

Configuration Manual

MSc Research Project

Data Analytics

Mayuresh Londhe

Student ID: X20137265

School of Computing

National College of Ireland

Supervisor: Dr. Christian Horn



**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name:** | **Mayuresh Londhe** | | |
| **Student ID:** | **X20137265** | | |
| **Programme:** | **Msc. Data Analytics** | **Year:** | **2020/2021** |
| **Module:** | **Research Project** | | |
| **Lecturer:** | **Dr. Christian Horn** | | |
| **Submission Due Date:** | **16/08/2021** | | |
| **Project Title:** | **Classification of Eye Diseases using Hybrid CNN-RNN Models.** | | |
| **Word Count:** | **1542 Page Count: 12** | | |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

|  |  |
| --- | --- |
| **Signature:** | ……………………………………………………………………………………………………………… |
| **Date:** | 16/08/2021 |

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

|  |  |
| --- | --- |
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

|  |  |
| --- | --- |
| **Office Use Only** | |
| Signature: |  |
| Date: |  |
| Penalty Applied (if applicable): |  |

Configuration Manual

Mayuresh Londhe

Student ID: X20137265

# Introduction

This Configuration Manual consists of all the perquisites required to reproduce the research and its outcomes on individual environment. The software and the hardware requirement along with a snapshot of code for Data Import and Pre-processing, Data Augmentation, Exploratory Data Analysis, all the models-built K- Fold Cross Validation and Evaluation are included.

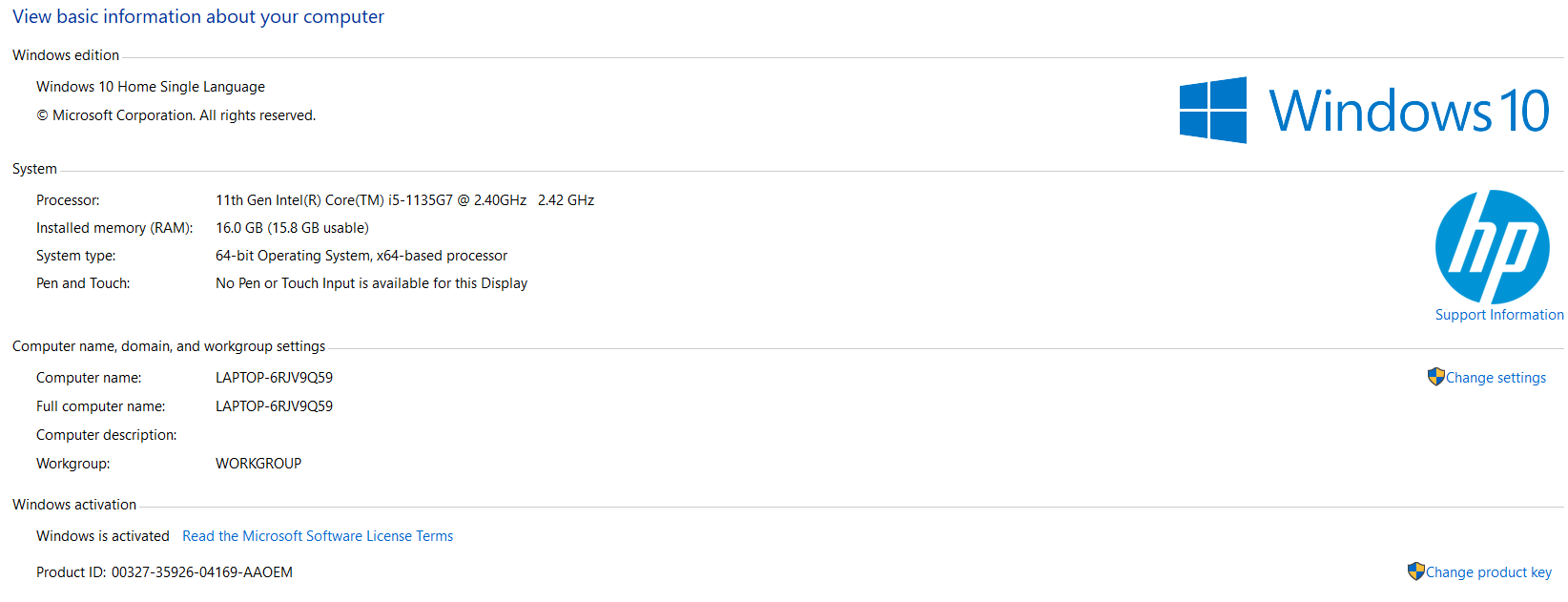
The structure of the report is as follows, Section 2, gives the information about environment configuration. Section 3, provides detail about data collection. Section 4 is data exportation consists of Data Pre-processing and Exploratory Data Analysis. Data Augmentation is explained in section 5. Section 6 provides the details about Data splitting to get equal images in each class for training, validation and testing phase, data transformations and feature label separation performed. Section 7 provides the details about the models built. Section 8, explains the implementation of K-Fold Validation. Section 9, explains how results are computed and visualized.

