SKIN CANCER CLASSIFICATION USING DEEP LEARNING

UWA BOOT CAMP - FINAL PROJECT

GROUP MEMBERS: Praveen, Lakna, Ryan & John

SKIN LESION CLASSIFICATION

PRESENTATION FLOW

INTRODUCTION → PROBLEM STATEMENT → DATASET OVERVIEW → TABLEAU ANALYSIS → MISSION LEARNING

WHAT IS SKIN CANCER?

DEFINITION:

- Skin cancer is an out-of-control growth of abnormal cells in the epidermis, the outermost skin layer.

CAUSES?

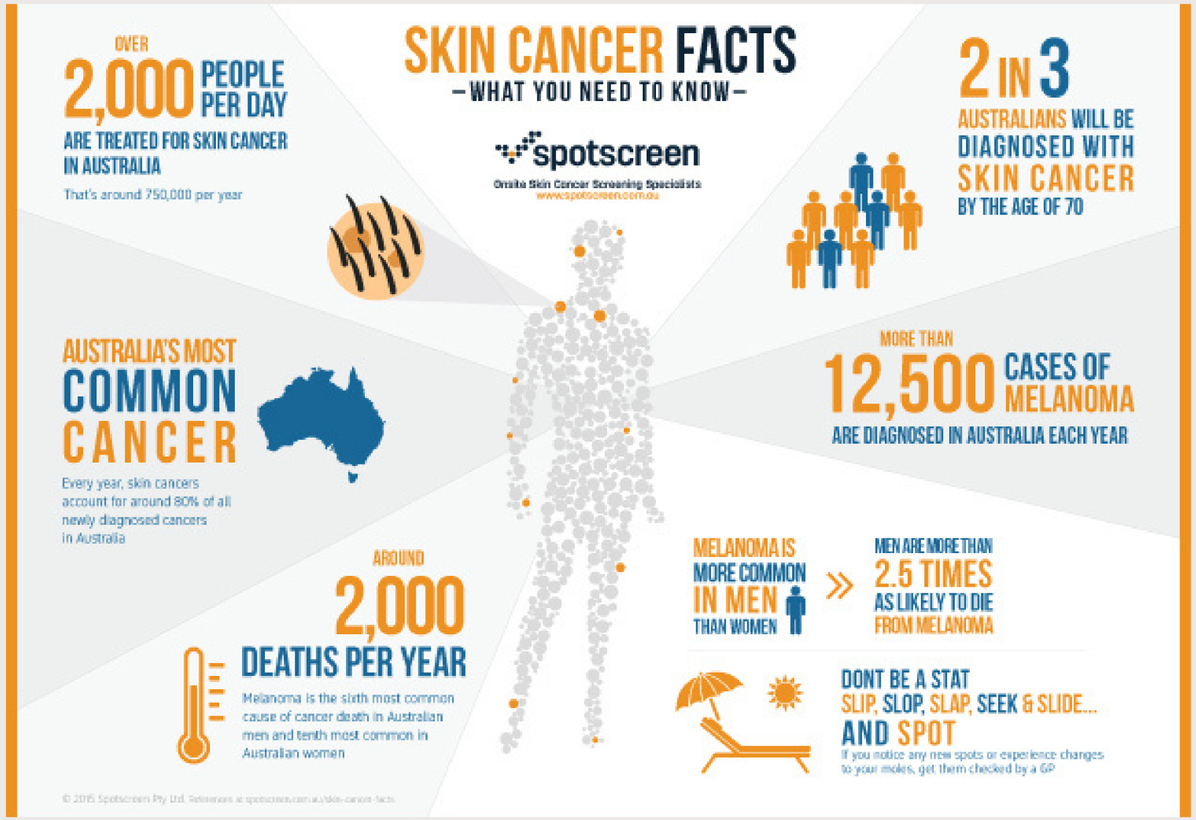
- The most common cause of skin cancer is too much exposure to UV rays. Accumulation of UV rays damages DNA in skin cells.

IMPACT?

- As the DNA in our skin cells is damaged, this causes cells to go 'out of control', which leads to skin cancer

TREATMENT OR DETECTION?

- Skin cancers are almost always surgically removed. Skin cancer is detected through physical examination, which ideally should be every 6 months.



Is This Malignant

YES...

But...

If diagnosed at an early stage, it will not be serious.

PROJECT OBJECTIVE

- Aims to develop a machine learning model for skin cancer diagnosis based on images through the development of a predictive model using Convolutional Neural Network (CNN) to classify types of skin cancer.

DATASET OVERVIEW

DATASETS:

- Kaggle: ISIC-2019
- ISIC 2019: Test Datasets
- ISIC 2019: Training Datasets
- Kaggle: ISIC-2019-Skin-Lesion-Images-For-Classification
- AIHW: Cancer data in Australia
- WHO: WHO Mortality Database
- GCO: Cancer Today

TECHNOLOGIES:

