# ****Melanoma skin cancer detection with CNN****

Classifying mole images with Efficientnet

Melanoma of the skin, one of the most commonly occurring forms of cancer, represents a serious risk to women’s and men’s health worldwide. The survival rates for melanoma skin cancer depend heavily on the cancer’s stage when diagnosed. If the melanoma of the skin is detected before it has the chance to spread, the chances of surviving are very high. If the cancer has already spread to other body parts, it is less likely that the treatment will be successful, and the risk of death is thus higher. Getting skin exams on a regular basis can help diagnose skin cancer early and reduce risks. According to the Canadian Cancer Society, doctors follow the ABCDE rule to differentiate a normal mole from skin cancer. This rule includes features such as the asymmetry, the irregularity of the edge, the colour, the size and the evolvement of a mole.

In this article, we are going to predict the diagnosis of suspicious moles based on images of benign and malignant skin moles by using a Convolution Neural Network with keras tensorflow.

The dataset used contains 1800 pictures (224x244) of benign skin moles and 1494 pictures (224x244) of malignant skin moles from the ISIC (International Skin Image Collaboration) Archive.

# ****Implementation****

You can find the full codebase on Google Colab with this link: [https://colab.research.google.com/drive/1jSpUEJIAz2N6A0rY\_rGQYtLXNgSUP8sK](https://colab.research.google.com/drive/1jSpUEJIAz2N6A0rY_rGQYtLXNgSUP8sK?usp=sharing)

****Loading pictures****

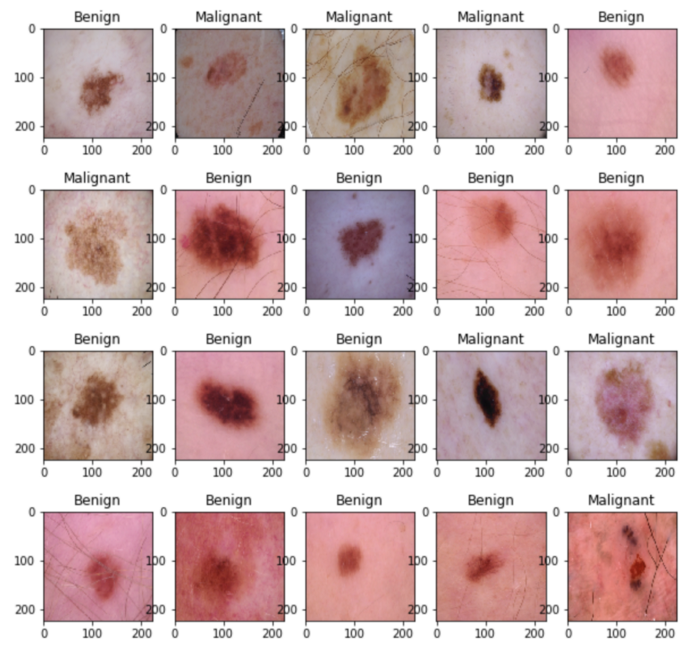
Pictures are loaded and turned into numpy arrays using their RGB values.

****Creating train, validation and test sets****

Pictures are labeled as 0 (benign) and 1 (malignant). Dictionary of images with their corresponding labels are created. Train and test data are shuffled. Validation set is created with 500 pictures from the original train set. The final train, validation and test sets contain respectively 2137, 500 and 660 images.

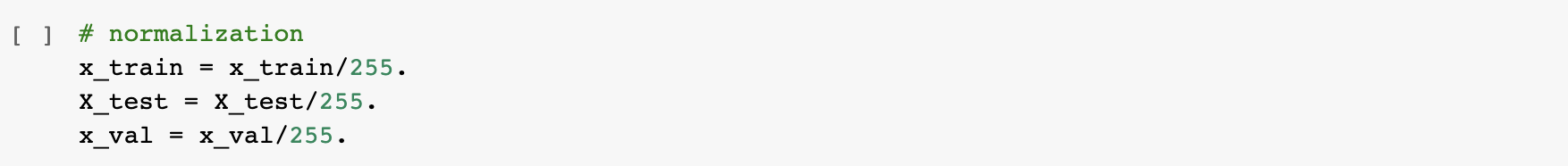
****Pictures example****

Here is a display of the first 20 images of moles belonging to the train set and their respective labels.