# Environment

This section provides the details of Software and Hardware requirements to implement the research done.

## Hardware Requirements

Below Figure 1, provides the hardware specifications required. Intel i5-1135G7 is the 11th Generation Intel Core CPU @ 2.42 GHz, 16 GB installed DDR4 RAM Memory at speed of 3200 Mhz, 64 Bit Windows 10 operating System, 512 GB SSD.



**Figure 1: Hardware Requirements**

## Software Requirements

* Anaconda Navigator for Windows (Version 1.9.12)
* Jupyter Notebook (Version 6.0.3)
* Python (Version 3.8.3)

# Data Collection

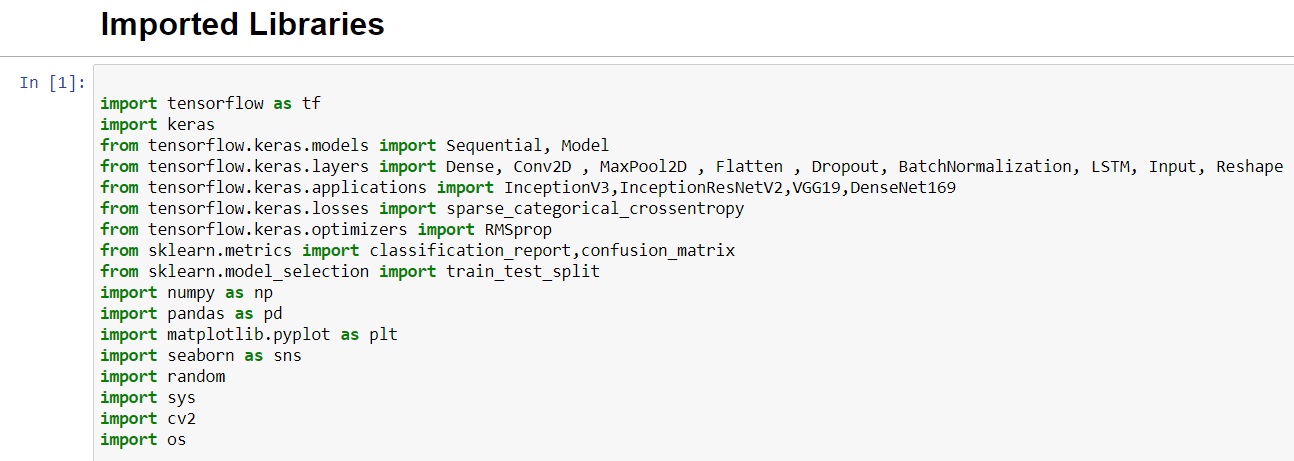
The dataset is taken from Kaggle public repository. <https://www.kaggle.com/jr2ngb/cataractdataset> is the link for the dataset. There are 601 images in dataset there is one extra image in the Glaucoma class which is removed therefore there would 300 images in Normal and 100 images each in Cataract, Glaucoma and Retina diseases class. Link given below can also be used.

https://studentncirl-my.sharepoint.com/:f:/g/personal/x20137265\_student\_ncirl\_ie/EtKlBNkXNIlOl7rKmHSqc30BbycJ40sThYMa2yjvzK9ufA?e=9iLT3F

# Data Exportation

## Importing the Libraries

All the Python libraries required to implement the entire project are listed in Figure 2.