- Tensorflow
- Keras3
- Scikit-learn
- Pandas
- Numpy
- Seaborn
- Tableau Public

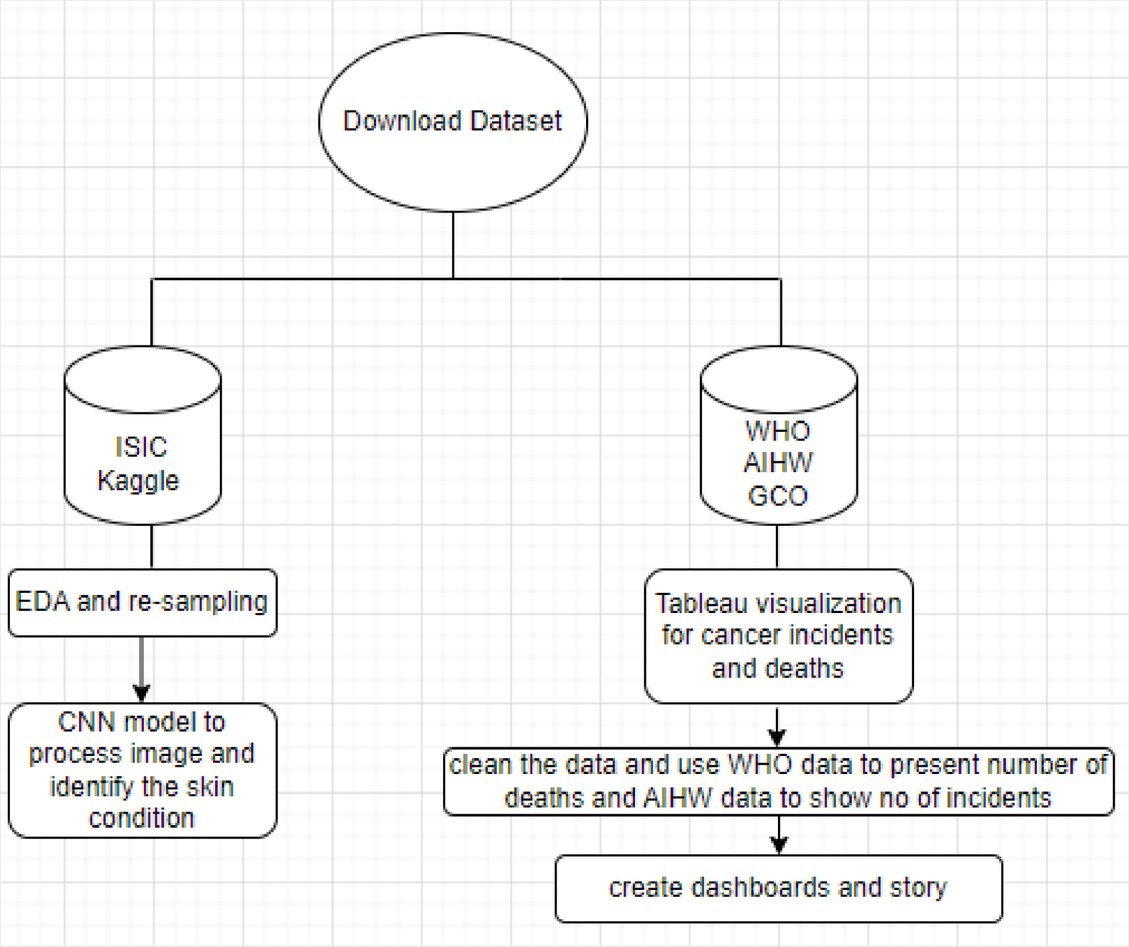
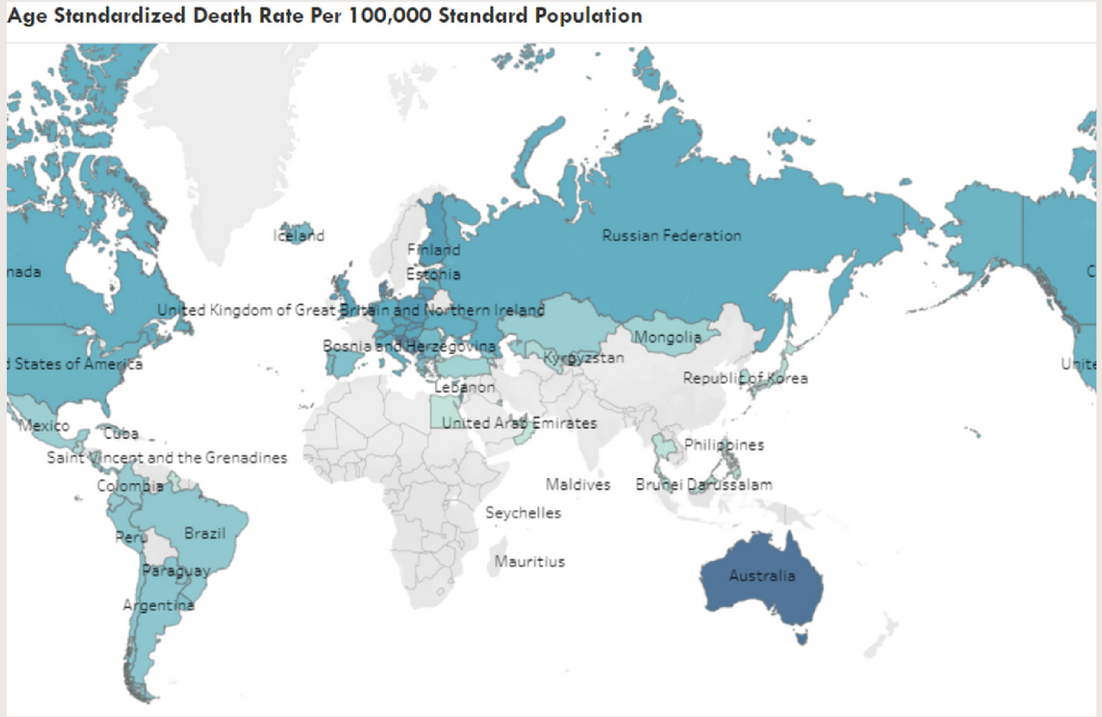
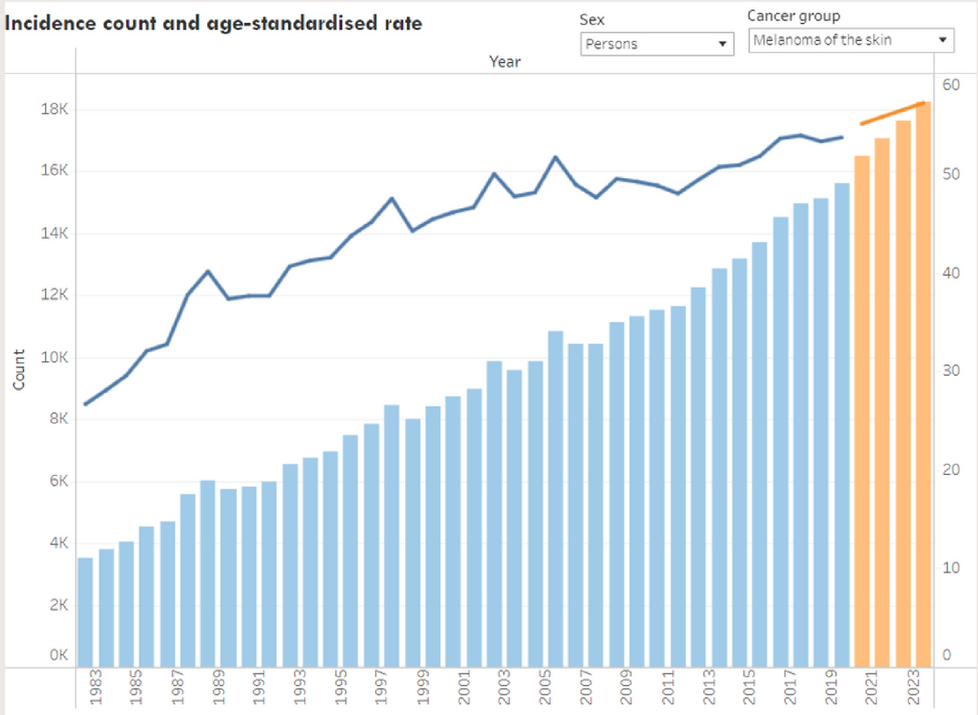


