Glaucoma Classification

Chaturong Tantibundhit, Ph.D. Adviser

Member

กุลนิษฐ์ ฉัตราฐิติพันธ์	6110612980
เจษฎากร แนวบรรทัด	6110613061
ประภานิช อมรวิสิทธิ์กุล	6110613244
ณัฐภัทร เลี่ยวไพโรจน์	6110613269

Problem Statement

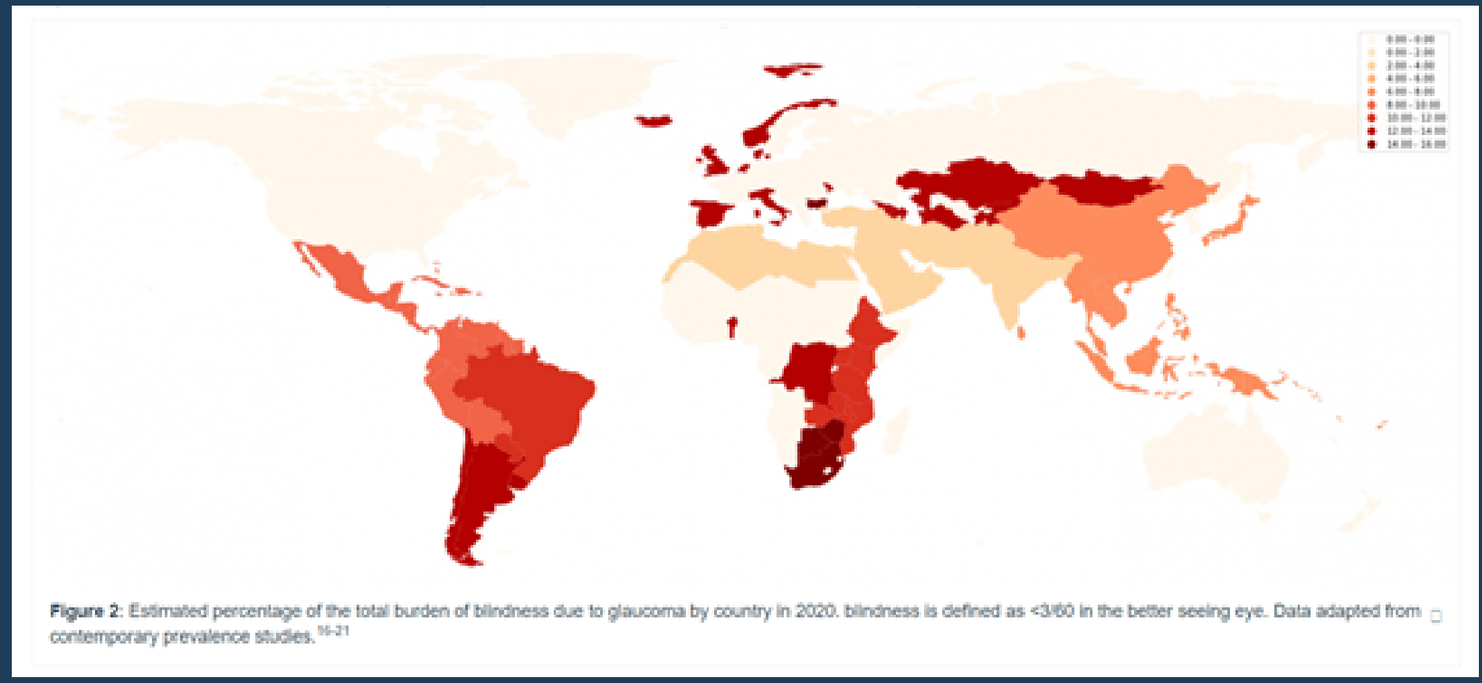


Figure 2: Estimated percentage of the total burden of blindness due to glaucoma by country in 2020. blindness is defined as <3/60 in the better seeing eye. Data adapted from contemporary prevalence studies.[16-21]

ต้อหินเป็นโรคที่พบได้บ่อยในผู้สูงอายุและถือเป็นโรคที่ทำให้เกิดภาวะสูญเสียการมองเห็นชนิดถาวรที่พบได้บ่อยที่สุดทั่วโลกเป็นอันดับ 2 รองจากโรคต้อกระจก โดยในปี ค.ศ. 2020 มีผู้เป็นโรคต้อหินถึง 76 ล้านคน

OBJECTIVES

เพื่อทำการสร้าง AI ที่ใช้ในการตรวจคัดกรองโรค glaucoma, ตาปกติ และ โรคอื่นๆด้วย Machine learning และ deep learning

Machine Learning

Data Collection

ประเภท	จำนวนข้อมูล
Glaucoma	1,363
Normal	2,009
Others	896
Total	4,268

EXPERIMENTAL SETUP

1) **คัดรูป และ ทำการสกัด** Features
2) **ทำการเก็บ** Dataset **จาก** Features **ที่สกัดได้**
3) **นำ** Dataset **ไปทำการ** Train Model **ด้วย** Quadratic Discriminant Analysis Algorithm **และ** RBF SVM Algorithm
4) **ทดสอบ** Model **ที่ได้ ด้วย** Test Data