**Figure 2: Required Python Libraries**

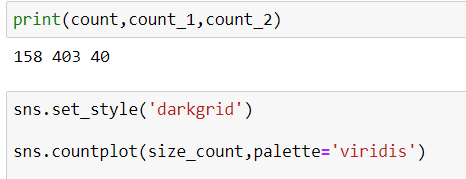
## Exploratory Data Analysis

### To Check Individual Image Size

The Figure 3 and 4 represents the code to visualize and count the number of images for each size. There are 3 different size images are available (1632 x 2464), (1224 x 1848) and (1728 x 2592).



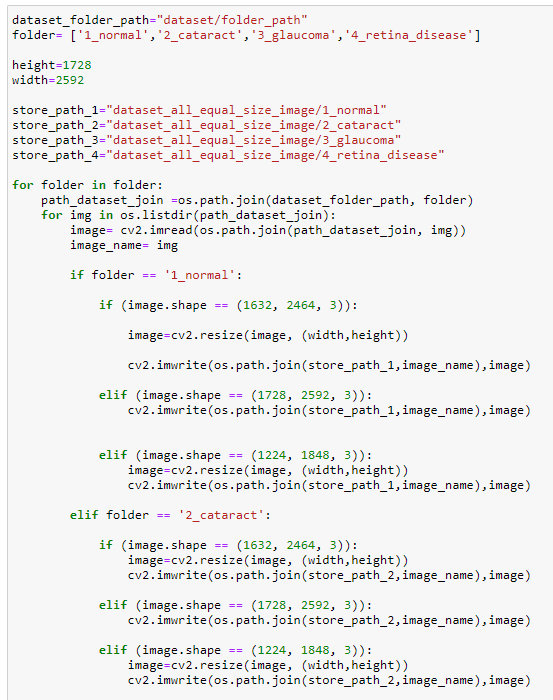
**Figure 3: EDA for Checking Individual Image Size**



**Figure 4: Print the Count and Visualization**

## Data Pre-processing

As seen in Figure 3 there are 158 that have (1632 x 2464) size, 403 images with (1728 x 2592) size and 40 images with size (1224 x 1848). Here the (1224 x 1848) and (1632 x 2464) images are converted to (1728 x 2592). The Figure 5, illustrate the code to convert all the images to one size. Here, original dataset path need to be provided along with that need to create new folder with name dataset\_all\_equal\_size\_images which contains four empty folders with the name 1\_normal, 2\_cataract,3\_glaucoma and 4\_retina\_diseases. Path of these four folders also used to store the resized Images.





**Figure 5: Code to Convert All Images Equal Size.**

**all equal size images :** [**https://studentncirl-my.sharepoint.com/:f:/g/personal/x20137265\_student\_ncirl\_ie/EvAXoG3kxslOksAk3-HU01UBdAYWLYXOZNM7CxiA-HbIUQ?e=ViIG7T**](https://studentncirl-my.sharepoint.com/:f:/g/personal/x20137265_student_ncirl_ie/EvAXoG3kxslOksAk3-HU01UBdAYWLYXOZNM7CxiA-HbIUQ?e=ViIG7T)

# Data Augmentation:

To accomplish the class balance across all labels. The images of the Cataract, Glaucoma and Retina Diseases need to be augmented using horizontal filliping and horizontally and vertically together. To achieve this, need to create a copy of the dataset\_all\_equal\_size\_images dataset and rename it as all\_equal\_300\_images. File path of the copied and renamed dataset along with 3 folder names for which images need to augmented is passed.



