# MELANOMA SKIN CANCER DETECTION

**UML501 Machine Learning Project Report**

**End-Semester Evaluation**

**Submitted by:**

**102103660 Krishnanshu Shoor**

**102103671 Anvesha Singh**

**BE Third Year, COE**

**Group No: 3COE24**

**Submitted to:**

**Dr Anjula Mehto**

**Computer Science and Engineering Department**

**TIET, Patiala**

**November 2023**

# INDEX

## Introduction

The advent of Convolutional Neural Networks (CNNs) has revolutionized the landscape of medicaldiagnostics, offering a potent avenue for early detection and precise classification of various ailments. In this vein, the endeavor to leverage CNN technology for skin cancer detection stands asa beacon of hope in the realm of dermatology and oncology. Skin cancer, notably melanoma, posesa significant health risk worldwide, with timely detection pivotal for successful treatment outcomes.This project embarks on a pioneering journey, aiming to develop an automated classification system using CNN architecture for the discernment of skin cancer types through the analysis of dermatoscopic images. The urgency lies in reducing the time lag between clinical screening and biopsy report, a critical juncture where delays can impact treatment efficacy and patient outcomes.

Harnessing a dataset sourced from the International Skin Imaging Collaboration (ISIC), comprisingdiverse malignant and benign oncological conditions, this project seeks to train and fine-tune a CNN model capable of distinguishing between nine distinct skin cancer categories. Augmentation techniques have been employed to address class imbalance, ensuring a robust and comprehensive learning process.The architecture of the CNN involves a sequence of layers, each meticulously designed to extract,abstract, and classify intricate patterns within the dermatoscopic images. Leveraging techniques such as convolution, pooling, and dropout, the model aims not only for accuracy but also for resilience against overfitting, enhancing its applicability in real-world scenarios.

The overarching goal is clear: to provide dermatologists and oncologists with a powerful tool thataugments their diagnostic acumen, potentially shortening the biopsy report turnaround time from weeks to a matter of days. Such an achievement not only holds promise in enhancing patient carebut also in combating the morbidity and mortality associated with skin cancer on a global scale.As this project progresses, its outcomes stand poised to make substantial contributions to the intersection of artificial intelligence, medical imaging, and dermatology, potentially reshaping theparadigm of skin cancer diagnosis and treatment planning.

## Problem Statement

Currently, the process of diagnosing skin cancer through biopsies involves a lengthy timeline, spanning from securing a dermatologist appointment to receiving the biopsy report, often taking aweek or more. This elongated duration can significantly impact treatment planning and patient outcomes. The primary aim of this initiative is to dramatically reduce this timeframe to just a fewdays by implementing a predictive model powered by Convolutional Neural Networks (CNNs).By employing CNN technology, this project endeavors to swiftly classify nine different types of skin cancer based on outlier lesion images. The intent is to streamline and expedite the diagnosticprocess, offering a faster alternative to the traditional biopsy method. This accelerated timeline holds immense potential to positively impact the lives of millions of individuals globally.

This innovative approach aims to empower dermatologists and oncologists with a tool that can efficiently analyze dermatoscopic images, allowing for quicker and more accurate identification ofvarious skin cancer types. This advancement could potentially transform the current diagnostic landscape, enabling medical professionals to make informed decisions promptly and initiate timely interventions.By reducing the waiting period for biopsy reports, this initiative not only enhances the speed ofdiagnosis but also holds the promise of improving treatment outcomes. The expedited process facilitated by the predictive model could potentially lead to earlier interventions and tailored treatments, ultimately positively impacting patient care and prognosis.