TABLEAU ANALYSIS

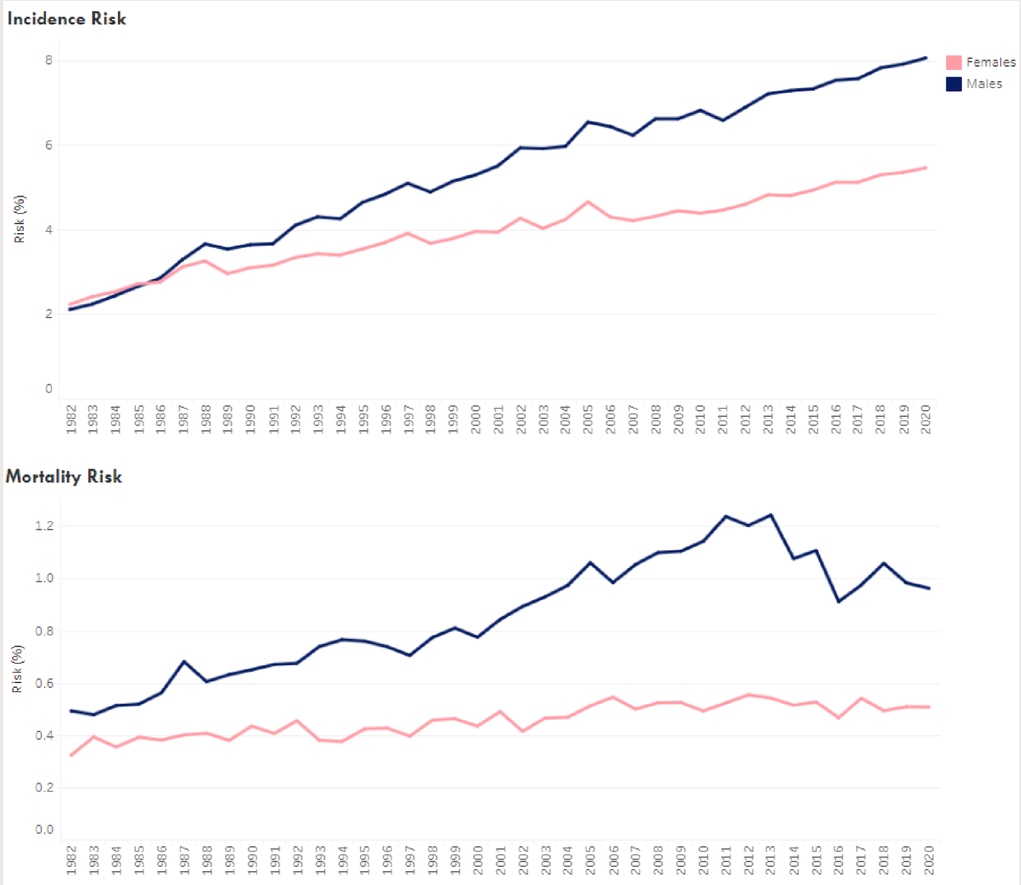


- According to the data from 2019, Australia has a significantly higher death rate compared to other countries shown on this map.

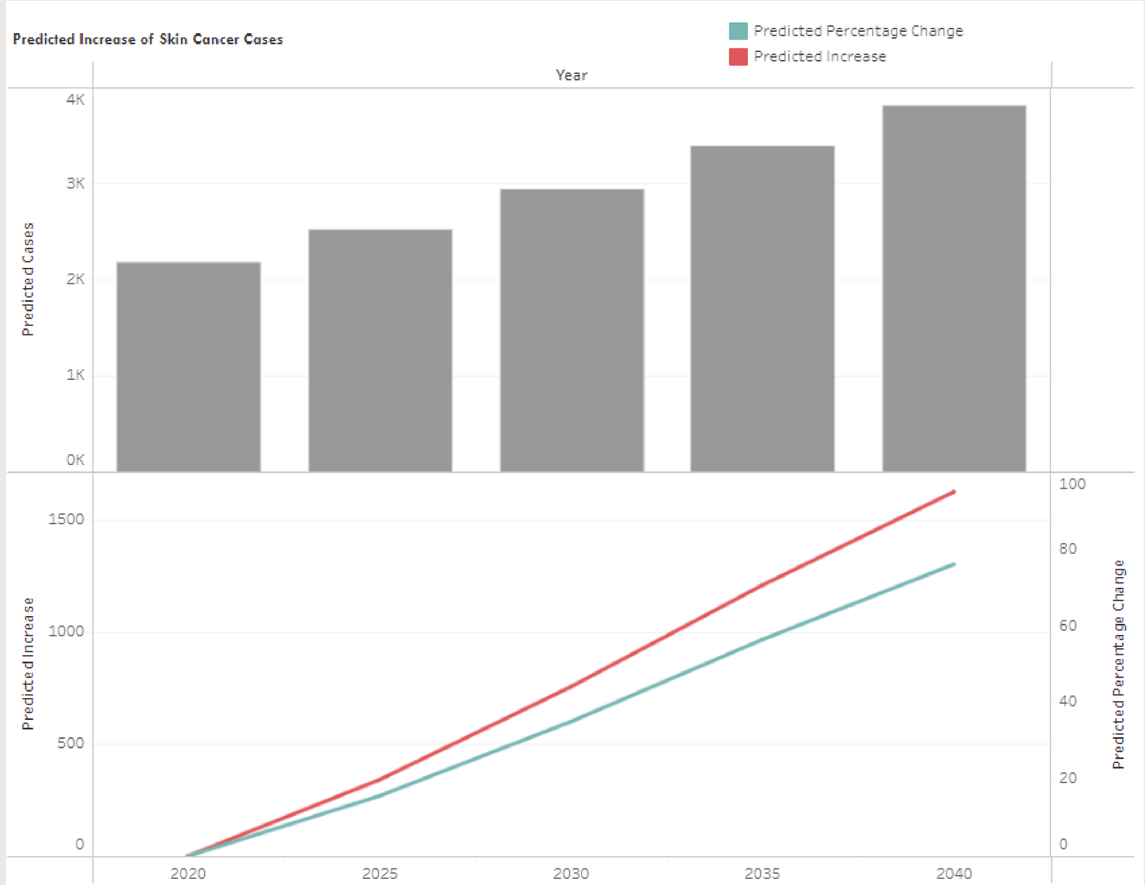


- This graph shows the number of cancer cases diagnosed from 1982 to 2023. The data up to 2019 is actual, while the projections for 2020-2023 are based on historical data. The graph indicates an upward trend in cancer cases over time.