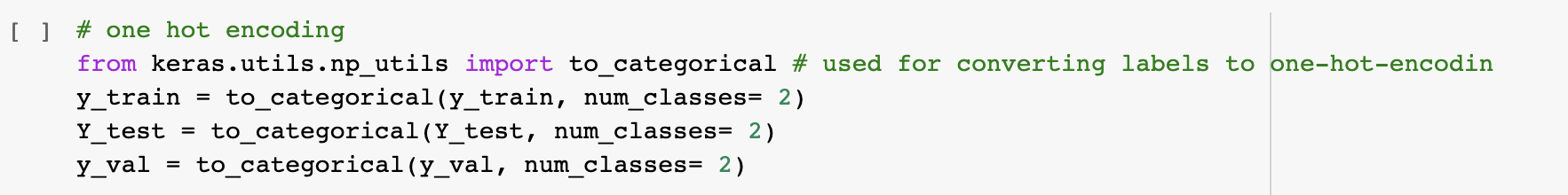
****Normalizing pictures****

Pictures’ RGB values are divided by 255 to have values between 0 and 1.



****One-hot encoding****

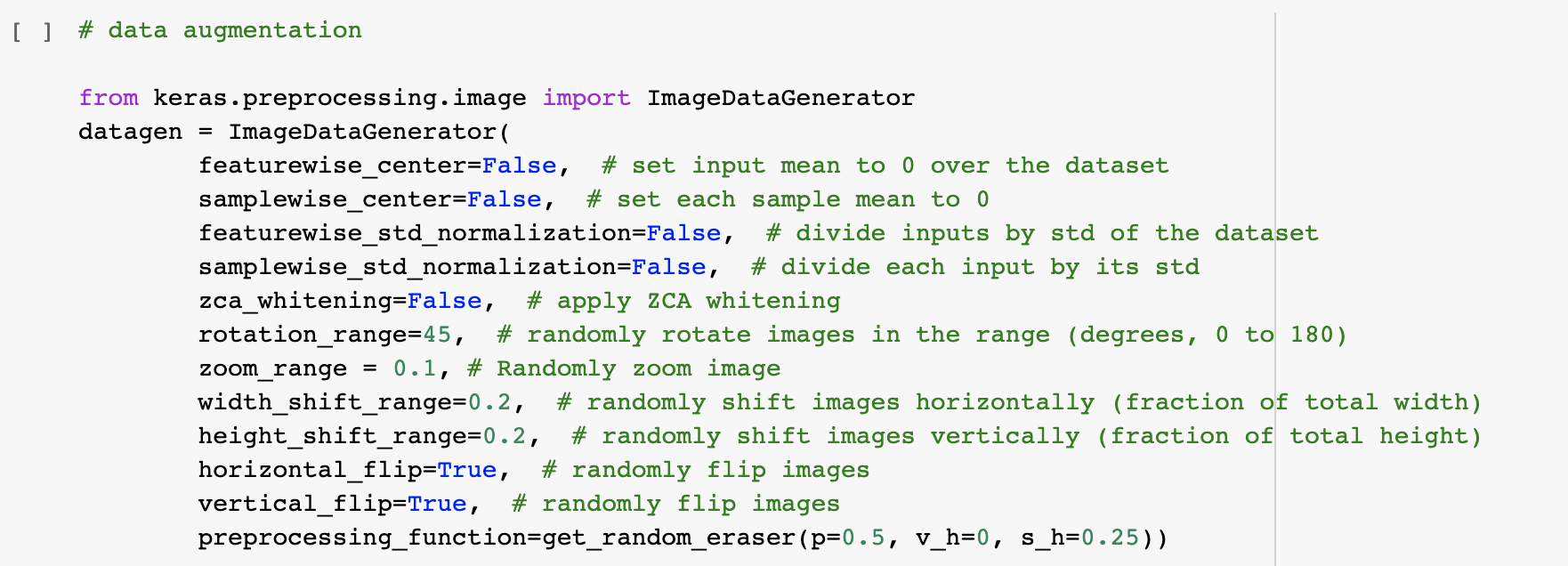
Categorical variables are transformed into vectors of 0 & 1 where vectors length correspond to the number of categories to classify.