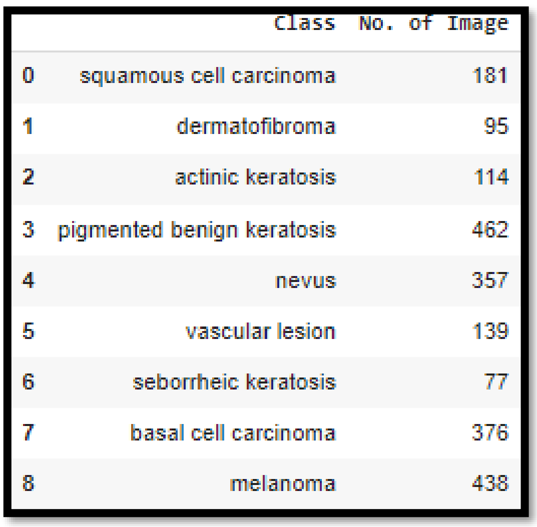
In summary, this project's ultimate goal is to leverage CNN-based technology to revolutionize skincancer diagnosis, significantly shortening the time between initial assessment and biopsy results.This paradigm shift in diagnostics has the potential to benefit a vast number of individuals,ensuring more timely and effective management of skin cancer cases on a global scale.

# Dataset

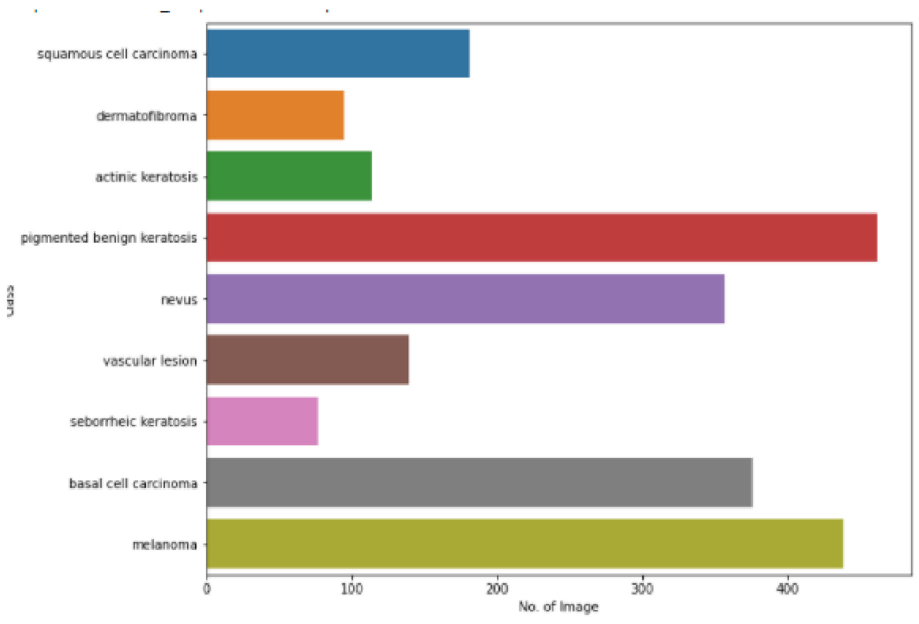
The dataset consists of **2357 images** of malignant and benign oncological diseases, **which were formed from the International Skin Imaging Collaboration (ISIC)**. All images were sorted according to the classification taken with ISIC, and all subsets were divided into the same numberof images.

**Dataset Link :** *https://www.kaggle.com/datasets/rajivaiml/isic-skin-cancer-dataset*

The data set contains the following diseases:

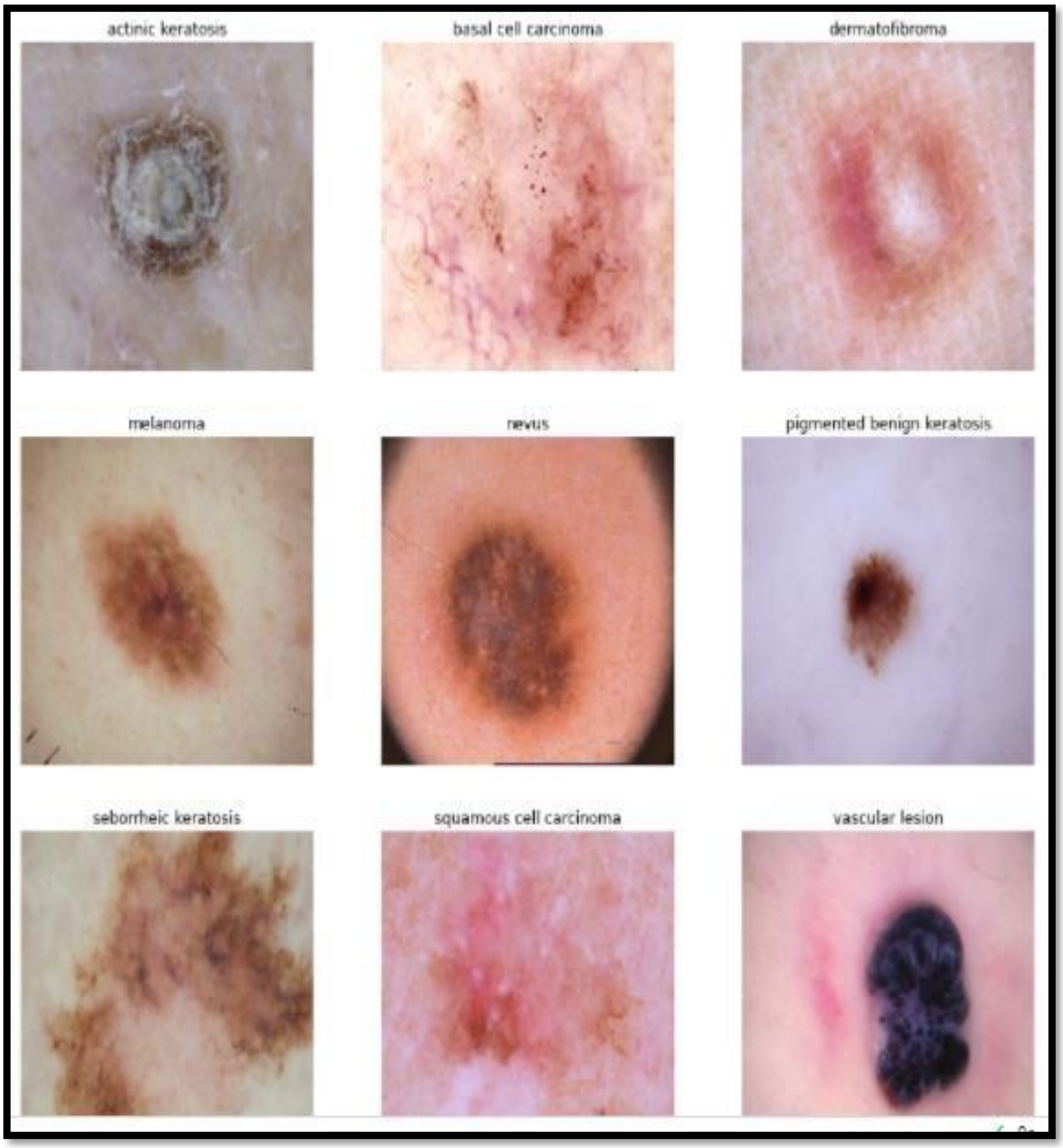


| | Class | No. of Image |
|---|---|---|
| 0 | squamous cell carcinoma | 181 |
| 1 | dermatofibroma | 95 |
| 2 | actinic keratosis | 114 |
| 3 | pigmented benign keratosis | 462 |
| 4 | nevus | 357 |
| 5 | vascular lesion | 139 |
| 6 | seborrheic keratosis | 77 |
| 7 | basal cell carcinoma | 376 |
| 8 | melanoma | 438 |



To overcome the issue of class imbalance, used a python package Augmentor (https://augmentor.readthedocs.io/en/master/) to add more samples across all classes so that noneof the classes have very few samples.

**Sample images from Dataset**



# Methodology/Workflow

1. **Understanding the Problem:**
    - Problem Definition: Melanoma detection using skin images.

      Objective: Develop an accurate model to classify skin lesions as either melanoma or non- melanoma.

2. **Data Preprocessing:**

- Data Loading: Load the dataset containing skin images for different skin conditions.
- Data Cleaning: Check for corrupt or missing images and handle any anomalies in the dataset.
- Data Augmentation: Augment the dataset to address class imbalances and enhance model generalization by applying transformations like rotation, flipping, scaling, and brightness adjustments.

3. **Exploratory Data Analysis (EDA) and Visualization:**

- Class Distribution Visualization: Plot graphs or charts to show the distribution of different classes in the dataset.
- Sample Visualization: Display sample images from each class to understand the characteristics of different skin conditions.
- Statistical Analysis: Perform statistical analysis if needed to gain deeper insights into the dataset.