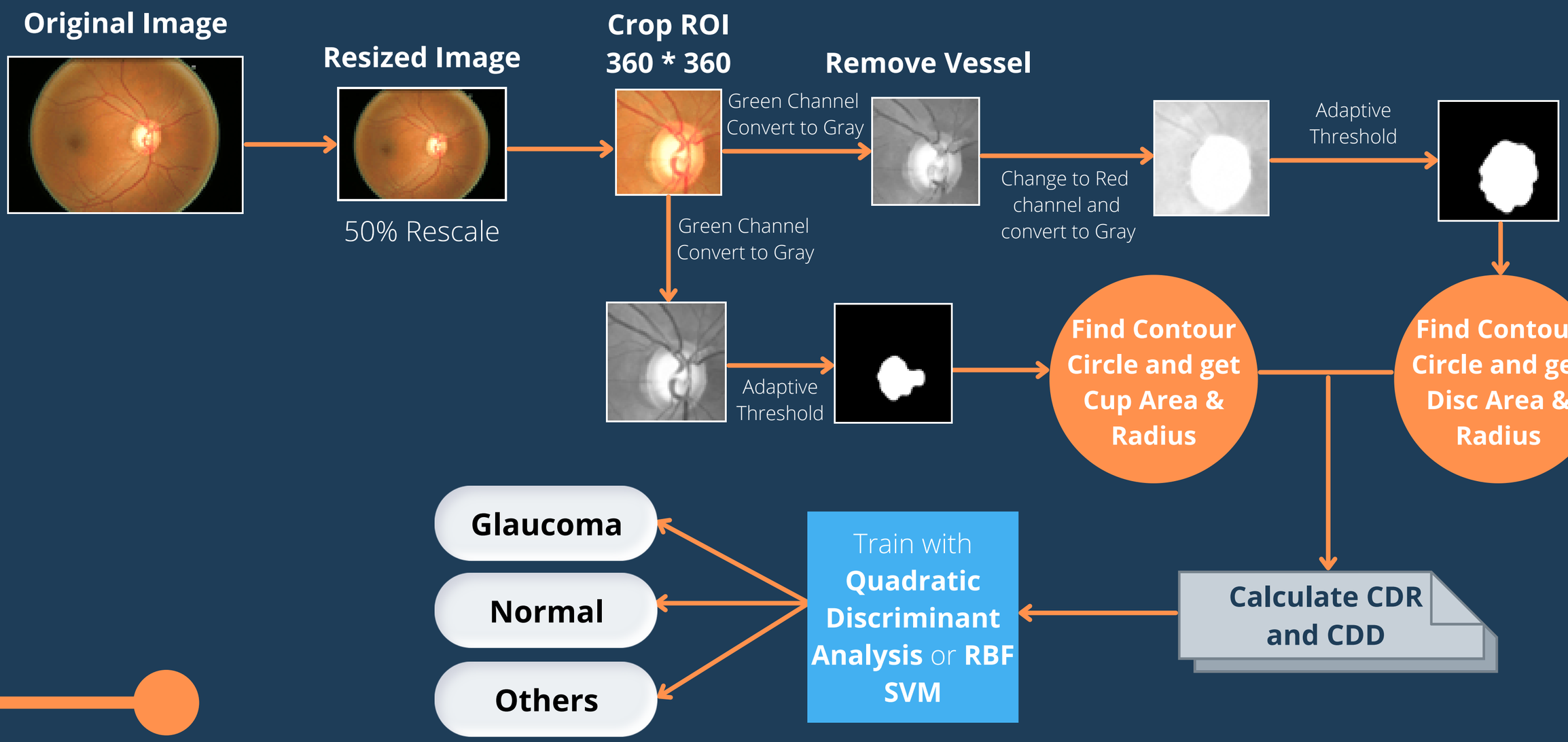
EXPERIMENTAL SETUP

5-Fold Cross Validation

	Data Train					Test Data
Model 1	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test Data
Model 2	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test Data
Model 3	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test Data
Model 4	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test Data
Model 5	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test Data

Data Validation Data Train Data Test

Machine Learning Methods



Ashish Issac, M. Parthasarthi, Malay Kishore Dutta, Flowchart for the segmentation of optic disc and cup, [Online]. Available: https://ieeexplore.ieee.org/document/7095384. (access Mar. 24th 2021)

ARCHITECTURE

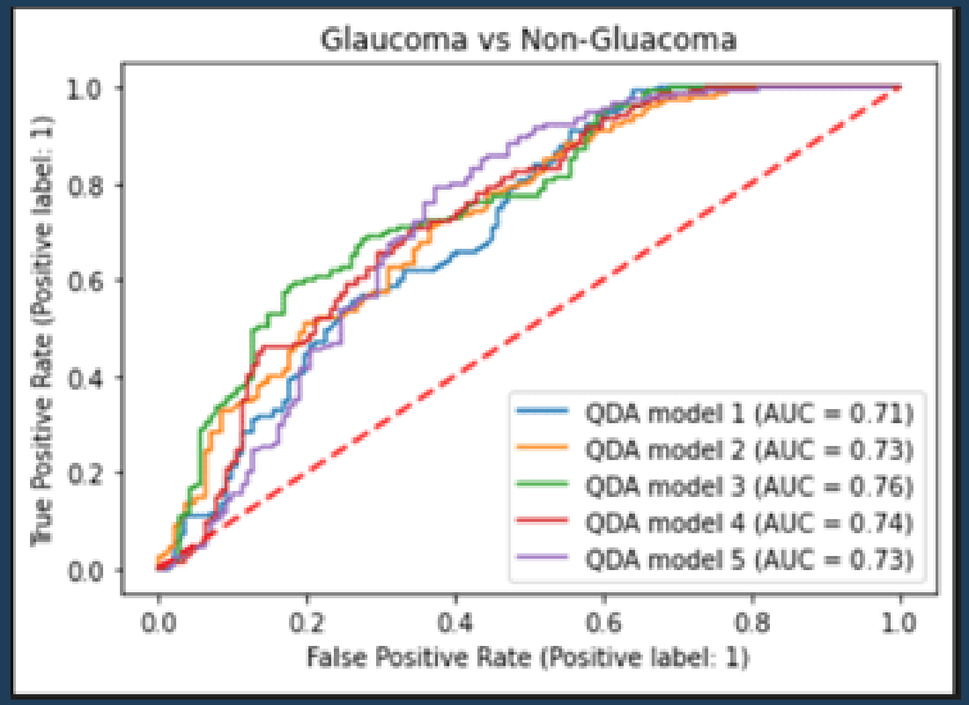
Quadratic Discriminant Analysis

RBF SVM

Experiment Result

Glaucoma vs Non- Glaucoma

Quadratic Discriminant Analysis (Model 3)