****Data augmentation****

Additional augmented data is created by reasonably modifying the data in our training set. The purpose is to add more [data](http://becominghuman.ai/) to our training set that is similar to the data we already have but is reasonably modified to some degrees so that it is not exactly the same.

Square regions of pictures are randomly masked out (cut out) using the Random Erasing implementation to improve overall performance of our convolutional neural network. Image Data Generator is used to generate batches of image data with real-time data augmentation.

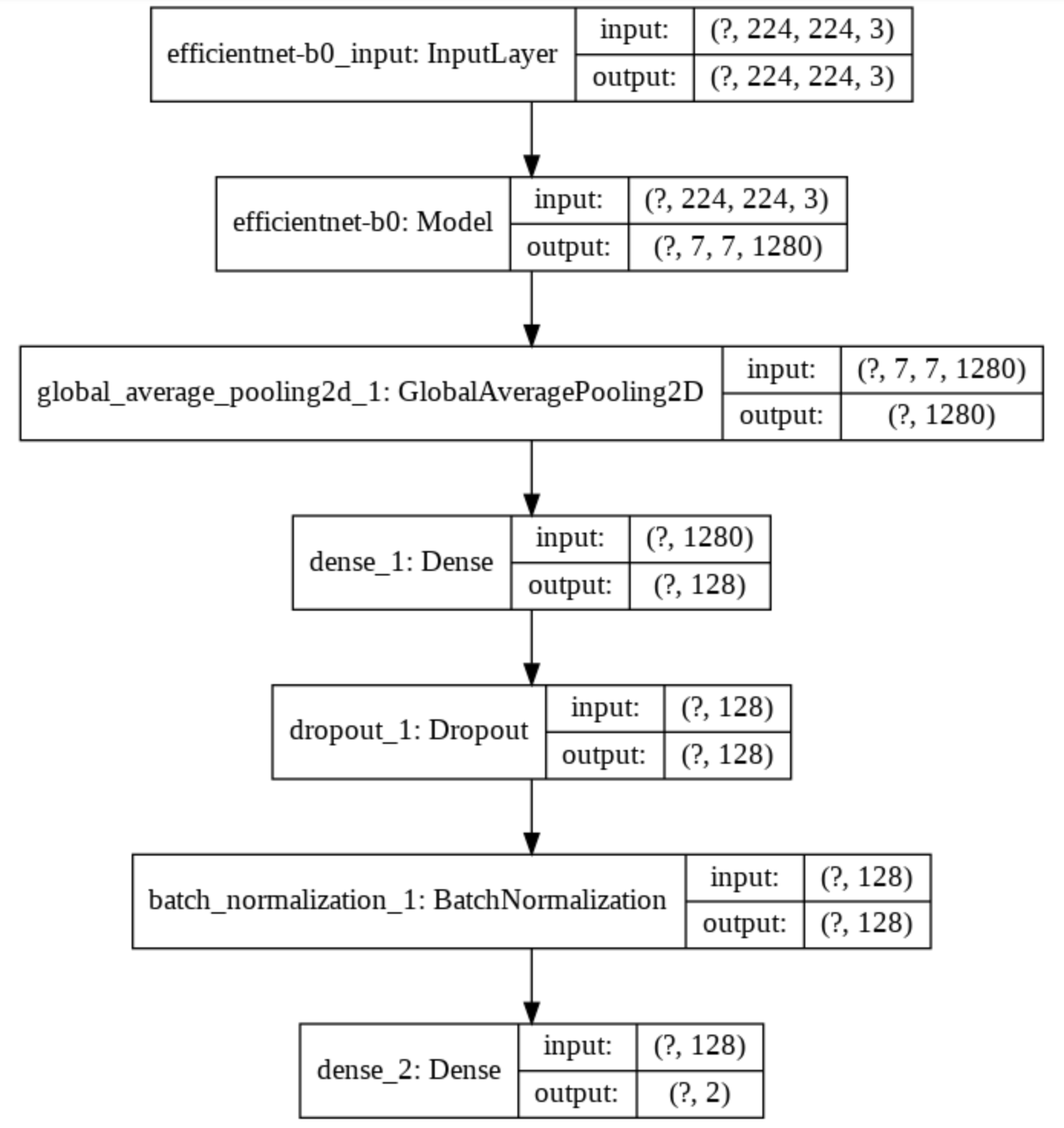