4. **Model Building:**

- Architecture Selection: Choose a CNN architecture suitable for image classification tasks. Consider architectures like VGG, ResNet, or custom architectures.
- Model Design: Define the layers of the CNN model, including convolutional layers, pooling layers, fully connected layers, and dropout layers.
- Normalization: Normalize the pixel values of the images to a standard scale (e.g., [0, 1] or [- 1, 1]).

5. **Model Training:**

- Dataset Splitting: Split the dataset into training, validation, and test sets.
- Model Compilation: Compile the model by specifying an optimizer (e.g., Adam), a suitable loss function (e.g., categorical cross-entropy), and evaluation metrics (e.g., accuracy).
- Training Process: Train the model on the training set while validating it on the validation set. Monitor metrics like accuracy and loss during training.

**6. Model Evaluation:**

- Performance Metrics: Evaluate the trained model on the test dataset to assess its performance.
- Confusion Matrix and Metrics: Calculate metrics like accuracy, precision, recall, F1-score, and plot the confusion matrix to analyze the model's predictions.

**7. Fine-tuning and Optimization:**

- Hyperparameter Tuning: Fine-tune hyperparameters like learning rate, batch size, and network architecture to improve performance.
- Transfer Learning: Experiment with pre-trained models or transfer learning techniques to leverage features learned from larger datasets (if the dataset size is limited).

**8. Deployment and Integration:**

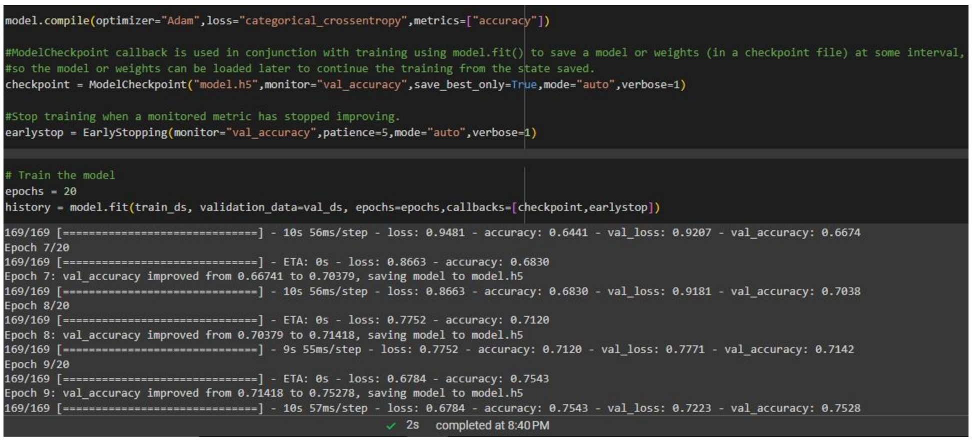
- Model Deployment: Deploy the trained model, making it accessible for real-world use, possibly through an API, web application, or integration with existing systems used by dermatologists.
- User Interface Design (if applicable): Design an intuitive interface for easy interaction with the model, enabling dermatologists to input images and receive predictions.

**9. Documentation and Reporting:**

- Documentation: Document each step of the workflow, including data preprocessing, model architecture, training process, evaluation results, and any challenges faced.
- Report/Presentation: Prepare a comprehensive report or presentation summarizing the methodology, insights gained, model performance, and future recommendations or improvements.

## Results

The model will be trained for the specified **number of epochs (20 in this case).** After each epoch, the model's performance on the validation dataset (val_ds) will be evaluated.

