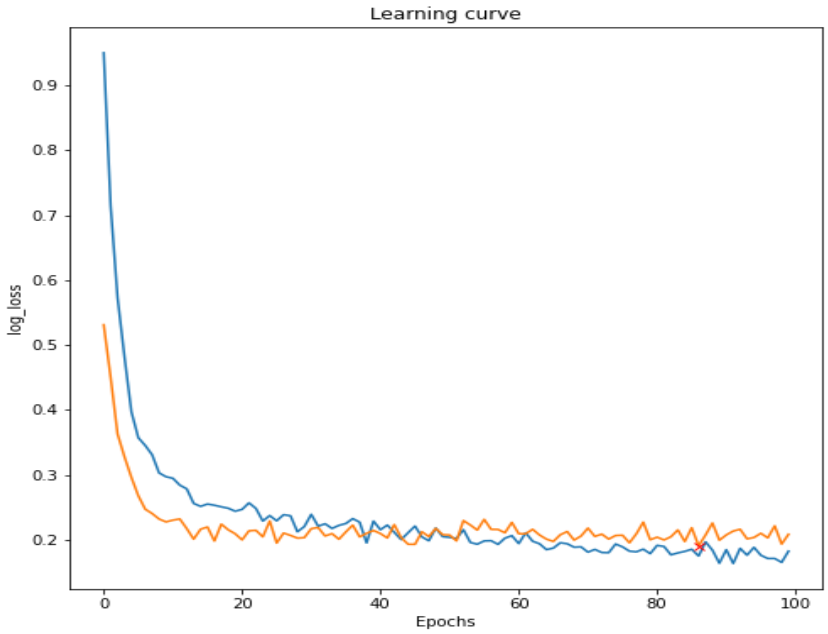


Figure 16: Loss curve for Final Arbitrator

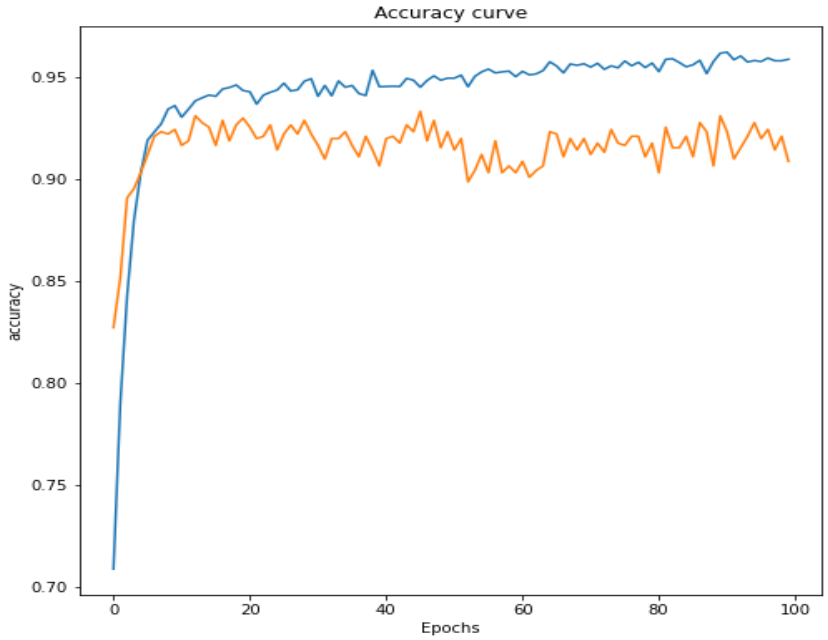


Figure 17: Accuracy curve for Final Arbitrator

4.3.2. Logistic Regression Arbitrator

We have also experimented with stacking 2 other different types of models as the meta-learner: Logistic Regression and Support Vector Machine. To train these meta-learners, the array of output probabilities from each base model (predicting melanoma or blue-white veil) based on the validation image dataset is fed into each classifier model, along with their ground truth labels. The 2 arbitrators are then tested on the test image dataset and their performance was evaluated using various metrics (shown in the results section).

The Logistic Regression arbitrator was set with class weights as 'balanced' to adjust for the class imbalance in its training data (which corresponds to validation image data in the train-val-test split). This helps to increase the recall of the melanoma class and reduce the false-negative rate in predictions. Even though the overall accuracy decreases by using these class weights, in the case of melanoma detection it is far more meaningful to have higher recall than to maximize the accuracy so that potential patients can seek early treatment.