**Figure 6: Code to Perform Data Augmentation**

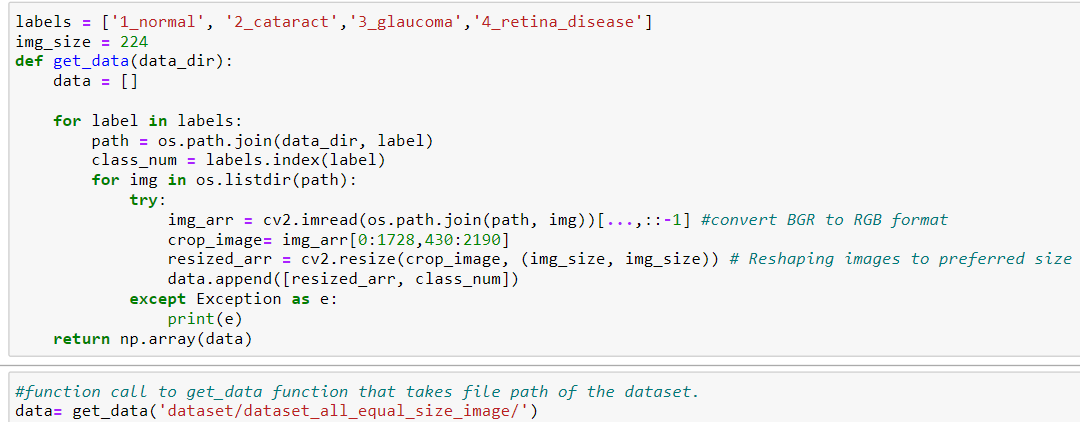
After this process, 3 datasets will be present original dataset, dataset\_all\_equal\_size\_images dataset that would be used for implementing the models on imbalanced dataset and all\_equal\_300\_images will be used as a balanced dataset.

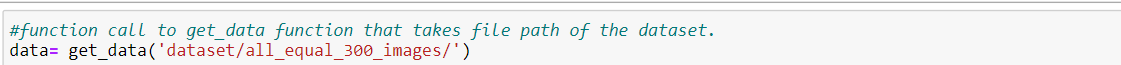
all equal 300 images: <https://studentncirl-my.sharepoint.com/:f:/g/personal/x20137265_student_ncirl_ie/Eg7mkMbUkgFKiPsNsBfMXUQBG8tTdqNX5Fhzc_tnPREP0Q>

# Data Preparation

## Dataset Import for Model building

The get\_data() is defined that takes the path of dataset, depending upon whether model is needed to be implemented on balanced or imbalanced dataset. Along with data import, the function performs the conversion of the images imported in BGR format to RGB, Black background reduction and image reseize to 224 x 224 dimensions.

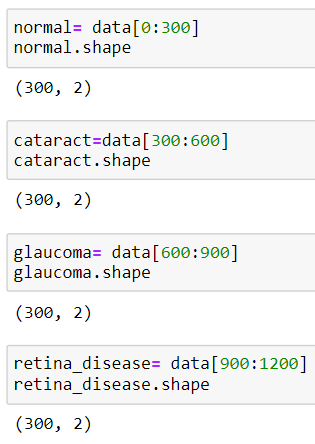
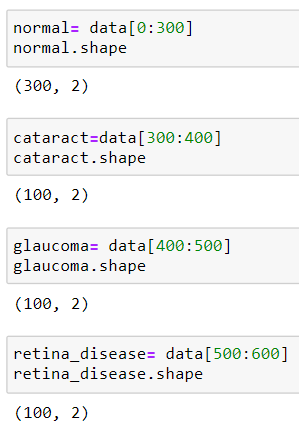




**Figure 7: Dataset Import for Model Building**

## Data Splitting

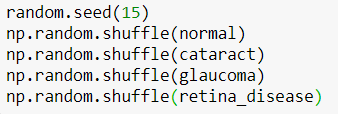
The Figure 8, below provides the code for splitting the imported data into four different data object that contains images for the respective class. This is done to ensure that number of images in each class after the training validation and testing phase would be equal.