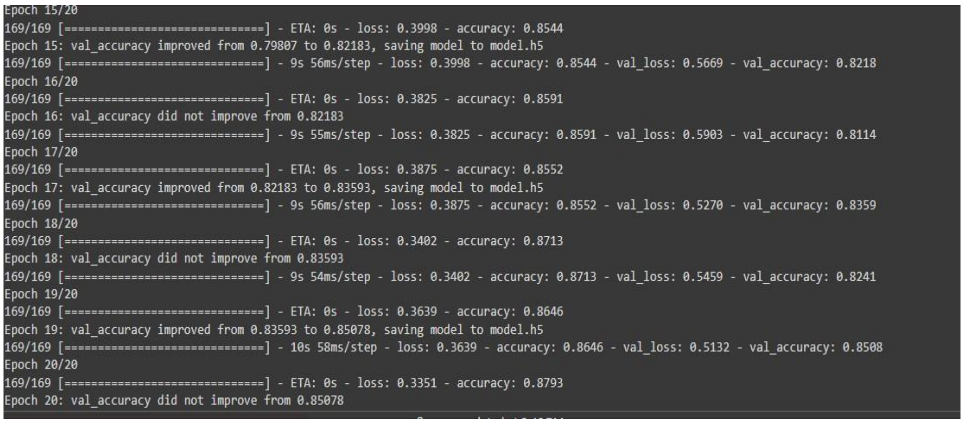


```
model.compile(optimizer="Adam",loss="categorical_crossentropy",metrics=["accuracy"])

#ModelCheckpoint callback is used in conjunction with training using model.fit() to save a model or weights (in a checkpoint file) at some interval,
#so the model or weights can be loaded later to continue the training from the state saved.
checkpoint = ModelCheckpoint("model.h5",monitor="val_accuracy",save_best_only=True,mode="auto",verbose=1)

#Stop training when a monitored metric has stopped improving.
earlystop = EarlyStopping(monitor="val_accuracy",patience=5,mode="auto",verbose=1)
```

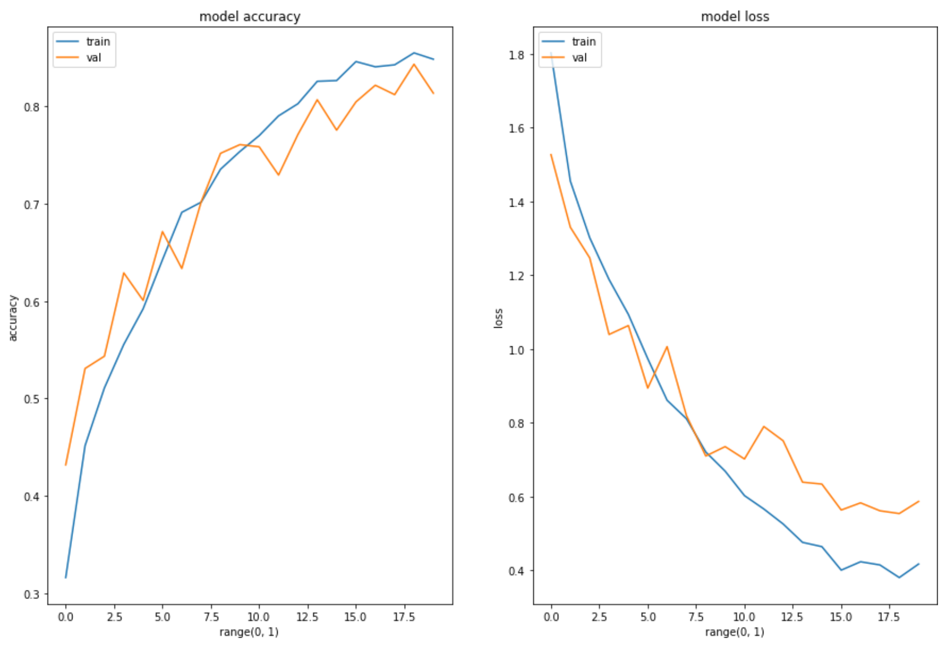
```
# Train the model
epochs = 20
history = model.fit(train_ds, validation_data=val_ds, epochs=epochs,callbacks=[checkpoint,earlystop])
```