4.3.3. Support Vector Machine (SVM) Arbitrator

The SVM arbitrator was also fitted with class weights as 'balanced' for the same reasons as stated above. It uses an 'rbf' kernel with C=100 and gamma='auto', which were the best parameters we found using Scikit-Learn's [6] GridSearchCV. The performance of the 3 arbitrators is compared in the result section.

5. RESULTS

To assess the performance of some individual components and the overall model, we used 10 % of the data from the HAM10000 [11] dataset as test data (the class proportion was maintained). In the test dataset, there are 111 dermoscopic images for melanoma class and 890 images for the non-melanoma class. The blue-white veil base learner was tested using 10% of the derm7pt dataset [12]. In this test dataset, there are 20 dermoscopic images for the Blue-white veil class and 81 images for the non-blue-white veil class.

5.1. Skin Lesion Segmentation

Before training a U-net model for image segmentation, we had tried image processing-based skin lesion segmentation, but it was unable to achieve good accuracy on the test data. We had tried multiple thresholding-based approaches to perform segmentation but were unable to produce a threshold that worked for all scenarios. So, we decided to move to a Deep learning-based approach. We trained a Lightweight U-net model to perform segmentation of skin lesions. We used two evaluation matrices, IoU (Intersection over Union) and Dice coefficient to evaluate the performance of the U-net model. Our U-net model achieved very good results in the test dataset. The results are illustrated in the table below.

Table 1: U-net Segmentation network performance in the test dataset

Model	IoU	Dice Coefficient
Unet Segmentation Network	90.03	96.03

5.2. Individual Base Learner Performance

The tables below show the performance of individual base learners in the test dataset. We were aiming to reduce the overall false-negative detection for melanoma because failing to detect melanoma for a person with melanoma will affect the early detection of melanoma and this could lead to the patient's death. Overall accuracy, precision, recall and f1 score are illustrated in Table 2. Table 3 illustrates the performance achieved by individual base learners for melanoma class.

Table 2: Overall accuracy, precision, recall, and f1 score achieved in test data

Model Name	Accuracy	Precision	Recall	F1 Score
Whole Base Learner	91	0.77	0.80	0.78
Center Base Learner	89	0.73	0.84	0.77
Border Base Learner	89	0.72	0.67	0.69
Asymmetry Base Learner	85	0.60	0.57	0.58

Table 3: Precision, recall, and f1 score for melanoma class achieved in test data

Model Name	Precision	Recall	F1 score
Whole Base Learner	0.57	0.67	0.62
Center Base Learner	0.49	0.79	0.60
Border Base Learner	0.51	0.39	0.44
Asymmetry Base Learner	0.29	0.22	0.25

The Blue-white veil base learner's performance was tested on the derm7pt dataset [12]. The results are provided in the table below.

Table 4: Overall precision, recall, and f1 score for blue-white veil base learner

Model Name	Accuracy	Precision	Recall	F1 Score
Blue-White Base Learner	90	0.85	0.82	0.83

5.3. Arbitrator performance

The best performing neural network arbitrator was trained on the last feature layer output of the base models, while the best performing Logistic Regression and SVM arbitrators were trained using the last layer outputs of the base models.

The table below shows a comparison of some relevant metrics of the 3 arbitrators. The detailed classification report for each arbitrator has also been included in the appendix section. The neural network was tested at both 0.5 and 0.67 threshold values to get a clearer comparison with other arbitrators.

Table 5: Accuracy, Precision (Melanoma), Recall (Melanoma), Precision (Overall), Recall (Overall) for arbitrators

Arbitrator	Accuracy (%)	Precision (Melanoma)	Recall (melanoma)	Precision (overall)	Recall (Overall)
Neural Network (threshold=0.5)	91.92	0.6027	0.7928	0.7879	0.8638
Neural Network (threshold=0.67)	89.12	0.5053	0.8559	0.7428	0.8757
Logistic Regression	89.22	0.5084	0.8198	0.7420	0.8605
SVM	87.92	0.4750	0.8559	0.7225	0.8690

A comparison of the ROC curves of the 3 arbitrators is shown below. The neural network arbitrator was the best performing out of the 3 arbitrators with a ROC area of 0.9472. The Logistic Regression arbitrator came second with a ROC area of 0.9421 and lastly, the SVM arbitrator with a ROC area of 0.9309.