**Figure 8: Left: Data Splitting for Imbalanced dataset Right: Data Splitting for Balanced data**

## Dataset Random Shuffle

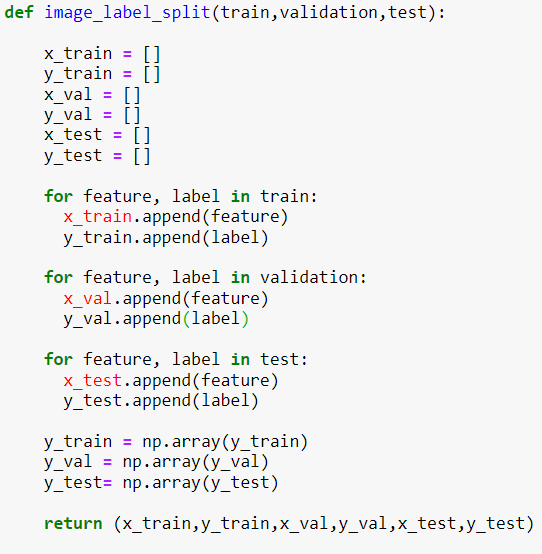
Figure 9, represents the code to shuffle the separated dataset for each class. This code block will remain same for both Imbalanced and Balanced datasets.



**Figure 9: Dataset Random Shuffle**

## Feature and Label Separation

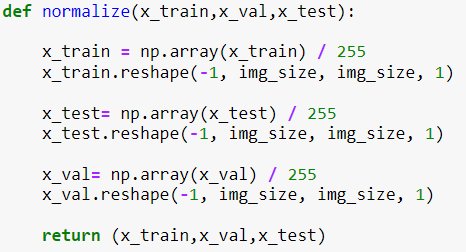
The model takes input as images and labels, therefore the function image\_label\_split() takes the three arguments train, validation and test.



**Figure 10: Feature and Label Separation**

## Data Transformation

The transformation method used is normalization, the function takes the image features as input for all three phases. The functions mentioned in Sections 6.3 and 6.4 need to be executed before model compilation.



**Figure 11: Normalization**

# Models Architecture Implementation

All the implemented models are used on both Imbalanced and Balanced dataset. All the models are implemented using pre-trained weights.

## InceptionV3-LSTM



**Figure 12: InceptionV3-LSTM Model Implementation**

## InceptionResNetV2-LSTM



**Figure 13: InceptionResNetV2-LSTM Model Implementation**

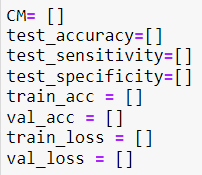
## DenseNet169-LSTM



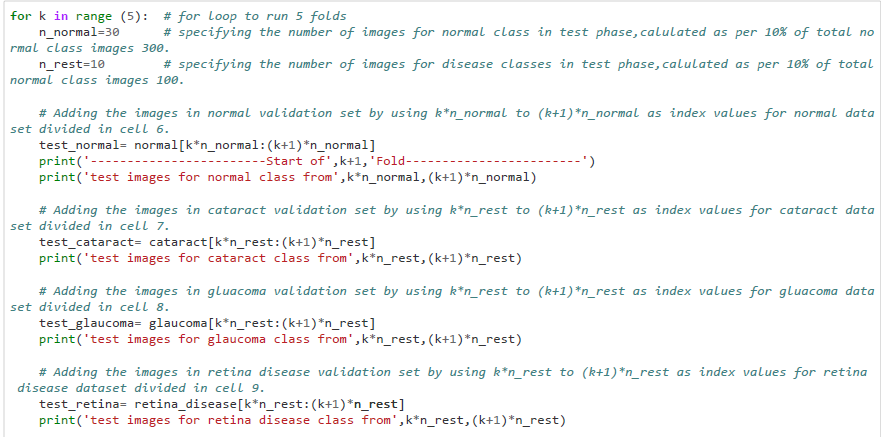
**Figure 14: DenseNet169-LSTM Model Implementation**