TABLEAU ANALYSIS



- The risk of both incidence and mortality has increased as time passes.



- According to GCO data, the predicted number of melanoma cases in Australia up to 2024 can be seen here

This is an overview of melanoma in Australia

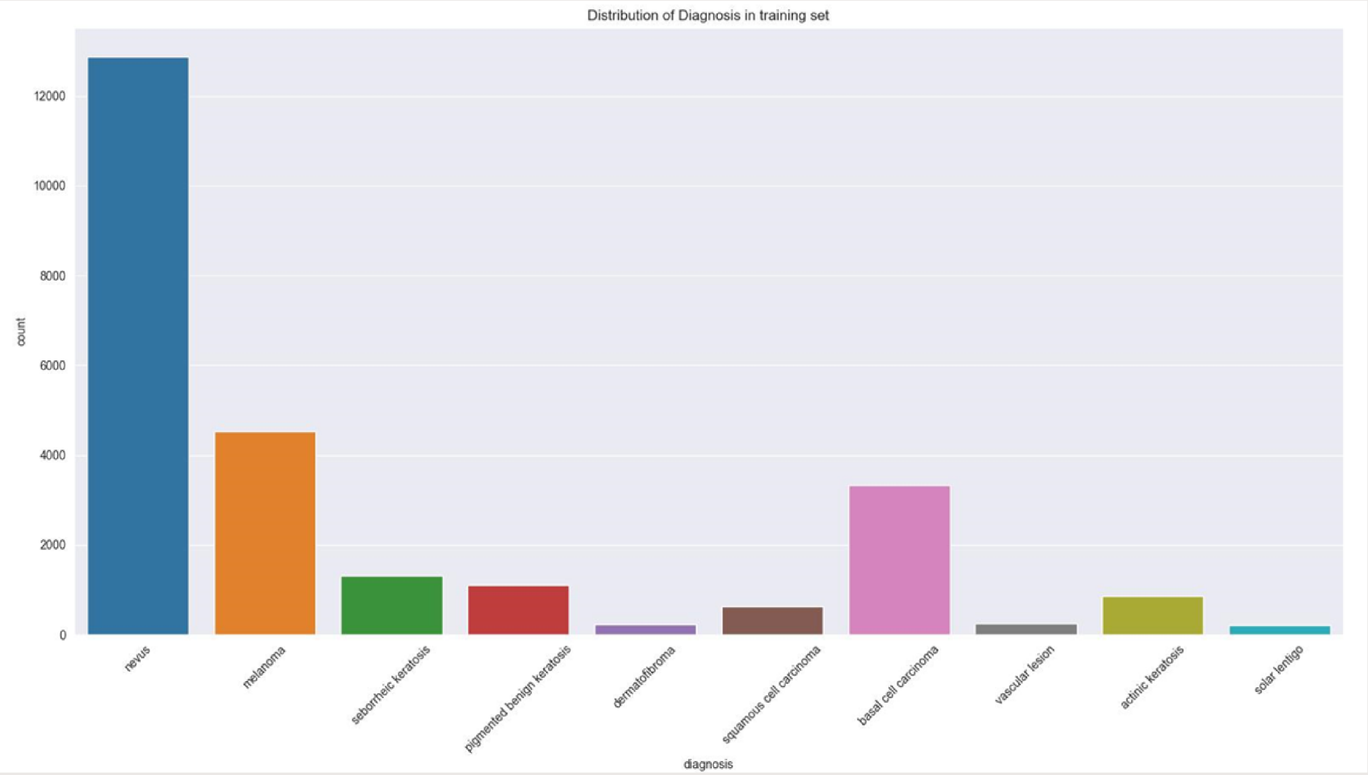
MACHINE LEARNING

EDA Analysis

Some Observations:

- Datasets has 25,313 images of different pixel sizes
- There are very few “NaN” values in the dataset.
- There are not a lot of patients in the training and test datasets and have same age distribution.
- **Dataset is highly imbalance** in case of diagnosis.
- A good correlation between Continuous and Categorical variables are observed.

	columns	NaN statistics (in %)
0	patient_id	98.353796
1	family_hx_mm	97.848486
2	personal_hx_mm	97.809009
3	clin_size_long_diam_mm	97.805061
4	nevus_type	97.706368



Diagnosis	Common_term
nevus	Mole
melanoma	A cancer that develops from skin cells that produce melanin pigment
seborrheic keratosis	A common skin growth and are harmless
pigmented benign keratosis	common noncancerous (benign) skin growth
dermatofibroma	a benign nodule in the skin, typically on the legs, elbows or chest of an adult.
squamous cell carcinoma	second most common form of skin cancer, most commonly formed at sun-exposed areas of the body.
basal cell carcinoma	most common form of skin cancer and lesions commonly arise in sun-exposed areas of the body.
vascular lesion	vascular system compromisation leading to skin marks, tumors, sores, ulcers, or wounds
actinic keratosis	a precancerous skin growth could lead to squamous cell carcinoma.
solar lentigo	a harmless patch of darkened skin results from exposure from UV radiation.

CNN Architecture

Model: Sequential

Model structure:

Input layer: Images

1st Block:

- 1 Conv2D layer
- 1 Max Pooling layer.