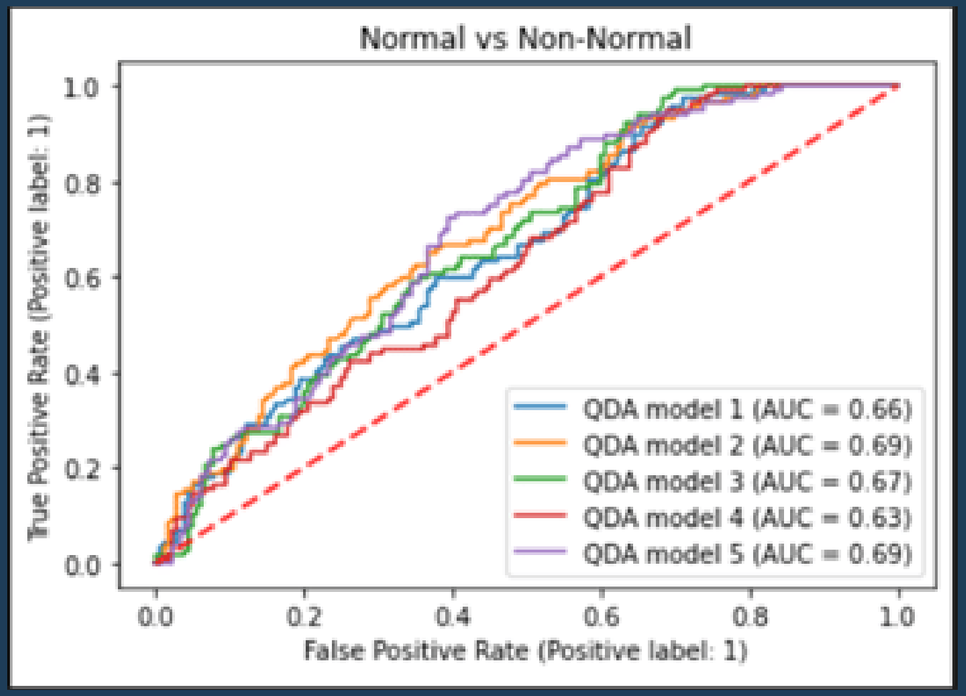
ประเภทจริง (True class)	ประเภทจริง		รวม (Total)
	บวก(Positive)	**ลบ**(Negative)	
บวก(Positive)	149(80.11%)	37(19.89%)	186
ลบ(Negative)	78(42.16%)	107(57.84%)	185
รวม(Total)	227	144	371

Accuracy	Specificity	Sensitivity	Precision	F1 Score
0.690027	0.578378	0.801075	0.656388	0.690027

Experiment Result

Quadratic Discriminant Analysis (Model 2)

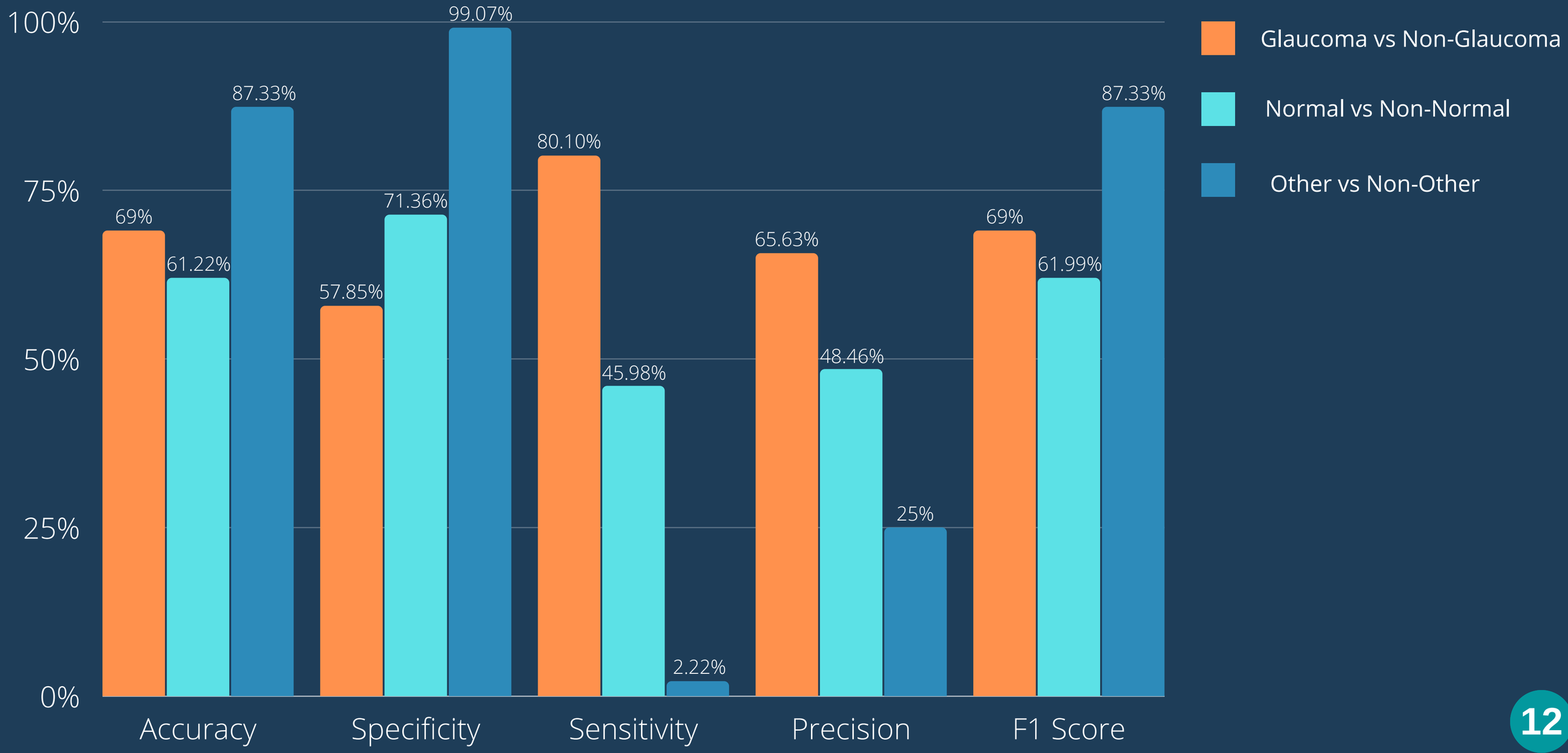
Normal vs Non- Normal



ประเภทจริง (True class)	ประเภทจริง		รวม (Total)
	บวก(Positive)	ลบ(Negative)	
บวก(Positive)	63(45.98%)	74(54.02%)	137
ลบ(Negative)	67(28.63%)	167(71.37%)	234
รวม(Total)	130	241	371

Accuracy	Specificity	Sensitivity	Precision	F1 Score
0.619946	0.713675	0.459854	0.484615	0.619946