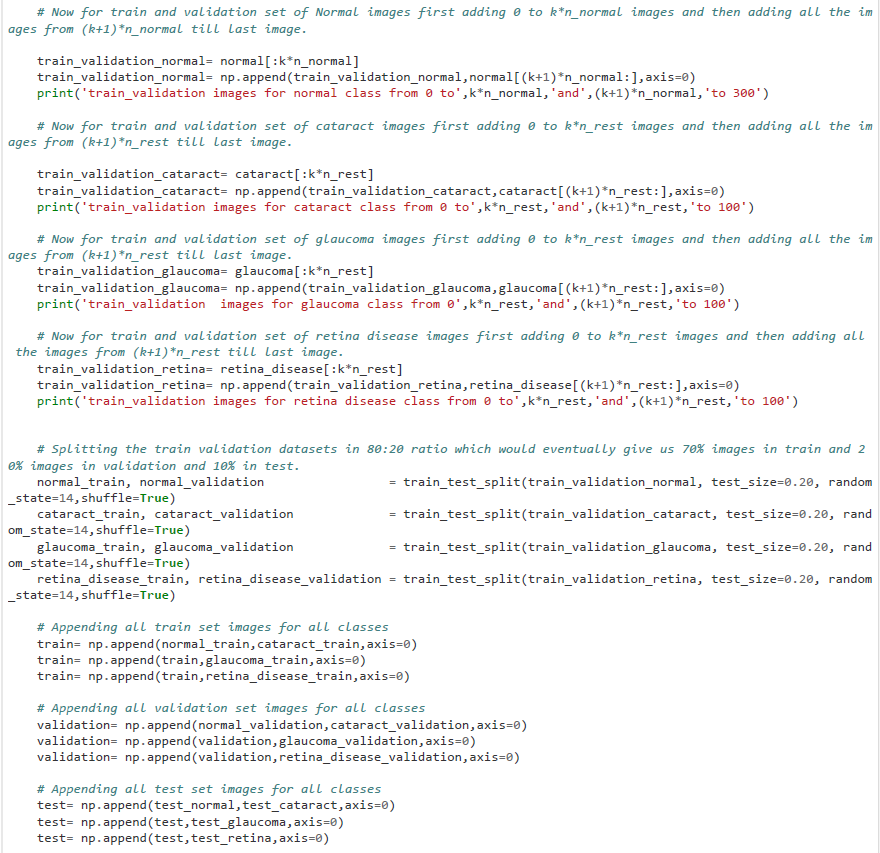
# K-Fold Cross Validation Implementation

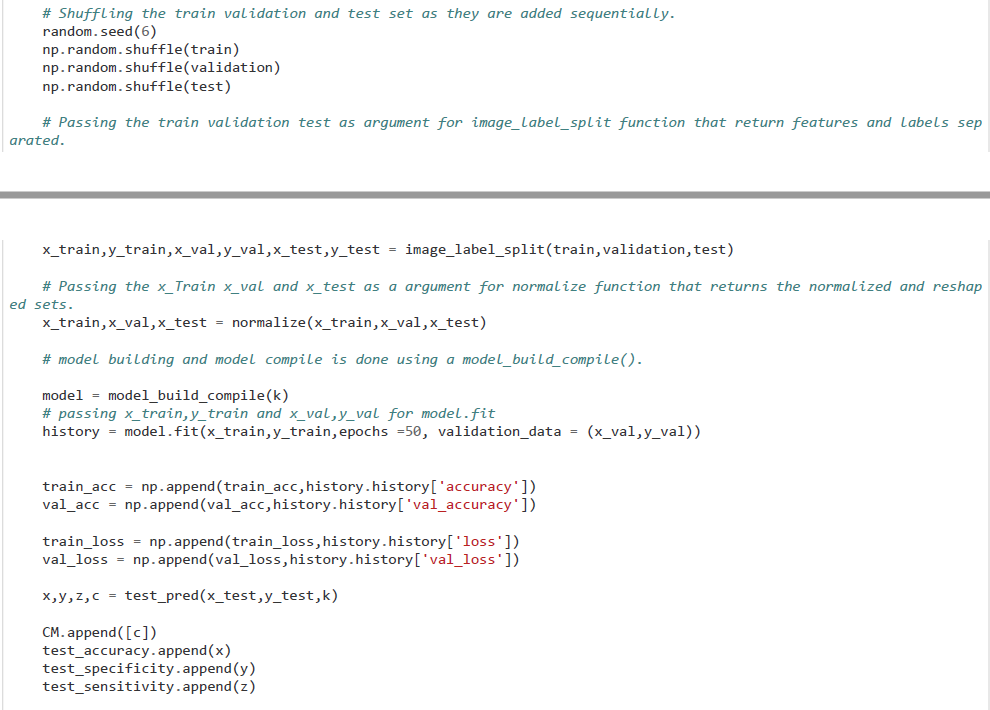
The steps taken till now are necessary to start the model compilation and model fitting stage, an additional step required which is not explained till now is explained in model evaluation section that is test\_pred() which is explained at section 9.1. This function predicts the test labels and after that test confusion matrix, sensitivity and specificity is derived. Figure 15,16 represents the K fold validation implementation for K=5. Figure 15, contains the empty lists created to store the model derived training and validation phase accuracy and loss along with test phase predicted confusion matrices, accuracy, sensitivity and specificity.