2nd - 4thBlock:

- 2 Conv2D layers
- 1 MaxPooling layer

5 Fully connected layers

Hyperparameters:

Optimiser = Adamax

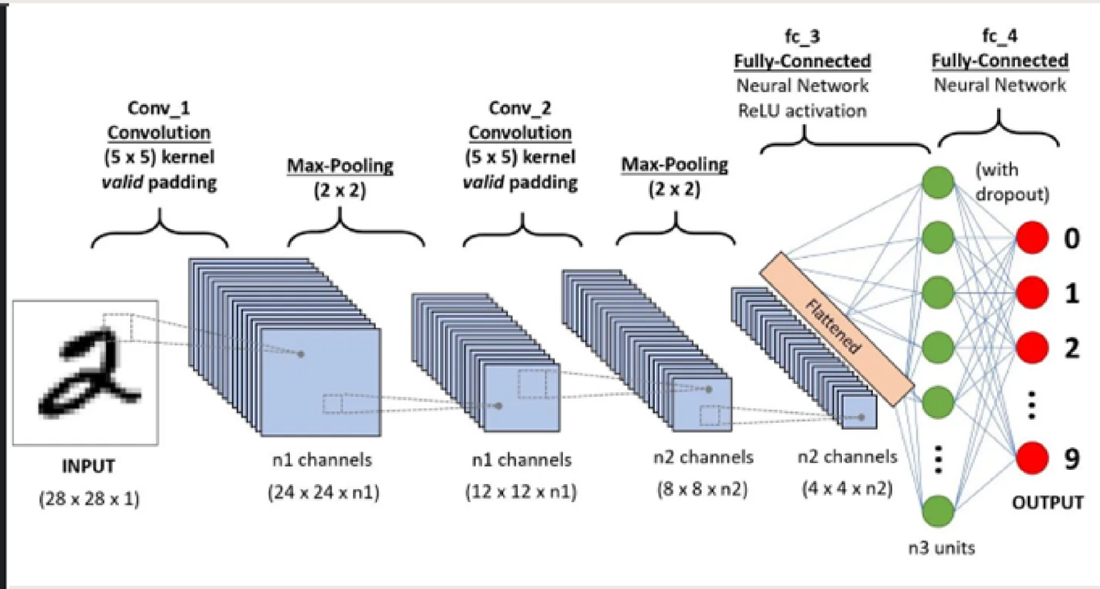
Loss Function = Sparse categorical crossentropy

Callbacks

Early stopping

Reducing learning

```
def create_sequential_model(model):
    model = model
    model.add(Input(shape=[180, 180, 3]))
    model.add(layers.Conv2D(32, (3, 3), activation='relu', padding='same',
kernel_initializer='he_normal'))
    model.add(layers.MaxPooling2D())
    model.add(layers.BatchNormalization())
    model.add(layers.Conv2D(64, (3, 3), activation='relu', padding='same',
kernel_initializer='he_normal'))
    model.add(layers.Conv2D(64, (3, 3), activation='relu', padding='same',
kernel_initializer='he_normal'))
    model.add(layers.MaxPooling2D())
    model.add(layers.BatchNormalization())
    model.add(layers.Conv2D(128, (3, 3), activation='relu', padding='same',
kernel_initializer='he_normal'))
    model.add(layers.Conv2D(128, (3, 3), activation='relu', padding='same',
kernel_initializer='he_normal'))
    model.add(layers.MaxPooling2D())
    model.add(layers.BatchNormalization())
    model.add(layers.Conv2D(256, (3, 3), activation='relu', padding='same',
kernel_initializer='he_normal'))
    model.add(layers.Conv2D(256, (3, 3), activation='relu', padding='same',
kernel_initializer='he_normal'))
    model.add(layers.MaxPooling2D())
    model.add(layers.Flatten())
    model.add(layers.Dropout(rate=0.2))
    model.add(layers.Dense(units=256, activation='relu',
kernel_initializer='he_normal'))
    model.add(layers.BatchNormalization())
    model.add(layers.Dense(units=128, activation='relu',
kernel_initializer='he_normal'))
    model.add(layers.BatchNormalization())
    model.add(layers.Dense(units=64, activation='relu',
kernel_initializer='he_normal'))
    model.add(layers.BatchNormalization())
    model.add(layers.Dense(units=32, activation='relu',
kernel_initializer='he_normal',
                           kernel_regularizer=regularizers.L1L2()))
    model.add(layers.BatchNormalization())
    model.add(layers.Dense(units=8, activation='softmax',
kernel_initializer='glorot_uniform', name='classifier'))
    optimizer = Adamax(learning_rate=0.001)
    model.compile(loss='sparse_categorical_crossentropy',
                  optimizer=optimizer,
                  metrics=['accuracy'])
    model.build((None, 180, 180, 3))
    print(model.summary())
    return model
early_stop = EarlyStopping(
    monitor='val_loss',
    patience=10,
    verbose=1,
    mode='auto'
)

reduce_lr = ReduceLROnPlateau(
    monitor='val_loss',
    factor=0.1,
    patience=3,
    verbose=1,
    mode='auto'
)

callback =
tf.keras.callbacks.ModelCheckpoint(filepath='model/best_model.h5',
                                   monitor='val_acc',
mode='max',
                                   save_best_only=True,
verbose=1)
```



Layer Definitions:

Convolution Layer: *a layer where convolution(mathematical) operation to the input data are performed.*

Pooling Layer: *Pooling layers are used to reduce the dimensions of the feature maps.*

Batch Normalization layer: take the outputs from the first hidden layer and normalize them before passing them on as the input of the next hidden layer.

Results

	Model V01	Model V02	Model V03	Model V04
Datasets	ISIC-2019	ISIC_2019_Training_GroundTruth	isic-2019-skin-lesion-images-for-classification	ISIC_2019_Training_GroundTruth
Methodolody for generation of x_test and x_train data	image_dataset_from_directory	ImageDataGenerator and datagen.flow	image_dataset_from_directory	sci-kit_learn train_test
Oversampling or Undersampling	No	Yes (Droped 30% of NV datasets)	No	Yes (RandomOverSampler)
Shape of Features	(22800, 180,180,3)	(21469,180,180,3)	(25331, 180,180,3)	(103000, 71, 71, 3)
Accuracy	0.576	0.651	0.706	0.963