****Building CNN model****

The model uses the pretrained model Efficientnet, a new [CNN](http://becominghuman.ai/) model introduced by Google in May 2019. In the paper called “[EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks](https://arxiv.org/pdf/1905.11946.pdf)”, EfficientNet showed a great improvement in accuracy and in computational efficiency on ImageNet compared to other state of the art CNNs.



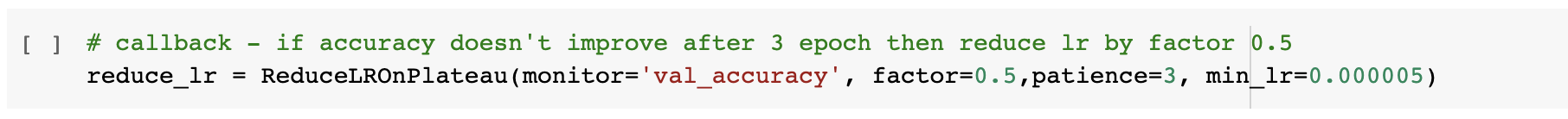
A weight decay of 0.001 is set to reduce overfitting of the CNN and improve its performance on new data. A dropout of 20% is set to randomly ignore some subsets of nodes by dropping some nodes in a given layer during training. This helps the model generalize better on data it has not seen before.