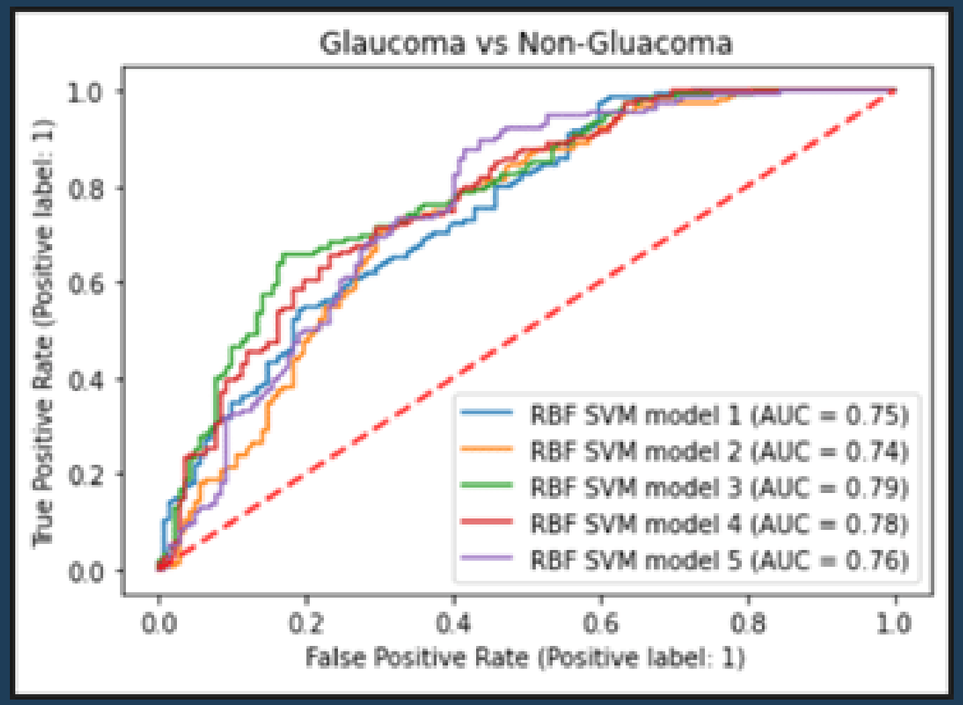
Quadratic Discriminant Analysis



Experiment Result

RBF SVM (Model 3)

Glaucoma vs Non- Glaucoma



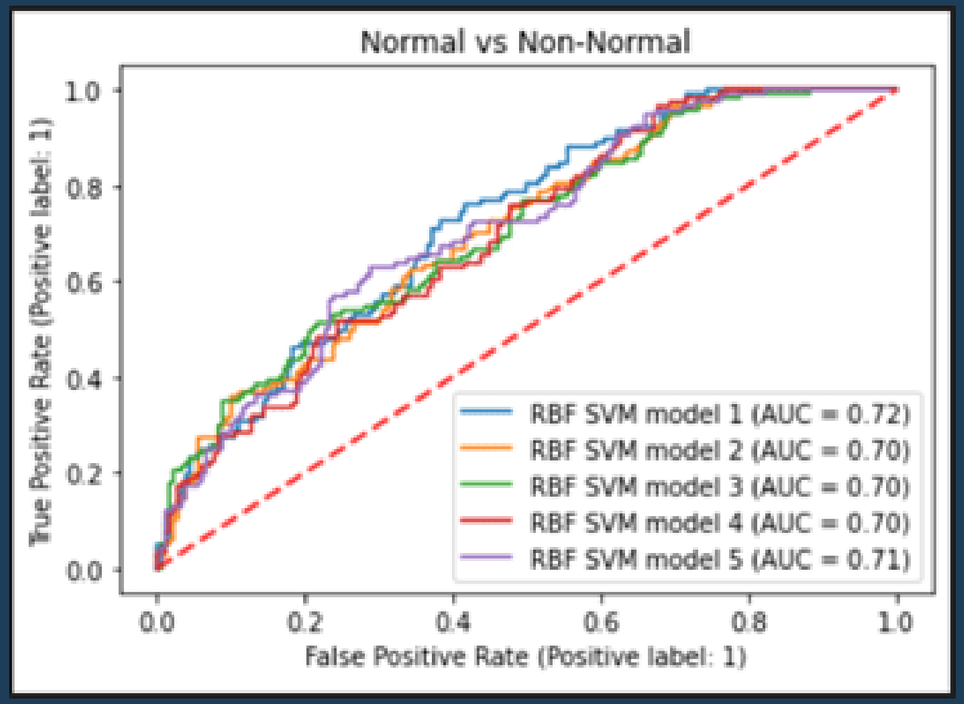
ประเภทจริง (True class)	ประเภทจริง		รวม (Total)
	บวก(Positive)	ลบ(Negative)	
บวก(Positive)	161(86.56%)	25(13.44%)	186
ลบ(Negative)	80(43.24%)	105(56.76%)	185
รวม(Total)	241	130	371

Accuracy	Specificity	Sensitivity	Precision	F1 Score
0.716981	0.567567	0.865591	0.668050	0.716981

Experiment Result

RBF SVM (Model 1)

Normal vs Non- Normal



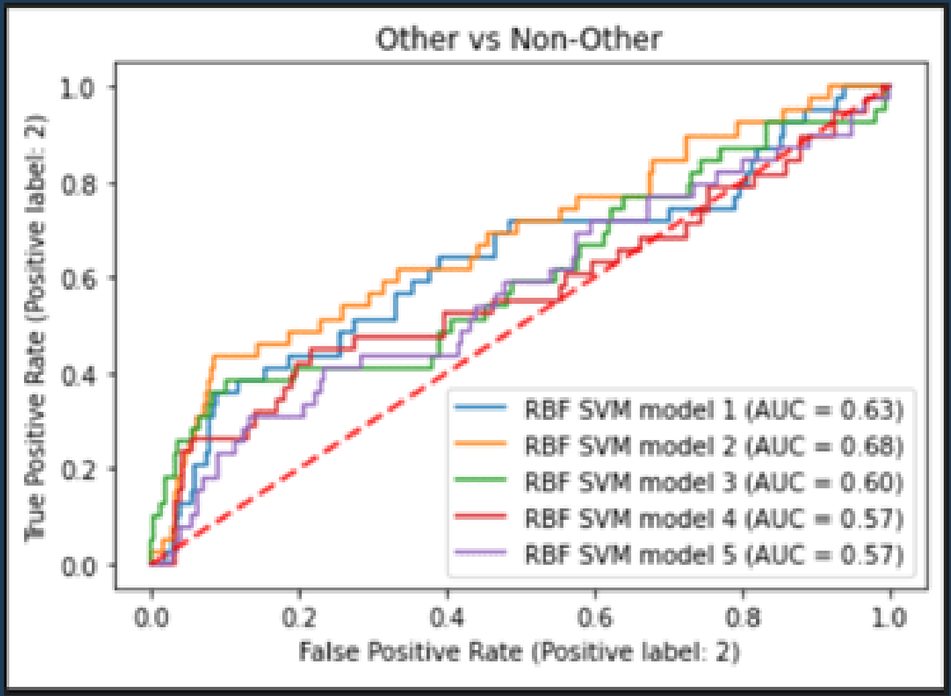
ประเภทจริง (True class)	ประเภทจริง		รวม (Total)
	บวก(Positive)	ลบ(Negative)	
บวก(Positive)	50(36.50%)	87(63.50%)	137
ลบ(Negative)	41(17.52%)	193(82.48%)	234
รวม(Total)	91	280	371