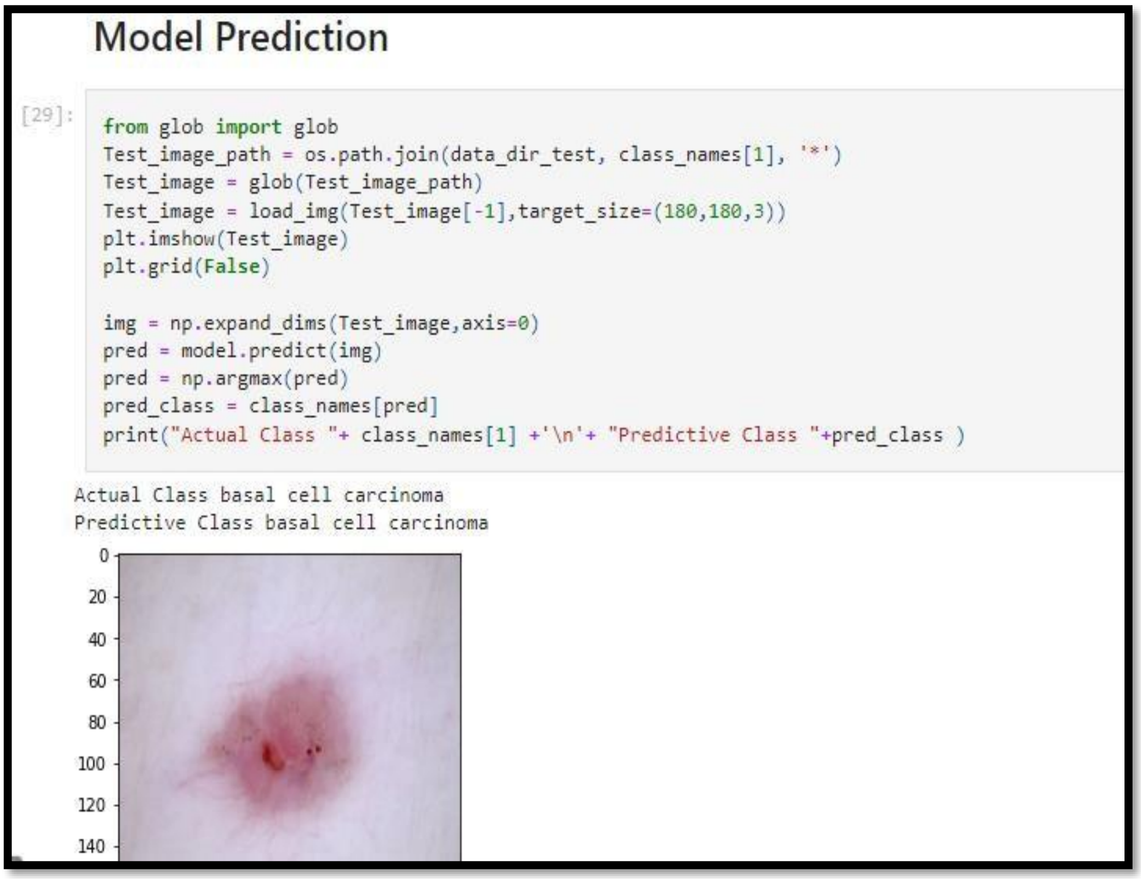
```
169/169 [==============================] - 10s 56ms/step - loss: 0.9481 - accuracy: 0.6441 - val_loss: 0.9207 - val_accuracy: 0.6674
Epoch 7/20
169/169 [==============================] - ETA: 0s - loss: 0.8663 - accuracy: 0.6830
Epoch 7: val_accuracy improved from 0.66741 to 0.70379, saving model to model.h5
169/169 [==============================] - 10s 56ms/step - loss: 0.8663 - accuracy: 0.6830 - val_loss: 0.9181 - val_accuracy: 0.7038
Epoch 8/20
169/169 [==============================] - ETA: 0s - loss: 0.7752 - accuracy: 0.7120
Epoch 8: val_accuracy improved from 0.70379 to 0.71418, saving model to model.h5
169/169 [==============================] - 9s 55ms/step - loss: 0.7752 - accuracy: 0.7120 - val_loss: 0.7771 - val_accuracy: 0.7142
Epoch 9/20
169/169 [==============================] - ETA: 0s - loss: 0.6784 - accuracy: 0.7543
Epoch 9: val_accuracy improved from 0.71418 to 0.75278, saving model to model.h5
169/169 [==============================] - 10s 57ms/step - loss: 0.6784 - accuracy: 0.7543 - val_loss: 0.7223 - val_accuracy: 0.7528
```