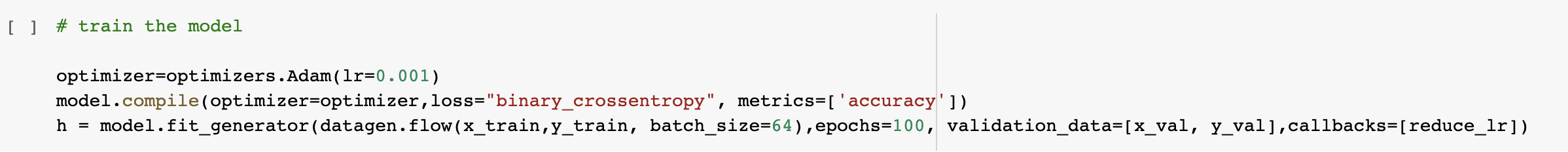
****Training the model****

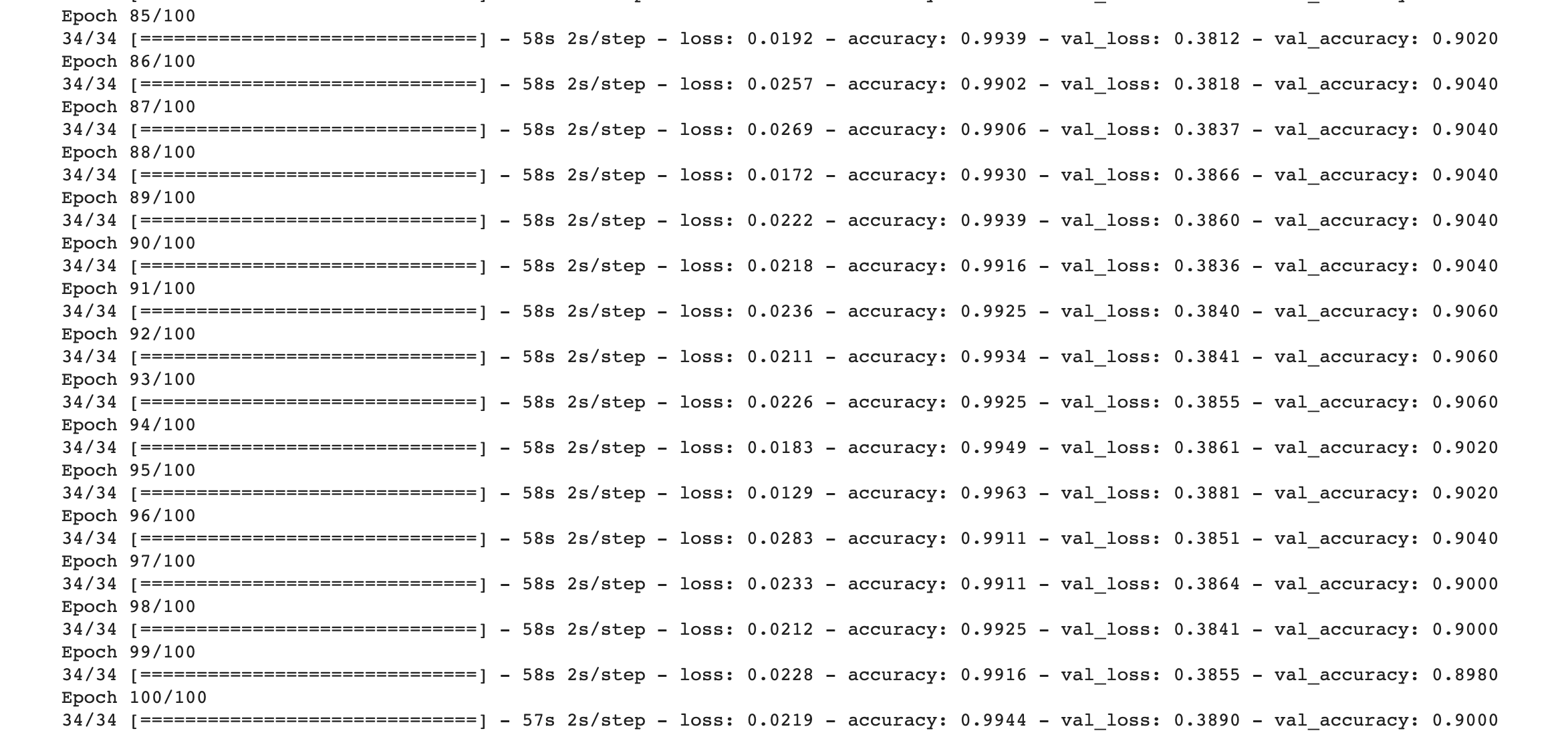
Our model is trained on the train set and uses a validation set to ensure that it is not overfitting and can generalize on data it has not seen before.

The callback ReduceLROnPlateau is used to reduce learning rate when a metric has stopped improving. In this part we set a factor of 0.5, a patience of 3 and a minimum learning rate of 0.000005 so if the validation accuracy doesn’t improve after 3 epoch, the learning rate will be reduced by factor 0.5