Accuracy	Specificity	Sensitivity	Precision	F1 Score
0.649596	0.829060	0.343066	0.540230	0.617251

Experiment Result

RBF SVM (Model 5)

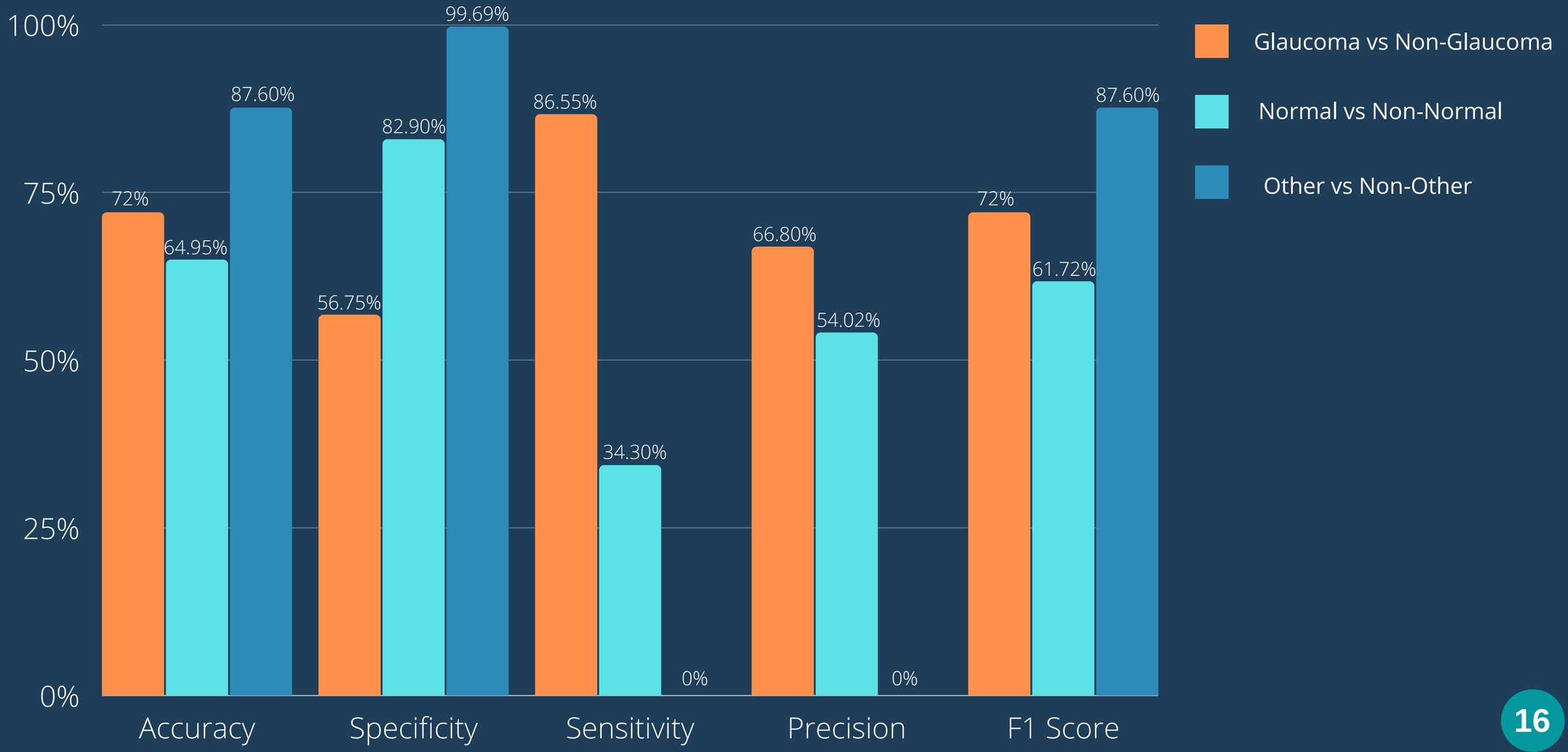
Other vs Non- Other



ประเภทจริง (True class)	ประเภทจริง		รวม (Total)
	บวก(Positive)	ลบ(Negative)	
บวก(Positive)	0(0%)	45(100%)	45
ลบ(Negative)	1(0.30%)	325(99.70%)	326
รวม(Total)	1	370	371

Accuracy	Specificity	Sensitivity	Precision	F1 Score
0.876010	0.996932	0	0	0.876010

RBF SVM


100%
75%
50%
25%
0%
72%
64.95%
87.60%
56.75%
82.90%
99.69%
86.55%
34.30%
0%
66.80%
54.02%
0%
72%
61.72%
87.60%
Accuracy
Specificity
Sensitivity
Precision
F1 Score
Glaucoma vs Non-Glaucoma
Normal vs Non-Normal
Other vs Non-Other

CODE
EXPLANATION

```
def delVessel(image):
    blue,green,red = cv.split(image)
    kernel = cv.getStructuringElement(cv.MORPH_ELLIPSE,(26,26))
    ves = cv.morphologyEx(green, cv.MORPH_BLACKHAT, kernel)

    vessel2 = cv.bitwise_or(ves,green)
    vessel = cv.bitwise_or(ves,red)
    vessel = cv.medianBlur(red,7)
    plt.imshow(vessel2)

    return vessel
```