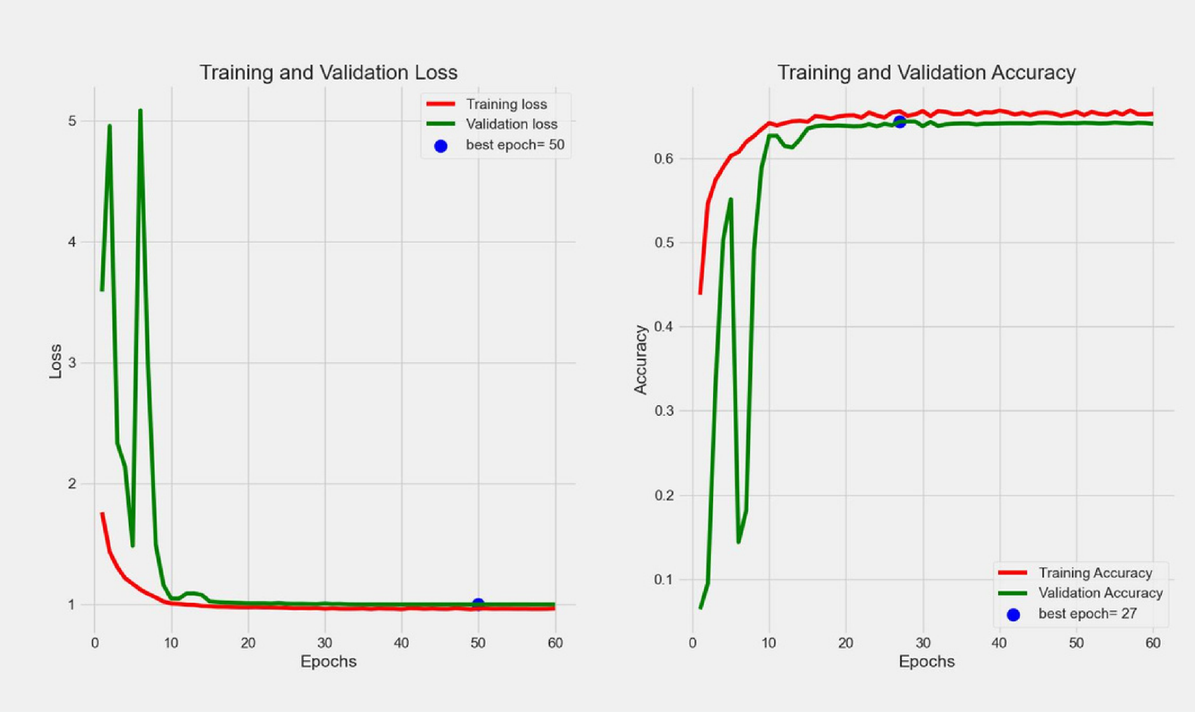
```
datagen = ImageDataGenerator(rescale=1. / 255,
                             validation_split=0.2,
                             rotation_range=20,
                             width_shift_range=0.2,
                             height_shift_range=0.2,
                             shear_range=0.005,
                             zoom_range=0.005,
                             horizontal_flip=True,
                             vertical_flip=True,
                             fill_mode='nearest')

train_generator = datagen.flow(X_aug, Y_aug, batch_size=128, shuffle=True, seed=133, ignore_class_split=True,
                               subset="training")

validation_generator = datagen.flow(X_aug, Y_aug, batch_size=128, shuffle=True, seed=133, ignore_class_split=True,
                                    subset="validation")
```

```
x_new = (X_aug.reshape([-1, 71*71*3]))
oversample = RandomOverSampler()
X_aug_new,Y_aug_new = oversample.fit_resample(x_new,Y_aug)
X_aug_new = np.array(X_aug_new).reshape(-1,71,71,3)

print('Shape of X :',X_aug_new.shape)
print('Shape of Y :',Y_aug_new.shape)

X_aug_new = (X_aug_new-np.mean(X_aug_new))/np.std(X_aug_new)

np.save( file: 'Cleaned_Data/X_aug_new', X_aug_new)
np.save( file: 'Cleaned_Data/Y_aug_new', Y_aug_new)

X_train, X_test, y_train, y_test = train_test_split( *arrays: X_aug_new,Y_aug_new, test_size=0.2, random_state=78)
```

```
# for training data
train_file_dir = 'Project_Datasets/isic-2019-skin-lesion-images-for-classification'
image_size = (180, 180)
train_ds = image_dataset_from_directory(
    train_file_dir,
    validation_split=0.2,
    subset='training',
    seed=133,
    shuffle=True,
    batch_size=None,
    image_size=image_size,
)

# for testing data
image_size = (180, 180)
val_ds = image_dataset_from_directory(
    train_file_dir,
    validation_split=0.2,
    subset='validation',
    seed=133,
    shuffle=True,
    batch_size=None,
    image_size=image_size,
)
```