**Figure 15: Variables to Store Model Generate Result**







**Figure 16: K-Fold Validation Implementation**

Figure 16, includes the function call that passes k as an argument to model\_build\_compile() function that returns the new model built and compiled for each fold. After that, the Model is fitted using the training and validation datasets.

## Train Test Validation images per label check

The below code block can be used to confirm the number of images in the test phase of the model. To check for train and validation need to replace test in the for loop with train and validation.



**Figure 17: Number of images check in Train Validation and Test**

# Model Evaluation and Visualizations

## Test Predictions, Accuracy, Sensitivity and Specificity

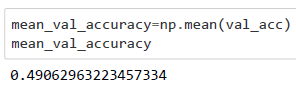
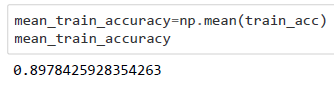
The function defined in this section needs to be executed before section 8.

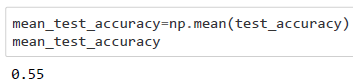


**Figure 18: Test Predictions and Confusion Matrix and Sensitivity and Specificity Computation**

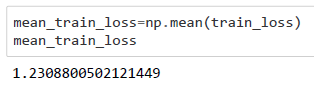
## Average of All the Evaluation Metrics for Train, Validation and Testing Phase

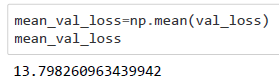
As the K fold validation generates 5 different results set for each fold, the average of all the evaluation matrices needs to be computed.



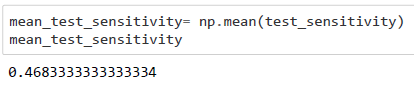
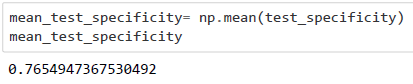


**Figure 19: Average Accuracy Computation for All Phases**



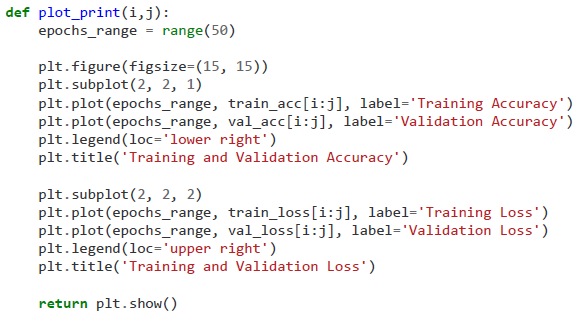


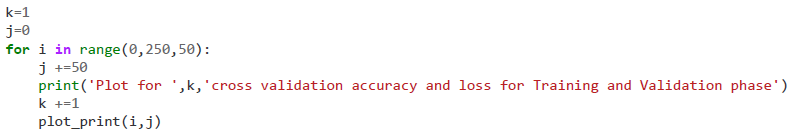
**Figure 20: Average Loss for Training and Validation**

