The purpose of this project is to create a tool that considering the image of a mole, can calculate the probability that a mole can be malign.

Pre-processing

CNN Model

Model Evaluation : Testing and validation accuracy, confusion matrix

Results presentation: Web App

As mention before the idea is to generate a tool to predict the probability of a malign mole. To do it, I provide Web App.

The web app will have the possibility that a user upload a high quality image of an specific mole. The results will be a prediction about the probability that the given mole be malign in terms of percentage.

**Pre-processing**

* **Data Cleaning**
* **Loading and resizing of images**
* **EDA:** explore different features of the dataset
* **Splitting data**

Skin cancer is the most common human malignancy, is primarily diagnosed visually. Automated classification of skin lesions using images is a challenging task owing to the fine-grained variability in the appearance of skin lesions.

Possible disease states are "Melanoma", "Melanocytic nevus", "Basal cell carcinoma", "Actinic keratosis", "Benign keratosis", "Dermatofibroma", and "Vascular lesion".

I will try to detect 7 different classes of skin cancer using Convolution Neural Network with keras tensorflow and then analyse the result to see how the model can be useful in practical scenario.

* **Data Cleaning**

In this step we check for Missing values. we will fill the null values by their mean.

* **Loading and resizing of images**

We resize the images as the original dimension of images are 450 x 600 x3 which TensorFlow can't handle, so that's why we resize it into 100 x 75.

Most interesting part its always better to see sample of images Below we will show images of each cancer type.

**● EDA**

In this we will explore different features of the dataset, their distrubtions and actual counts.

1. Plot to see distribution of 7 different classes of cell type.

Its seems plot that in this dataset cell type Melanecytic nevi has very large number of instances in comparison to other cell types.

1. Plotting the distribution of localization field.

It seems back, lower extremity, trunk and upper extremity are heavily compromised regions of skin cancer.

1. Now, check the distribution of Age

It seems that there are larger instances of patients having age from 30 to 60.

1. Lets see the distribution of males and females
2. Now lets visualize agewise distribution of skin cancer types

It seems that skin cancer types 0,1, 3 and 5 which are Melanocytic nevi,dermatofibroma,Basal cell carcinoma and Vascular lesions are not much prevalant below the age of 20 years.

* **Splitting data**

**Train Test Split**

In this step we have splitted the dataset into training and testing set of 80:20 ratio.

**Normalization**

I choosed to normalize the x\_train, x\_test by subtracting from their mean values and then dividing by their standard deviation.

**Splitting training and validation split**

I choosed to split the train set in two parts : a small fraction (10%) became the validation set which the model is evaluated and the rest (90%) is used to train the model.

**CNN Model**

* **Model Building**
* **Setting Optimizer**
* **Data Augmentation**
* **Fitting the model**

**I used the Keras Sequential API, where you have just to add one layer at a time, starting from the input.**

**The first is the convolutional (Conv2D) layer.**

**The CNN can isolate features that are useful everywhere from these transformed images (feature maps).**

**The CNN can isolate features that are useful everywhere from these transformed images (feature maps).**

**The second important layer in CNN is the pooling (MaxPool2D) layer. These are used to reduce computational cost, and to some extent also reduce overfitting.**

**Combining convolutional and pooling layers, CNN are able to combine local features and learn more global features of the image.**

**Dropout is a regularization method.This technique also improves generalization and reduces the overfitting.**

**'relu' is the rectifier (activation function max(0,x). The rectifier activation function is used to add non linearity to the network.**

**The Flatten layer is use to convert the final feature maps into a one single 1D vector.**

**This flattening step is needed so that you can make use of fully connected layers after some convolutional/maxpool layers. It combines all the found local features of the previous convolutional layers.**

**In the end i used the features in two fully-connected (Dense) layers which is just artificial an neural networks (ANN) classifier. In the last layer(Dense(10,activation="softmax")) the net outputs distribution of probability of each class.**

* **Model Building**

I used the Keras Sequential API, where you have just to add one layer at a time, starting from the input.

The first is the convolutional (Conv2D) layer.

The second important layer in CNN is the pooling (MaxPool2D) layer.

Dropout is a regularization method.

This flattening step is needed so that you can make use of fully connected layers after some convolutional/maxpool layers. It combines all the found local features of the previous convolutional layers.

In the end i used the features in two fully-connected (Dense) layers which is just artificial an neural networks (ANN) classifier.

* **Setting Optimizer**

Once our layers are added to the model, we need to set up a score function, a loss function and an optimisation algorithm.

We use a specific form for categorical classifications (>2 classes) called the "categorical\_crossentropy".

The most important function is the optimizer. I choosed Adam optimizer.

Adam is a popular algorithm in the field of deep learning because it achieves good results fast.

* **Data Augmentation**

In order to avoid overfitting problem, we need to expand artificially our dataset.

* **Fitting the model**

In this step finally I fit the model into x\_train, y\_train. In this step I have chosen 100 batch size and I have chosen 50 epochs to give the model sufficient epochs to train.

**Model Evaluation**

In this step we will check the testing accuracy and validation accuracy of our model, plot confusion matrix**.**