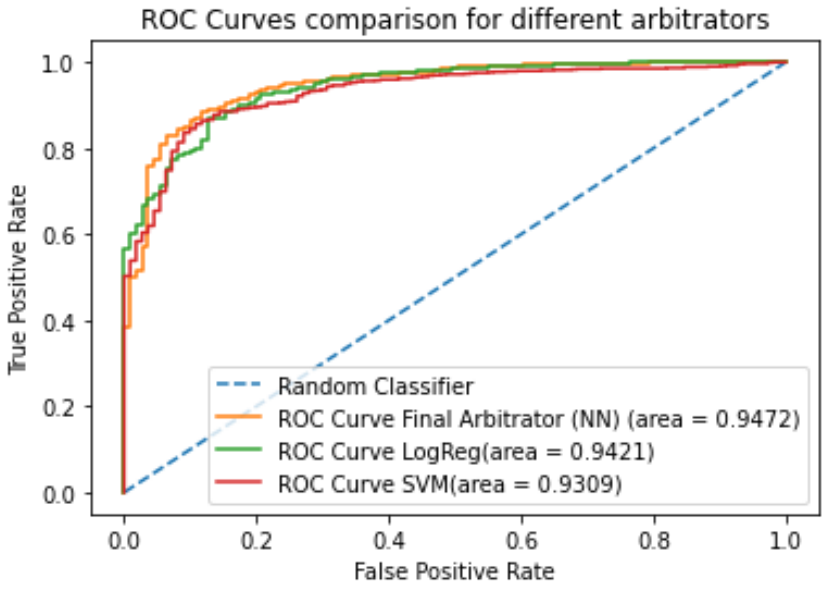


Figure 18: ROC Curve comparison across different arbitrators

A comparison of the Precision-Recall (PR) curves of the 3 arbitrators is shown below. The neural network arbitrator was the best performing out of the 3 arbitrators with a PR area of 0.99244. The Logistic Regression arbitrator came second with a PR area of 0.99208 and lastly, the SVM arbitrator with a PR area of 0.99060.

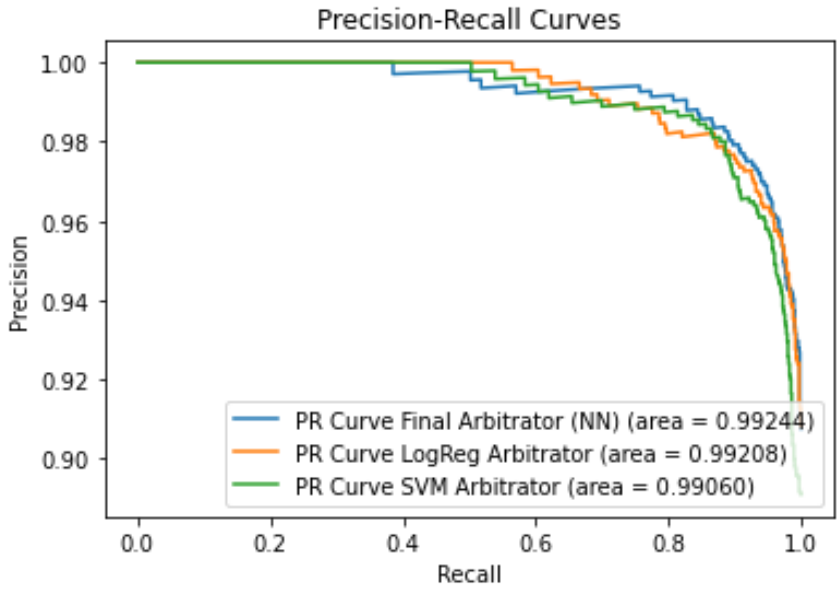


Figure 19: Precision-Recall curves of arbitrators

Overall, it is observed that the neural network arbitrator performed the best out of all 3 arbitrators. Hence, it is used in the final inference model.

6. USER INTERFACE

6.1. UI Description

6.1.1. Features

We have used Streamlit [10] framework to build our web application. It performs the following things on the Python backend:

a. Inference – When the user uploads an image of a skin lesion under the "Skin Lesion Image" section on the sidebar (left side), the system shows whether the cancer is melanoma/non-melanoma.