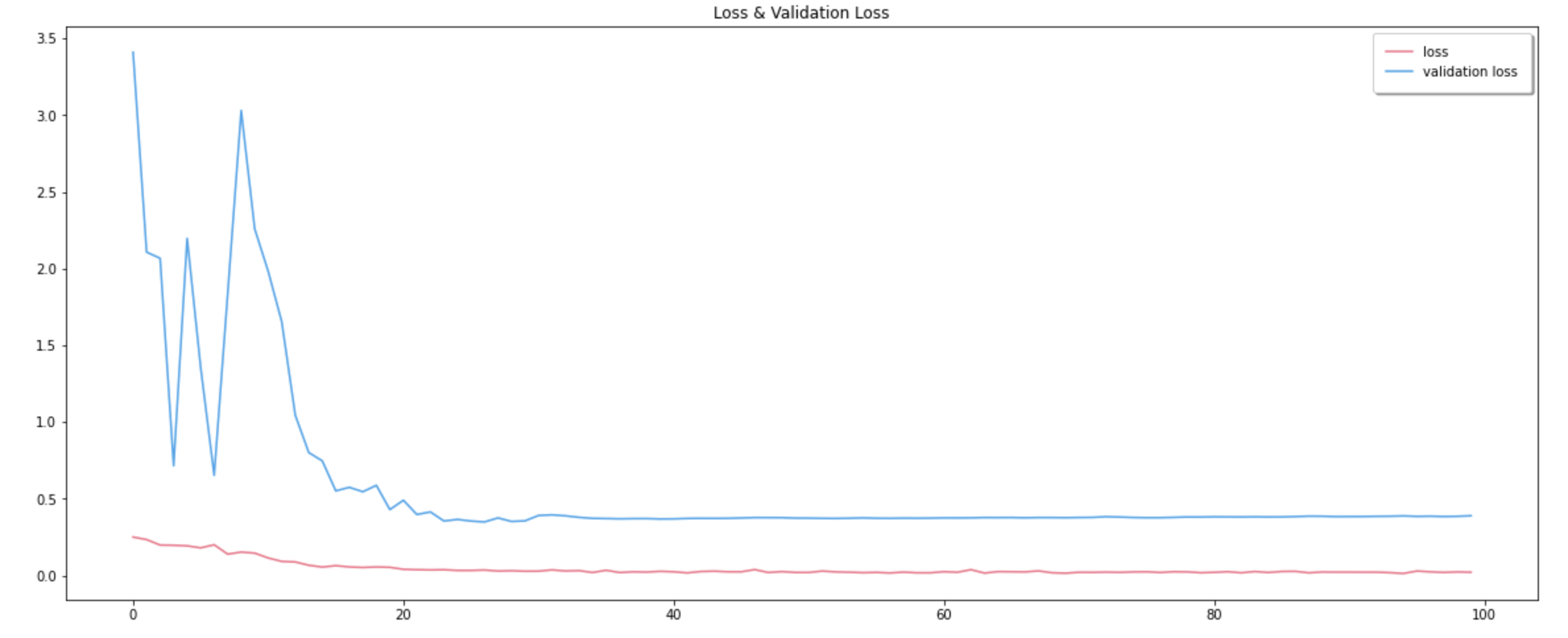


The model is trained on 100 epochs with a batch size of 64 which corresponds to the number of observations propagated through the network.