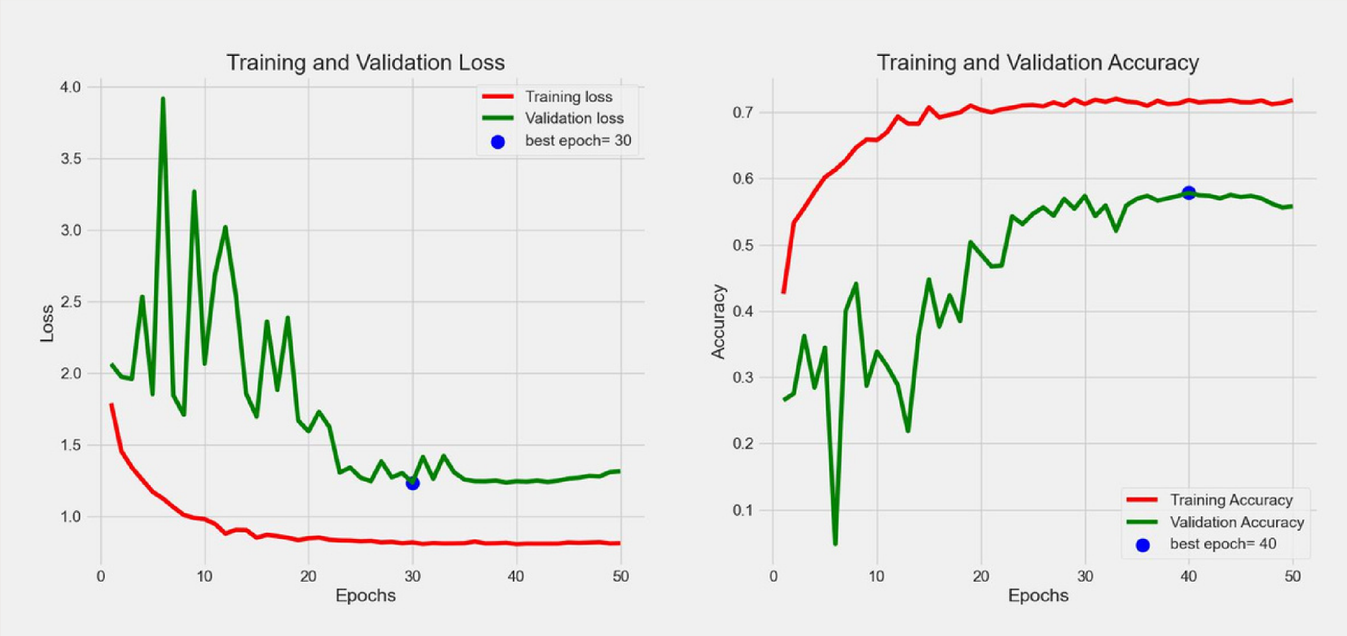
Training History

Model V01



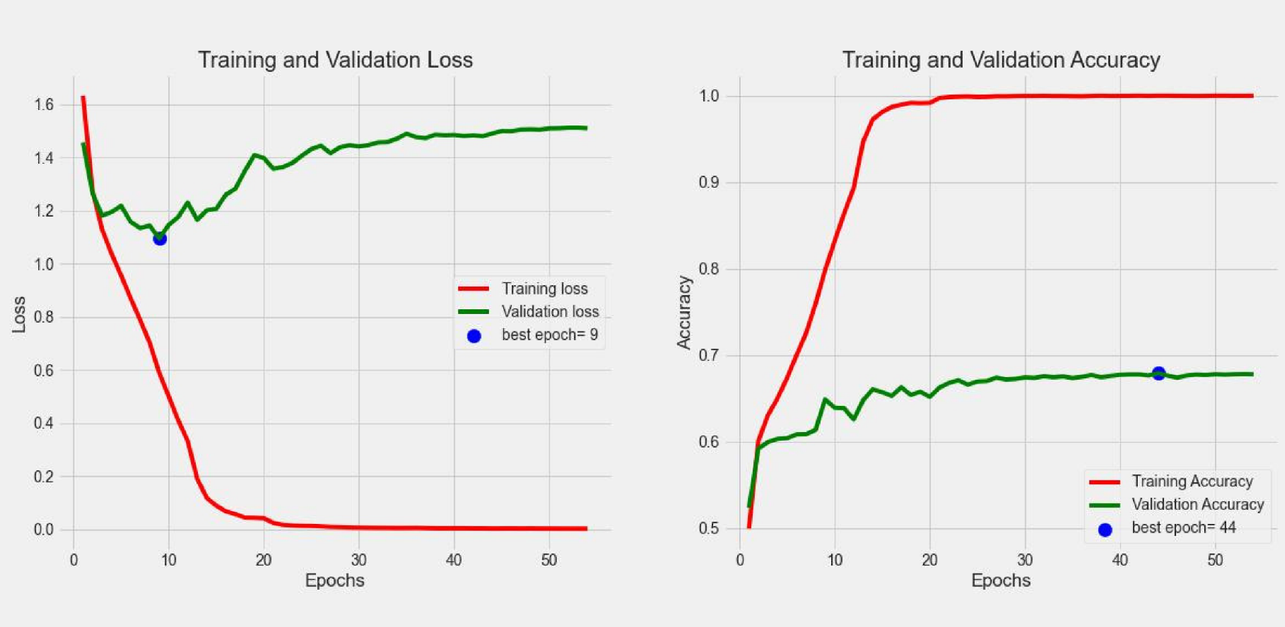
	Model V01
Datasets	ISIC-2019
Methodolody for generation of x_test and x_train data	image_dataset_from_directory
Oversampling or Undersampling	No
Shape of Features	(22800, 180,180,3)
Accuracy	0.576

Model V02



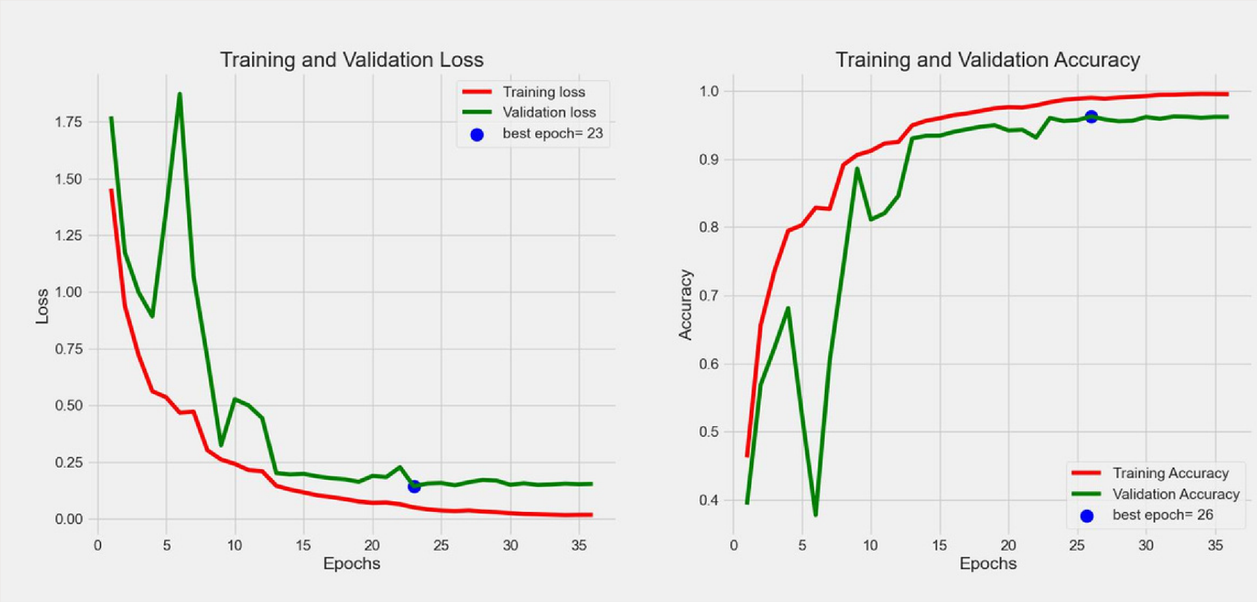
	Model V02
Datasets	ISIC_2019_Training_GroundTruth
Methodolody for generation of x_test and x_train data	ImageDataGenerator and datagen.flow
Oversampling or Undersampling	Yes (Droped 30% of NV datasets)
Shape of Features	(21469,180,180,3)
Accuracy	0.651