CODE
EXPLANATION

```python
def GetDisc(image):
    M = 60    #filter size

    filter = signal.gaussian(M, std=7) #Gaussian Window
    filter=filter/sum(filter)
    STDf = filter.std()  #It'standard deviation

    image_pre = image-image.mean()-image.std()

    thr = (0.5 * M) - (2*STDf) - image_pre.std()

    r,c = image.shape
    Dd = np.zeros(shape=(r,c))

    for i in range(1,r):
        for j in range(1,c):
            if image_pre[i,j]>thr:
                Dd[i,j]=255
            else:
                Dd[i,j]=0

    Dd = cv.morphologyEx(Dd, cv.MORPH_CLOSE, cv.getStructuringElement(cv.MORPH_ELLIPSE,(2,2)), iterations = 1)
    Dd = cv.morphologyEx(Dd, cv.MORPH_OPEN, cv.getStructuringElement(cv.MORPH_ELLIPSE,(7,7)), iterations = 1)
    Dd = cv.morphologyEx(Dd, cv.MORPH_CLOSE, cv.getStructuringElement(cv.MORPH_ELLIPSE,(1,21)), iterations = 1)
    Dd = cv.morphologyEx(Dd, cv.MORPH_OPEN, cv.getStructuringElement(cv.MORPH_ELLIPSE,(21,1)), iterations = 1)
    Dd = cv.morphologyEx(Dd, cv.MORPH_CLOSE, cv.getStructuringElement(cv.MORPH_ELLIPSE,(23,23)), iterations = 1)
    Dd = cv.morphologyEx(Dd, cv.MORPH_OPEN, cv.getStructuringElement(cv.MORPH_ELLIPSE,(43,43)), iterations = 1)

    Dd = np.uint8(Dd)

    return Dd
```

CODE
EXPLANATION

```
def GetCup(image):
    blue, green, red = cv.split(image)
    green = cv.medianBlur(green,7)

    M = 60    #filter size

    filter = signal.gaussian(M, std=7) #Gaussian Window
    filter = filter/sum(filter)
    STDf = filter.std()  #It'standard deviation

    green_pre = green-green.mean()-green.std()

    thr = (0.5 * M) + (2 * STDf) + (green_pre.std()) + (green_pre.mean())
    r,c = green.shape
    Dc = np.zeros(green.shape[:2])

    for i in range(1,r):
        for j in range(1,c):
            if green_pre[i,j]>thr:
                Dc[i,j]=255
            else:
                Dc[i,j]=0

    Dc = cv.morphologyEx(Dc, cv.MORPH_CLOSE, cv.getStructuringElement(cv.MORPH_ELLIPSE,(2,2)), iterations = 1)
    Dc = cv.morphologyEx(Dc, cv.MORPH_OPEN, cv.getStructuringElement(cv.MORPH_ELLIPSE,(7,7)), iterations = 1)
    Dc = cv.morphologyEx(Dc, cv.MORPH_CLOSE, cv.getStructuringElement(cv.MORPH_ELLIPSE,(1,21)), iterations = 1)
    Dc = cv.morphologyEx(Dc, cv.MORPH_OPEN, cv.getStructuringElement(cv.MORPH_ELLIPSE,(21,1)), iterations = 1)
    Dc = cv.morphologyEx(Dc, cv.MORPH_CLOSE, cv.getStructuringElement(cv.MORPH_ELLIPSE,(33,33)), iterations = 1)
    Dc = cv.morphologyEx(Dc, cv.MORPH_OPEN, cv.getStructuringElement(cv.MORPH_ELLIPSE,(33,33)), iterations = 1)

    Dc = np.uint8(Dc)
    return Dc
```