✓ 2s completed at 8:40 PM



```
Epoch 15/20
169/169 [==============================] - ETA: 0s - loss: 0.3998 - accuracy: 0.8544
Epoch 15: val_accuracy improved from 0.79807 to 0.82183, saving model to model.h5
169/169 [==============================] - 9s 56ms/step - loss: 0.3998 - accuracy: 0.8544 - val_loss: 0.5669 - val_accuracy: 0.8218
Epoch 16/20
169/169 [==============================] - ETA: 0s - loss: 0.3825 - accuracy: 0.8591
Epoch 16: val_accuracy did not improve from 0.82183
169/169 [==============================] - 9s 55ms/step - loss: 0.3825 - accuracy: 0.8591 - val_loss: 0.5903 - val_accuracy: 0.8114
Epoch 17/20
169/169 [==============================] - ETA: 0s - loss: 0.3875 - accuracy: 0.8552
Epoch 17: val_accuracy improved from 0.82183 to 0.83593, saving model to model.h5
169/169 [==============================] - 9s 56ms/step - loss: 0.3875 - accuracy: 0.8552 - val_loss: 0.5270 - val_accuracy: 0.8359
Epoch 18/20
169/169 [==============================] - ETA: 0s - loss: 0.3402 - accuracy: 0.8713
Epoch 18: val_accuracy did not improve from 0.83593
169/169 [==============================] - 9s 54ms/step - loss: 0.3402 - accuracy: 0.8713 - val_loss: 0.5459 - val_accuracy: 0.8241
Epoch 19/20
169/169 [==============================] - ETA: 0s - loss: 0.3639 - accuracy: 0.8646
Epoch 19: val_accuracy improved from 0.83593 to 0.85078, saving model to model.h5
169/169 [==============================] - 10s 58ms/step - loss: 0.3639 - accuracy: 0.8646 - val_loss: 0.5132 - val_accuracy: 0.8508
Epoch 20/20
169/169 [==============================] - ETA: 0s - loss: 0.3351 - accuracy: 0.8793
Epoch 20: val_accuracy did not improve from 0.85078
```





## Model Prediction

```
[29]:
from glob import glob
Test_image_path = os.path.join(data_dir_test, class_names[1], '*')
Test_image = glob(Test_image_path)
Test_image = load_img(Test_image[-1],target_size=(180,180,3))
plt.imshow(Test_image)
plt.grid(False)

img = np.expand_dims(Test_image,axis=0)
pred = model.predict(img)
pred = np.argmax(pred)
pred_class = class_names[pred]
print("Actual Class "+ class_names[1] +'\n'+ "Predictive Class "+pred_class )
```

```
Actual Class basal cell carcinoma
Predictive Class basal cell carcinoma
```

## Conclusion

Building a CNN-based model to detect melanoma involved several key steps, from data preprocessing to model training and evaluation. The process began with understanding the problem: accurately identifying melanoma in skin images to aid in early detection, crucial for saving lives. The dataset was visualized, and class distributions were analyzed, highlighting the need to address class imbalances.

Augmentation techniques were employed to balance the dataset, ensuring adequate representation across all classes. The CNN model architecture was designed and trained using TensorFlow and Keras, with multiple convolutional layers, pooling, dropout, and dense layers to learn features and make predictions.The model's training progress was monitored through checkpoints and early stopping to prevent overfitting. Over the epochs, the model showed significant improvement in accuracy and loss, indicating its ability to learn and generalize well to unseen data.Visualizing the training curves revealed how the model's accuracy and loss evolved over the training epochs, showcasing its learning process.Finally, the model was put to the test with a sample image from the test dataset, demonstrating its ability to correctly predict the class of the skin lesion.

In conclusion, this project laid out a robust pipeline for melanoma detection using deep learning techniques. While the model showed promising results, continuous refinement and evaluation are crucial for its real-world deployment. Early detection through AI-driven solutions holds immense potential in reducing manual effort in diagnosis, potentially saving lives by enabling timely interventions for melanoma detection.

## <u>Github Link:</u>
*https://github.com/Anvesha9/ML_project*

-