   The image is input to the saved model for inference. The model predicts whether the cancer is melanoma/non-melanoma.

b. Report Generation – When the user clicks the “Download & Email Report” button on the main page, a detailed report for the diagnosis is generated by the system.
   We have stored a template on the backend. All the end-user information along with the network prediction probability bar plot gets automatically populated on the template. We have used PyFPDF [17] for this task. It is an easy-to-use library for PDF document generation.
c. Email ID verification – When the user clicks the “Download & Email Report” button on the main page, the email ids given by the patient and the physician on the sidebar are verified by the system.
   We have used Email Verifier API [18] provided by Hunter. The API requires an email address and returns detailed verification results.
d. Send Email – When the user clicks the “Download & Email Report” button on the main page, the system sends emails to the verified email ids. The email gets sent by our team’s Gmail account. It has personalized body content and generated Report attached. We have used the “SMTP lib” [19] library for this task. It creates a Simple Mail Transfer Protocol client session object which is used to send emails to any valid email id on the internet.

6.1.2. User Guide

1. Go to the Web Application.
2. Complete the following things on the Sidebar (left side).
   i. Upload image of skin lesion under “Skin Lesion Image” section.
   ii. Fill in Patient details under the "Patient Information" section.
   iii. Fill Dermatologist details under the "Physician Information" section.
3. Click on the “Download & Email Report” button on the main page.

6.1.3. To run on Local Machine (Windows)

1. Navigate to the following directory:
   Melanoma-Detection-System/SystemCode/Front-end/

2. Download models from the link given in ***Models/link.txt*** and move both models to ***Front-end*** folder.
3. Open ***Front-end/app.py*** and give your API_KEY (line 196), EMAIL_ADDRESS (line 236), and EMAIL_PASSWORD (line 237).
4. Allow less secure apps to access your google account.
5. Open Anaconda command prompt.
6. Create a new anaconda environment:
   conda create -n project python==3.7
7. Activate anaconda environment:
   conda activate project
8. Navigate to the following directory in anaconda command prompt:
   cd …/Melanoma-Detection-System/SystemCode/Front-end/
9. Install the required dependencies:
   pip install -r requirements.txt
10. Run the app:
    streamlit run app.py