CODE
EXPLANATION

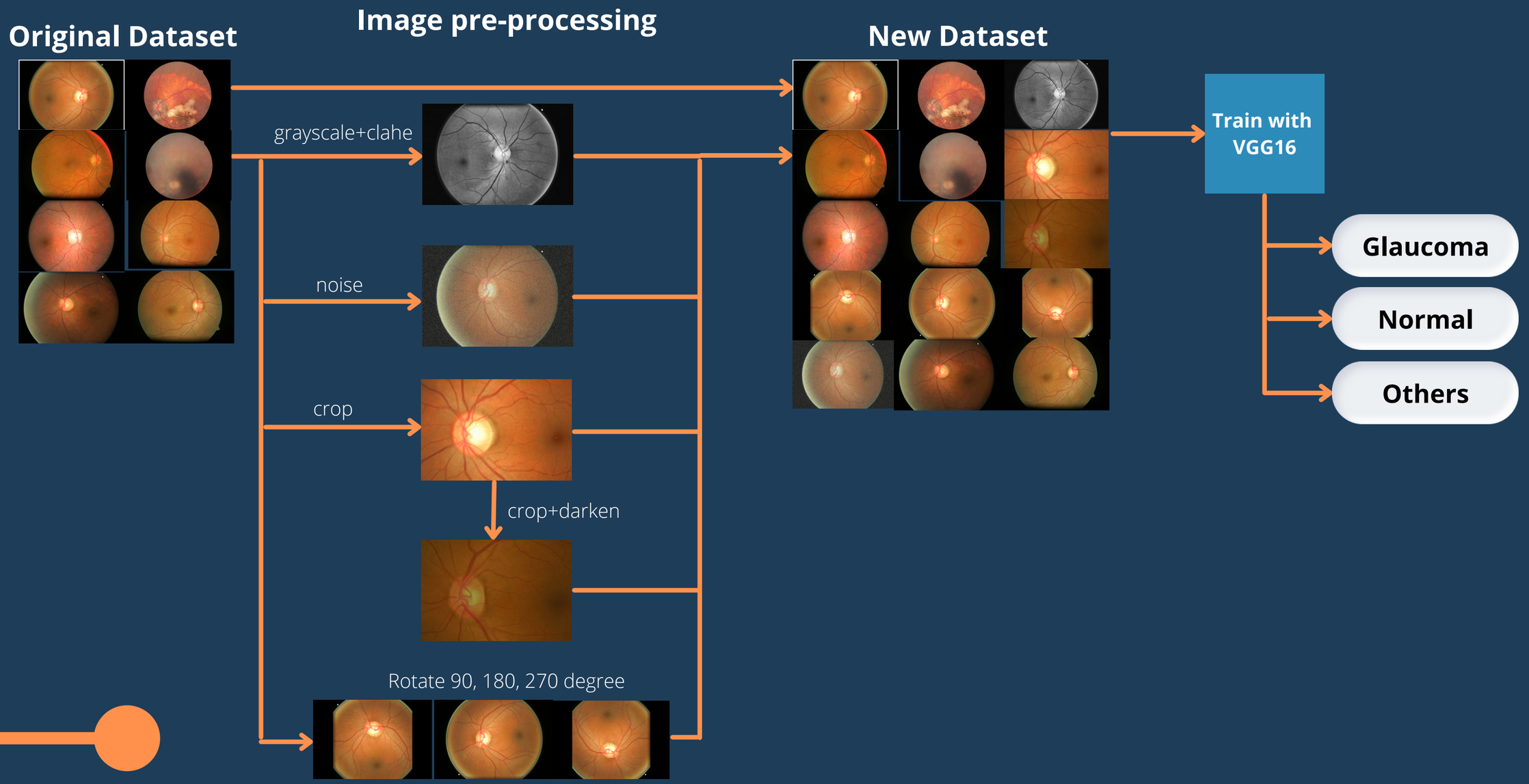
```
trainData01,testData01,trainType01,testType01 = train_test_split(data01, eye01, test_size=0.2 , random_state=1)
skf = StratifiedKFold(n_splits=5)

plt.scatter(data01[:,0],data01[:,1], c=eye01)
plt.xlabel('CDR')
plt.ylabel('y')


# QDA model for Glucoma vs Non-Glaucoma
clf = QuadraticDiscriminantAnalysis()
tprs = []
aucs = []
mean_fpr = np.linspace(0, 1, 100)
fig, ax = plt.subplots()
i = 1

for train_index, test_index in skf.split(trainData01, trainType01):
    x_train, x_test = trainData01[train_index], trainData01[test_index]
    y_train, y_test = trainType01[train_index], trainType01[test_index]
    clf.fit(x_train,y_train)
    filename = 'savemodel/GOQDA_model'+str(i)+'.sav'
    joblib.dump(clf,filename)

    viz = plot_roc_curve(clf,x_test, y_test,name='QDA model {}'.format(i), ax=ax)
    interp_tpr = np.interp(mean_fpr, viz.fpr, viz.tpr)
    interp_tpr[0] = 0.0
    tprs.append(interp_tpr)
    aucs.append(viz.roc_auc)

    i+=1
```

Deep Learning

Data Collection

ประเภท	จำนวนข้อมูล
Glaucoma	12,516
Normal	9,050
Others	5,049
Total	26,615

Deep Learning Methods



ARCHITECTURE

Model 1

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 224, 224, 64)      1792
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 222, 222, 64)      36928
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 111, 111, 64)      0
_________________________________________________________________
dropout (Dropout)            (None, 111, 111, 64)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 111, 111, 32)      18464
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 109, 109, 32)      9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 54, 54, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 54, 54, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 54, 54, 16)        4624
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 52, 52, 16)        2320
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 26, 26, 16)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 26, 26, 16)        0
_________________________________________________________________
flatten (Flatten)            (None, 10816)             0
_________________________________________________________________
dense (Dense)                (None, 64)                692288
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 32)                2080
_________________________________________________________________
dropout_4 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_2 (Dense)              (None, 16)                528
_________________________________________________________________
dropout_5 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_3 (Dense)              (None, 3)                 51
=================================================================
Total params: 768,323
Trainable params: 768,323
Non-trainable params: 0
```

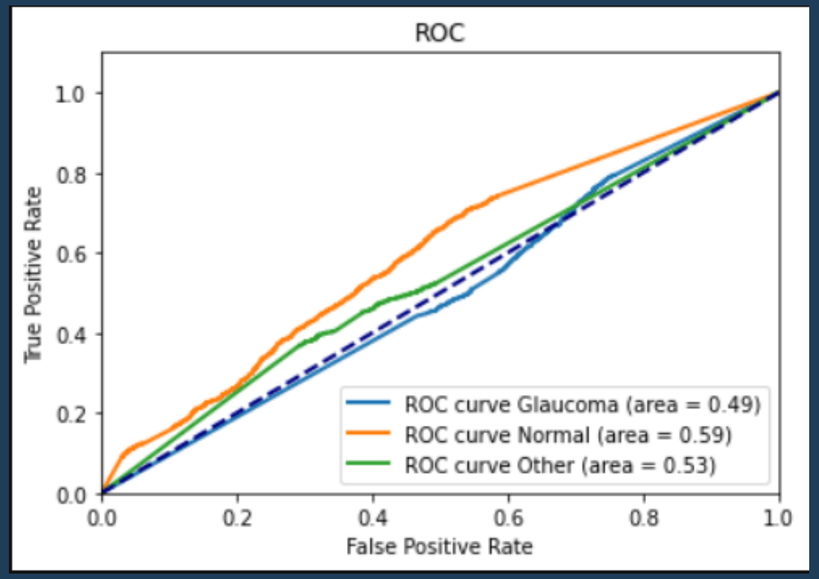
Model 2

VGG16

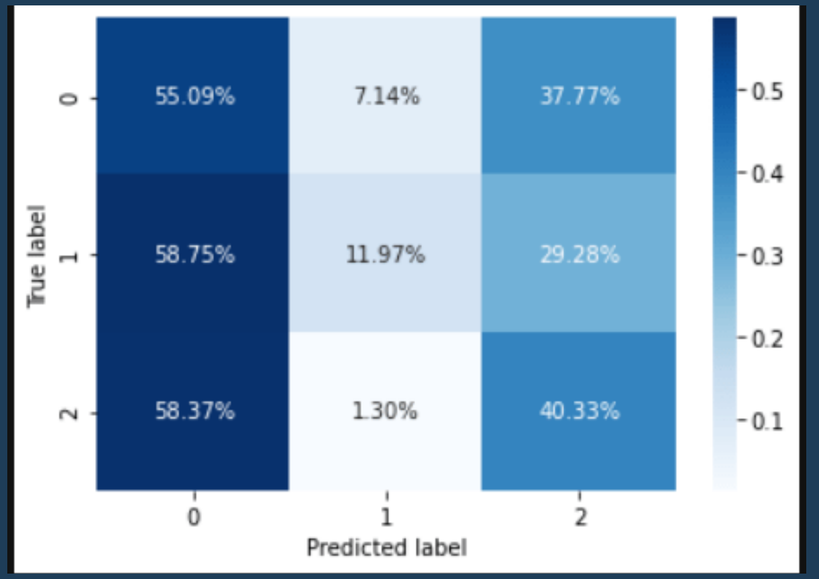
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 7, 7, 512)         14714688
_________________________________________________________________
global_average_pooling2d (Gl (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 3)                 1539
=================================================================
Total params: 14,716,227
Trainable params: 1,539
Non-trainable params: 14,714,688
_________________________________________________________________
```