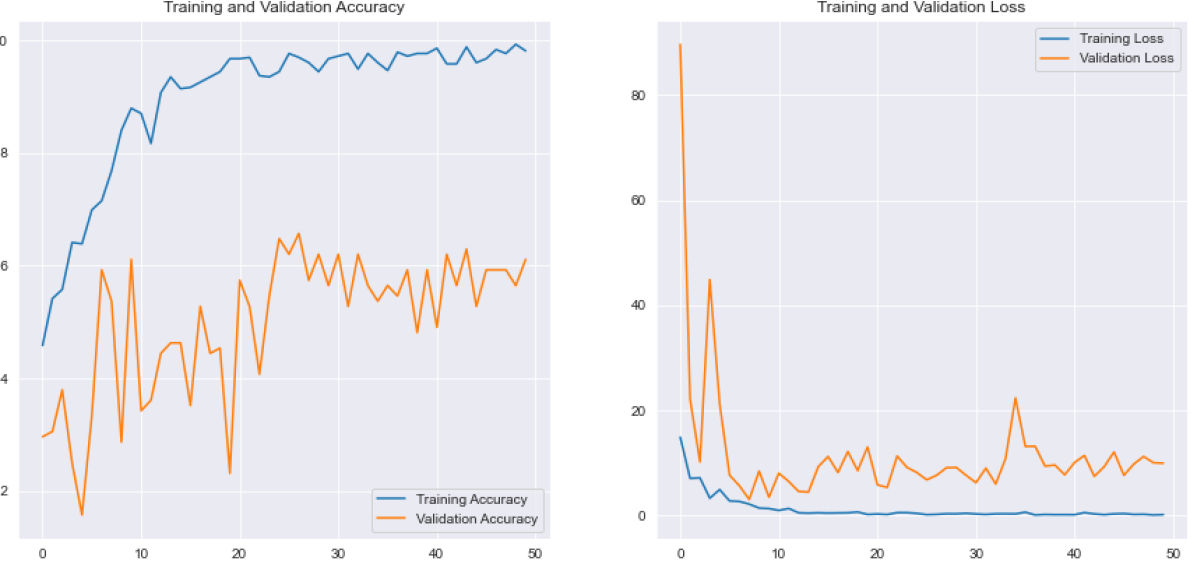


**Figure 21: Average Sensitivity and Specificity for Test**

## Training v/s Validation Accuracy and Loss Plots

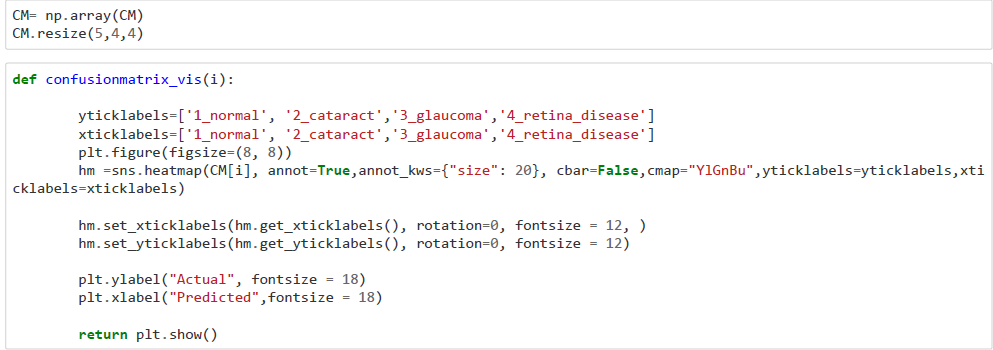


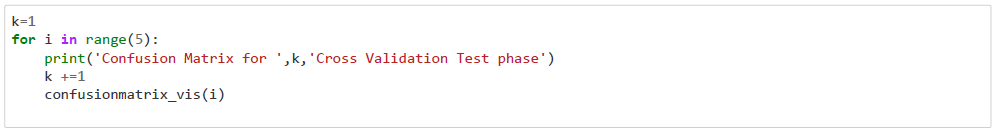




**Figure 22: Training v/s Validation Plots**

## Test Phase Confusion Matrices Visualization





**Figure 23: Confusion Matrices Visualization**

## Summarizing the Confusion Matrix for All Folds



**Figure 23: Summarized Confusion Matrix**

## Reconfirming the Test Evaluation Results



**Figure 24: Test Results computed using Overall Confusion Matrix**

# References

Data Source:

<https://www.kaggle.com/jr2ngb/cataractdataset>

Code Reference for Confusion Matrix, Accuracy, Sensitivity and Specificity Computation:

<https://statinfer.com/204-4-2-calculating-sensitivity-and-specificity-in-python/>

<https://towardsdatascience.com/visual-guide-to-the-confusion-matrix-bb63730c8eba>