6.2. UI Mockup

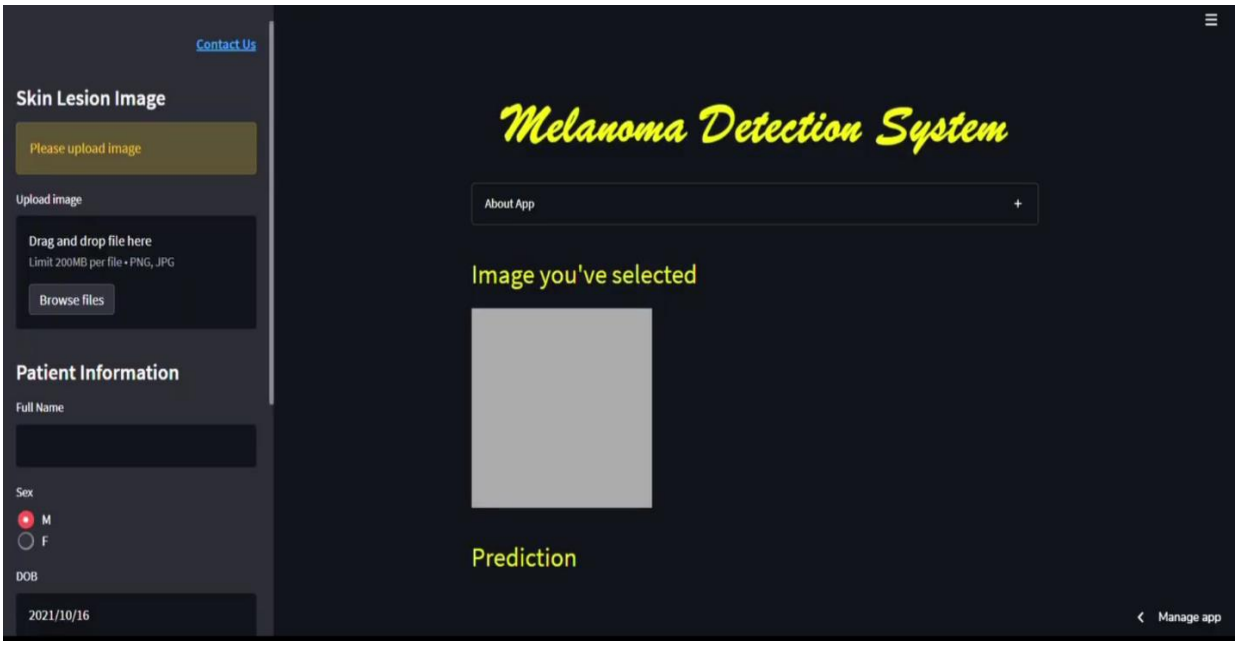


Figure 20: Landing page of UI

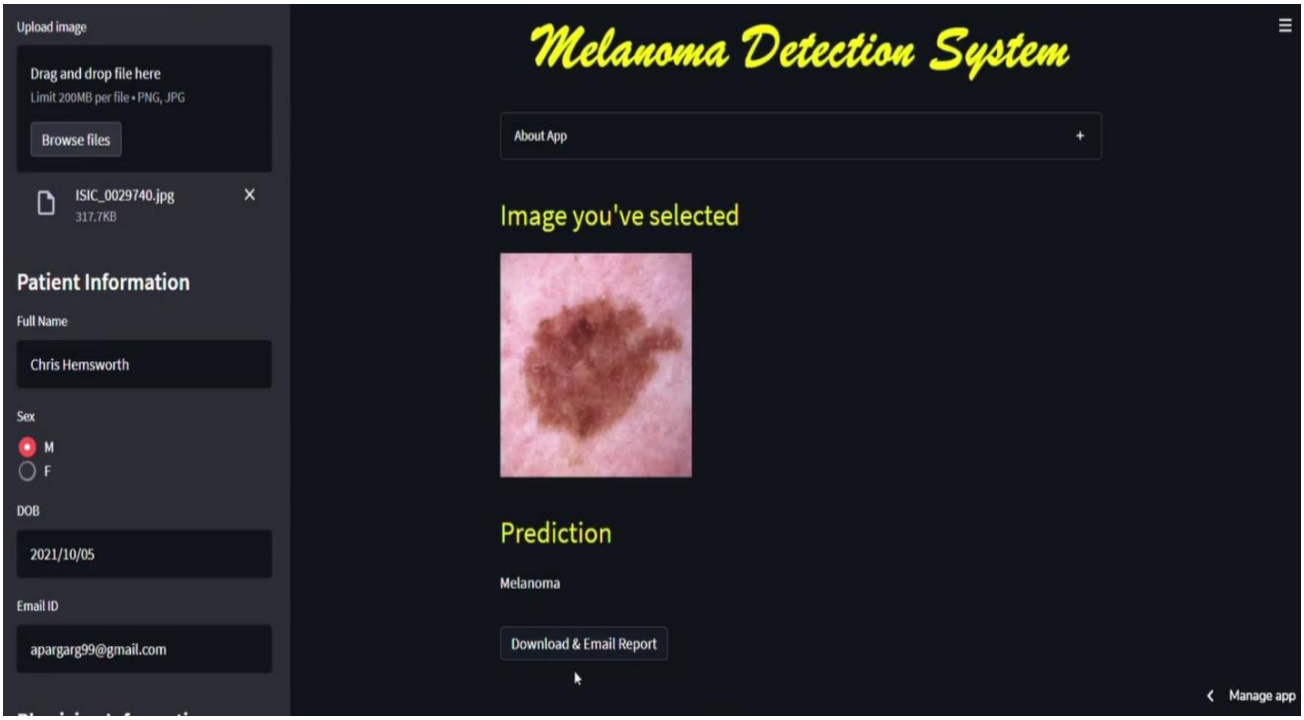


Figure 21: UI showing results of the prediction

7. CONCLUSION AND FUTURE WORK

This section summarizes what has been done in this project and some of the future works that may enhance the solution.

7.1. Conclusion

In this project, we have approached the problem of image-based melanoma classification using a stacked ensemble model. The proposed model consists of 5 base learners, each trained on different features of the lesion (whole image, centre, border, asymmetry, blue-white veil). The neural network arbitrator concatenates the last feature layer of all the 5 base learners as input and outputs a single value predicting the likelihood of melanoma. We have also implemented a U-Net segmentation network to perform image segmentation as the image pre-processing steps required for the centre, border, and asymmetry base learners require lesion segmentation masks along with the original images.