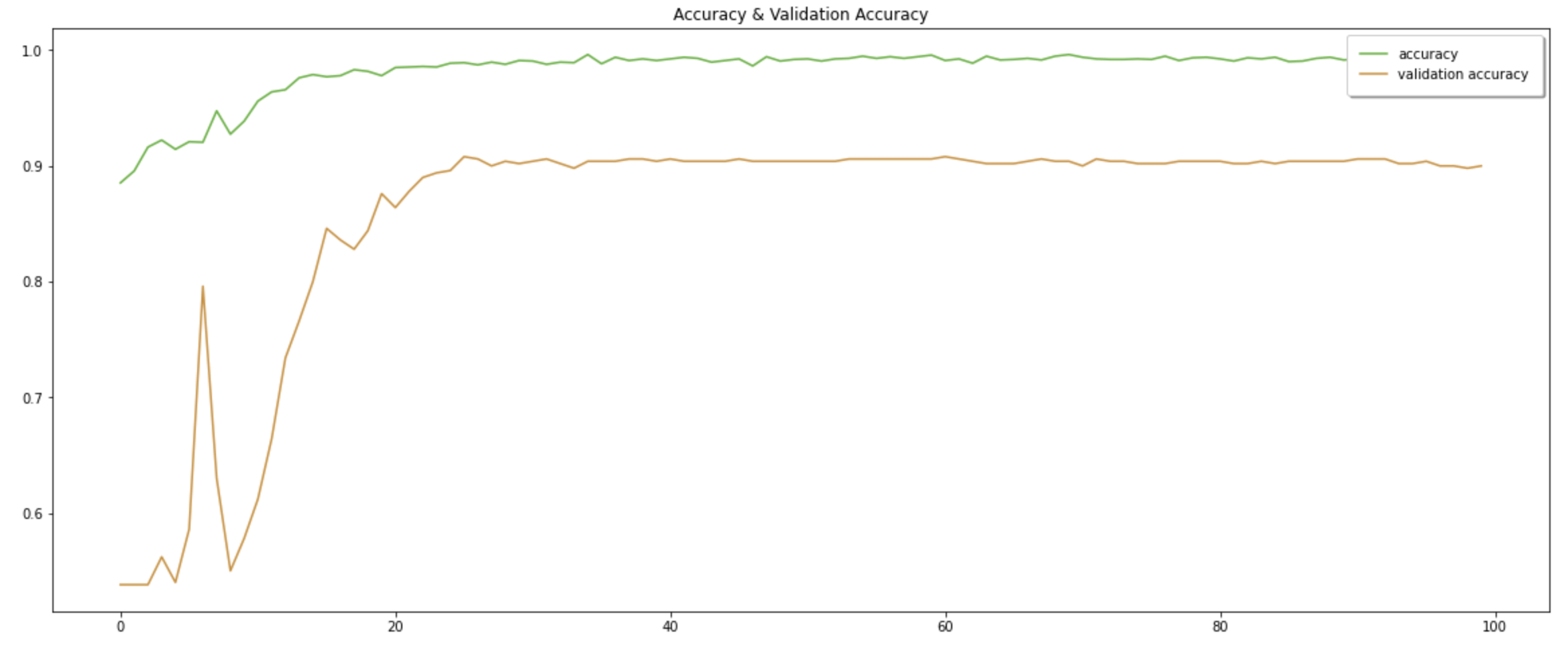




Loss & Validation Loss:



Accuracy & Validation Accuracy:



Overall, validation metrics are getting closer to the training metrics after the 20th epoch which is a good sign indicating that our model is not overfitting.

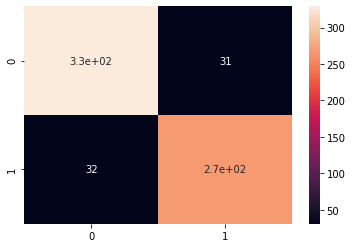
****Testing the model****

The test set is structured just like the training set and is used to assess the performance of our model. We want to make sure the model is able to generalize well. This is definitely the case, the accuracy on the test set is 90.45% which even higher than the validation accuracy.

****Confusion Matrix****

A confusion matrix showing a table with 4 different combinations of predicted and actual values is used to evaluate the performance of our model.

* Number of benign moles classified as benign: 329
* Number of benign moles classified as malignant: 32
* Number of malignant moles classified as malignant: 268
* Number of malignant moles classified as benign: 31



# ****Conclusion****

In this task, we focused on the classification’s accuracy which is 90.45% based on the test data. This does not give the full picture however. In the context of a patient waiting for melanoma test results, the worst situation is to wrongly diagnose a malignant mole as benign. The patient would not be starting any treatments, and this would decrease the chances of survival. This means it is more important for us to know how many malignant cases are being caught rather than just how good the model is at guessing between the two options. By calculating the recall rate (Number of malignant moles classified as malignant/Total number of malignant moles), we can see that the model correctly identifies pictures of malignant moles 89.63% of the time. This means that there is 10.37% chance that the model will wrongly identify a malignant mole as “Benign“. The specificity (Number of benign moles classified as benign/Total number of benign moles) shows how many benign moles are correctly identified by the model. In this case, the model classifies correctly benign moles 91.11% of the time. Our CNN model seems better at classifying benign moles than malignant ones. This could be due to the dataset containing more values from benign moles (1800 pictures) than from malignant moles (1494 pictures).