Model V03



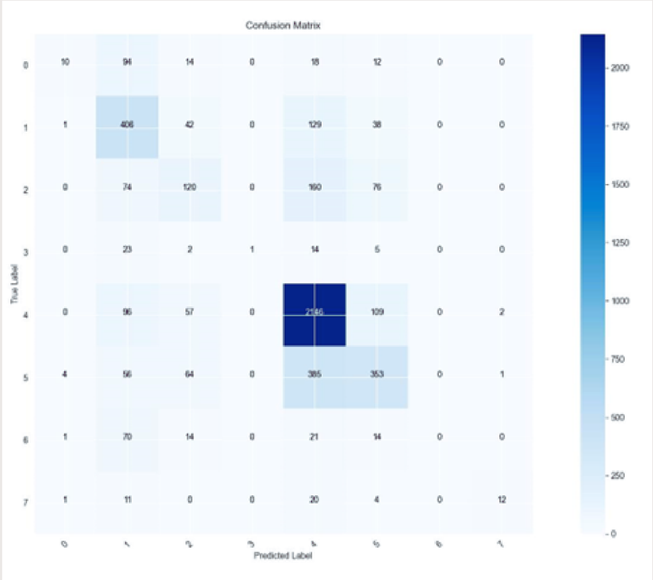
	Model V03
Datasets	isic-2019-skin-lesion-images-for-classification
Methodolody for generation of x_test and x_train data	image_dataset_from_directory
Oversampling or Undersampling	No
Shape of Features	(25331, 180,180,3)
Accuracy	0.706

Model V04

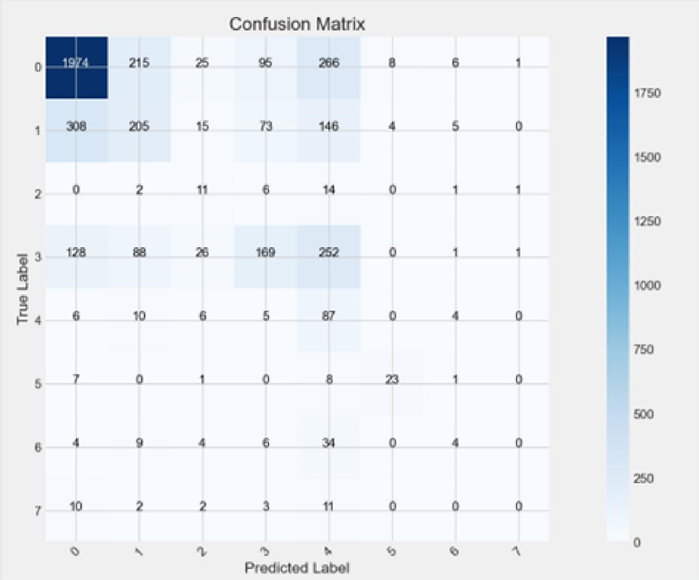


	Model V04
Datasets	ISIC_2019_Training_GroundTruth
Methodolody for generation of x_test and x_train data	sci-kit_learn train_test
Oversampling or Undersampling	Yes (RandomOverSampler)
Shape of Features	(103000, 71, 71, 3)
Accuracy	0.963

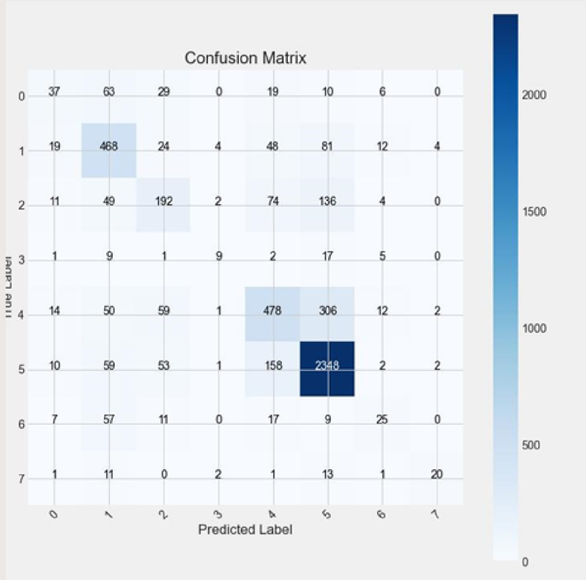
Confusion Matrix



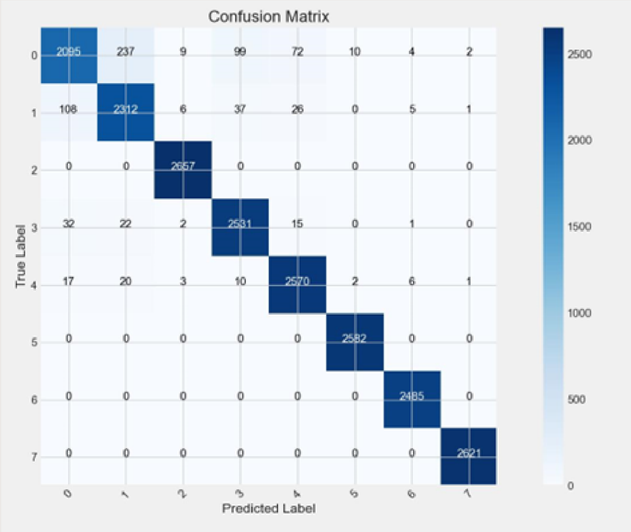
Model V01



Model V02



Model V03



Model V04

Labels

```
lesion_type_dict = {
    'nevus': "NV",
    'melanoma': "MEL",
    'actinic keratosis': "AK",
    'pigmented benign keratosis': "BKL",
    'basal cell carcinoma': "BCC",
    'vascular lesion': "VASC",
    'squamous cell carcinoma': 'SCC',
    'dermatofibroma': "DF",
    'solar lentigo': "UNK",
    'seborrheic keratosis': "BKL"
}

label_mapping = {
    0: 'NV',
    1: 'MEL',
    2: 'AK',
    3: 'BKL',
    4: 'BCC',
    5: 'VASC',
    6: 'SCC',
    7: 'DF',
}
```

Summary

Based on analysis, Model V04 performance is better in comparison with other three Models, suggesting that RandomOverSampling Methodology is best suited for this of datasets.

Limitations

Performance of Model V04 for datasets sourced from external websites, needs improvement as model is trained on a low scaled pixel images.

Model needs to be trained on different types of datasets, such as dicom, tiff.

Further Work

As part of this project, CNN Sequential Model is explored for accuracy and efficiency. Other model CNN architectures, such as Efficientnetb1, fasiai, K-fold methods could also be explored for accuracy and efficiency in Skin Condition Classification.

PyTorch libraries could also be expolored for accelerated processing.

QUESTION