7.2. Future Work

One way to further improve the performance of the system in the future could be by using patient demographic information along with the images, this extra feature may improve the overall performance of our model. Currently, we have experimented with the available demographic features in HAM10000 dataset, since only some images had relevant demographic information we were not able to achieve a conclusive improvement in performance. The results and observations are given in the appendix session of this report. In addition to that collecting more melanoma skin lesion images will reduce the class imbalance problem that we faced, and this will, in turn, improve the performance of melanoma skin lesion classification.

One future expansion that we are planning to include in our pipeline is melanoma stage prediction, this can be achieved by using the thickness information of skin lesions. Currently, none of the public datasets have this information.

A faster UI would also help to improve the user experience of our system.

REFERENCES

[1] American Society of Clinical Oncology (ASCO), "Melanoma: Statistics." https://www.cancer.net/cancer-types/melanoma/statistics (accessed Sep. 01, 2021).

[2] H. Skvara, L. Teban, M. Fiebiger, M. Binder, and H. Kittler, "Limitations of Dermoscopy in the Recognition of Melanoma," *Archives of Dermatology*, vol. 141, no. 2, pp. 155–160, Feb. 2005, doi: 10.1001/archderm.141.2.155.

[3] "Frequently Asked Questions about a dermatology skin or mole biopsy by the team at Mosaic Dermatology Houston." https://www.mcdermatology.com/biopsy.html (accessed Sep. 01, 2021).

[4] "TensorFlow." https://www.tensorflow.org/ (accessed Oct. 01, 2021).

[5] "Keras." https://keras.io/about/ (accessed Oct. 01, 2021).

[6] "Scikit-learn." https://scikit-learn.org/stable/ (accessed Oct. 01, 2021).

[7] M. Naveenkumar and A. Vadivel, " OpenCV for Computer Vision Applications," *proceedings of National Conference on Big Data and Cloud Computing (NCBDC)* , 2005.

[8] S. van der Walt *et al.*, "scikit-image: image processing in Python," *PeerJ*, vol. 2, Jun. 2014, doi: 10.7717/peerj.453.

[9] P. Barrett, J. Hunter, J. T. Miller, J.-C. Hsu, and P. Greenfield, "matplotlib -- A Portable Python Plotting Package," *Astronomical Data Analysis Software and Systems XIV ASP Conference Series*, vol. Vol. 347, no. Proceedings of the Conference held 24-27 October, 2004 in Pasadena, California, USA. Edited by P. Shopbell, M. Britton, and R. Ebert. San Francisco: Astronomical Society of the Pacific, pp. 91–91, 2005.

[10] "Streamlit." https://streamlit.io/ (accessed Oct. 01, 2021).

[11] P. Tschandl, C. Rosendahl, and H. Kittler, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific Data*, vol. 5, no. 1, Dec. 2018, doi: 10.1038/sdata.2018.161.

[12] J. Kawahara, S. Daneshvar, G. Argenziano, and G. Hamarneh, "Seven-Point Checklist and Skin Lesion Classification Using Multitask Multimodal Neural Nets," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 2, Mar. 2019, doi: 10.1109/JBHI.2018.2824327.

[13] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015.

[14] T. Mendonca, P. M. Ferreira, J. S. Marques, A. R. S. Marcal, and J. Rozeira, "PH2 - A dermoscopic image database for research and benchmarking," Jul. 2013. doi: 10.1109/EMBC.2013.6610779.

[15] The Skin Cancer Foundation, “Melanoma Warning Signs.” https://www.skincancer.org/skin-cancer-information/melanoma/melanoma-warning-signs-and-images/ (accessed Oct. 01, 2021).

[16] S. W. Menzies, “Frequency and Morphologic Characteristics of Invasive Melanomas Lacking Specific Surface Microscopic Features,” *Archives of Dermatology*, vol. 132, no. 10, Oct. 1996, doi: 10.1001/archderm.1996.03890340038007.

[17] “fpdf PyPI.” https://pypi.org/project/fpdf/ (accessed Oct. 01, 2021).

[18] “Email Verifier API.” https://hunter.io/api/email-verifier (accessed Oct. 01, 2021).

[19] “smtplib — SMTP protocol client.” https://docs.python.org/3/library/smtplib.html (accessed Oct. 01, 2021).

[20] “ISIC Archive.” https://www.isic-archive.com/#!/topWithHeader/wideContentTop/main (accessed Oct. 01, 2021).

[21] A. Pacheco *et al.*, “PAD-UFES-20: a skin lesion dataset composed of patient data and clinical images collected from smartphones,” *Mendeley Data*, vol. V1, 2020, doi: 10.17632/zr7vgbcyr2.1.

[22] “pandas-profiling PyPI.” https://pypi.org/project/pandas-profiling/ (accessed Oct. 01, 2021).

[23] “autoviz PyPI.” https://pypi.org/project/autoviz/ (accessed Oct. 01, 2021).

[24] “evalml PyPI.” https://pypi.org/project/evalml/ (accessed Oct. 01, 2021).

[25] “autoviml PyPI.” https://pypi.org/project/autoviml/ (accessed Oct. 01, 2021).

APPENDIX A: CLASSIFICATION REPORTS FOR NEURAL NETWORK, LOGISTIC REGRESSION, SVM ARBITRATORS

Neural Network Arbitrator (threshold=0.5)

```
Best accuracy (on testing dataset): 91.92%
Balanced Accuracy:86.38%
              precision    recall  f1-score   support

    melanoma     0.6027    0.7928    0.6848       111
      non_me     0.9731    0.9349    0.9536       891

    accuracy                         0.9192      1002
   macro avg     0.7879    0.8638    0.8192      1002
weighted avg     0.9321    0.9192    0.9239      1002
```

Neural Network Arbitrator (threshold=0.67)

```
Best accuracy (on testing dataset): 89.12%
Balanced Accuracy:87.57%
              precision    recall  f1-score   support

    melanoma     0.5053    0.8559    0.6355       111
      non_me     0.9803    0.8956    0.9361       891

    accuracy                         0.8912      1002
   macro avg     0.7428    0.8757    0.7858      1002
weighted avg     0.9277    0.8912    0.9028      1002
```

Logistic Regression (balanced class weights)

```
Accuracy:89.22%
Balanced Accuracy:86.05%
Report:              precision    recall  f1-score   support

    melanoma     0.5084    0.8198    0.6276       111
      non_me     0.9757    0.9012    0.9370       891

    accuracy                         0.8922      1002
   macro avg     0.7420    0.8605    0.7823      1002
weighted avg     0.9239    0.8922    0.9027      1002
```

SVM (balanced class weights)

```
Accuracy:87.92%
Balanced Accuracy::86.90%
Report:              precision    recall  f1-score   support

    melanoma     0.4750    0.8559    0.6109       111
      non_me     0.9800    0.8822    0.9285       891

    accuracy                         0.8792      1002
   macro avg     0.7275    0.8690    0.7697      1002
weighted avg     0.9241    0.8792    0.8933      1002
```

APPENDIX B: MELANOMA CLASSIFICATION MODEL TRAINED ON PATIENT-LEVEL CONTEXTUAL INFORMATION

Furthermore, to check if there is a strong correlation between patient-level contextual information and melanoma classes, we decided to develop a melanoma classification model trained on patient-level contextual information. ISIC Archive and PAD-UFES-20 datasets were used for this analysis.

ISIC Archive

EDA

The dataset consists of 69,445 images from different patients. There are 29 columns in the metadata among which 7 are completely null. We used Automated EDA libraries – Pandas Profiling and AutoViz; to further perform a detailed analysis of the remaining 22 columns. To view the analysis got to –

Feature Selection

Out of those 22 columns only 4 provided patient-level contextual information Family History, Patient History, Patient Sex, Patient Age. Hence these were selected for training the model. Now the new data has the following 5 columns: Independent = Family History, Variable Patient History, Patient Sex, Patient Age Dependent Variable = Melanoma (nevus, melanoma, etc.).

Preprocessing

The following steps were performed to preprocess the new data:

1. Drop rows will null value(s) in any column.
2. Modify Independent variable column - if the value is melanoma, then replace with true, else with false.
3. Drop duplicate rows.

```
df.head()
```

	Family_History	Patient_History	Patient_Sex	Patient_Age	Melanoma
0	False	False	female	70.0	True
1	False	True	female	40.0	False
2	False	False	male	45.0	False
3	False	False	female	50.0	False
4	False	True	female	30.0	False

```
df.shape
```

```
(189, 5)
```

```
df['Melanoma'].value_counts()
```

```
False    114
True      75
Name: Melanoma, dtype: int64
```

Model Development

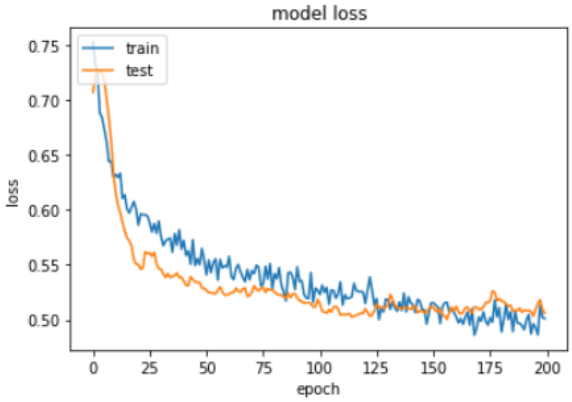
We used Automated ML libraries – EvalML and AutoViML; for training and testing the ML models for melanoma classification.
The libraries trained the following models:

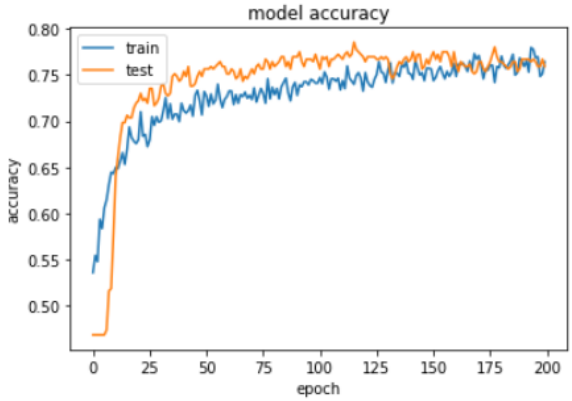
- Elastic Net Classifier.
- Decision Tree Classifier.
- Random Forest Classifier.
- LightGBM Classifier.
- Logistic Regression Classifier.
- XGBoost Classifier.
- Extra Trees Classifier.
- CatBoost Classifier.
- Naïve Bayes.
- Adaboost.
- Bagging Classifier.

The best model result is as follows:

```
Cross Validation
----------------
             Balanced Accuracy Binary  Log Loss Binary    F1   AUC # Training # Validation
0                                0.611            0.650 0.571 0.611         59           30
1                                0.583            0.692 0.500 0.569         59           30
2                                0.523            0.717 0.462 0.525         60           29
mean                             0.572            0.686 0.511 0.569          -            -
std                              0.045            0.034 0.056 0.043          -            -
coef of var                      0.079            0.049 0.109 0.076          -            -
```

From the results, we observe that model is not performing much better than a random classifier. But the results are inconclusive because of limited data available.
A similar analysis was done with the PAD-UFES-20 dataset. The results are inconclusive for that as well because of limited data available.
We also tried to increase the arbitrator's performance by considering the 4 patient-level contextual features Family History, Patient History, Patient Sex, Patient Age. The output of the previous dense layer of the arbitrator is concatenated with the 4 patient-level contextual features to produce a new input for a new arbitrator (having a similar architecture as the original arbitrator). But since only a few images in the ISIC Archive have corresponding patient-level contextual features, the model could only achieve 77% accuracy.





Accuracy : 77.0

Report :

	precision	recall	f1-score	support
0	0.75	0.82	0.78	50
1	0.80	0.72	0.76	50
accuracy			0.77	100
macro avg	0.77	0.77	0.77	100
weighted avg	0.77	0.77	0.77	100

Predict	0	1
Actual		
0	41	9
1	14	36

Cross Validation

	Balanced Accuracy	Binary Log Loss	Binary F1	AUC	# Training	# Validation
0	0.712	0.568	0.509	0.744	771	386
1	0.694	0.570	0.498	0.758	771	386
2	0.776	0.567	0.596	0.831	772	385
mean	0.727	0.568	0.534	0.778	-	-
std	0.043	0.002	0.054	0.047	-	-
coef of var	0.059	0.003	0.101	0.060	-	-