Model 1

ROC



confusion table

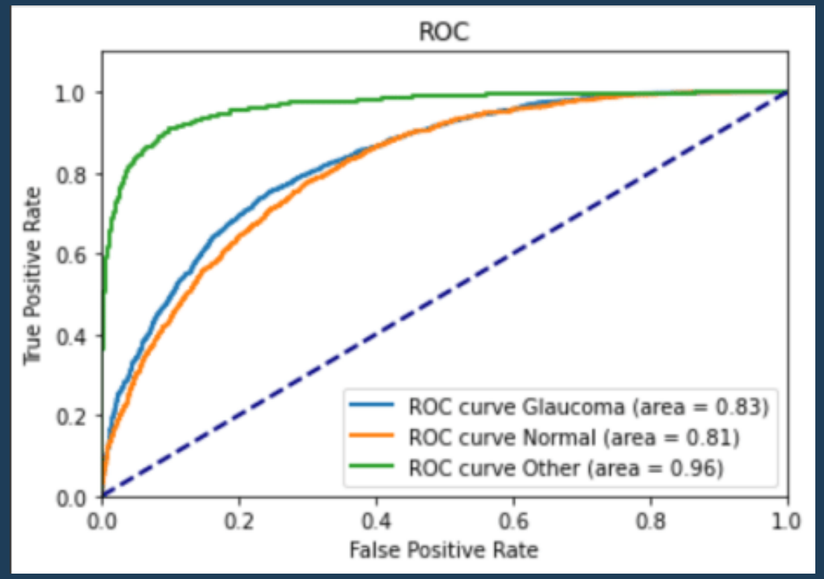


Type	Precision	Recall	F1 Score
Glaucoma	0.45483	0.55090	0.49828
Normal	0.53061	0.11974	0.19538
Other	0.21632	0.40330	0.28160

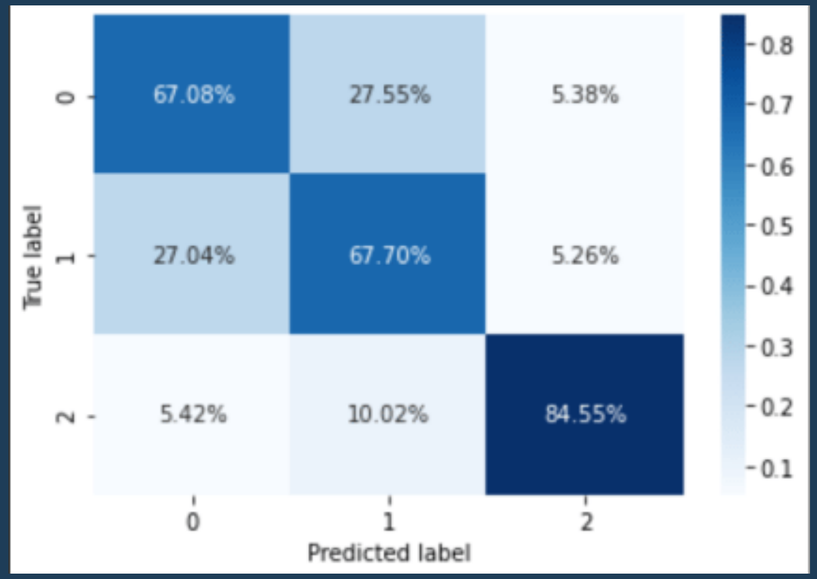
Accuracy = 0.37629

Model 2

ROC



confusion table



Type	Precision	Recall	F1 Score
Glaucoma	0.75522	0.67079	0.71051
Normal	0.60780	0.67697	0.64052
Other	0.78791	0.84552	0.81570

Accuracy = 0.70604

CODE
EXPLANATION

```
import tensorflow as tf
import numpy as np
from tensorflow.keras.preprocessing import image
from sklearn.metrics import accuracy_score, f1_score, precision_score, confusion_matrix, roc_curve, auc, plot_roc_curve
```

```
path_train = "dataset/train"

img_train = tf.keras.preprocessing.image_dataset_from_directory(
    path_train,
    validation_split=0.2,
    subset = "training",
    seed = 125,
    image_size = (224,224),
    batch_size = 32
)

img_validation = tf.keras.preprocessing.image_dataset_from_directory(
    path_train,
    validation_split=0.2,
    subset = "validation",
    seed = 125,
    image_size = (224,224),
    batch_size = 32
)
```

CODE
EXPLANATION

```
VGG16_MODEL=tf.keras.applications.VGG16(input_shape=(224,224,3),
                                        include_top=False,
                                        weights='imagenet')
VGG16_MODEL.trainable=False
global_average_layer = tf.keras.layers.GlobalAveragePooling2D()
prediction_layer = tf.keras.layers.Dense(3,activation='softmax')
```

```
model = tf.keras.Sequential([
  VGG16_MODEL,
  global_average_layer,
  prediction_layer
])

model.compile(optimizer="adam",
              loss=tf.keras.losses.sparse_categorical_crossentropy,
              metrics=["accuracy"])
```

```
model.summary()
```

CODE
EXPLANATION

```
history = model.fit(img_train,
                    epochs=18,
                    steps_per_epoch=30,
                    validation_steps=2,
                    validation_data=img_validation)
model.save("deeplearningVGG16.h5")
```

```
#import Test img

path_test = "dataset/test"

img_test = tf.keras.preprocessing.image_dataset_from_directory(
    path_test,
    image_size = (224,224)
)


label = ["glaucoma","normal","other"]

loadmodel = tf.keras.models.load_model("deeplearningVGG16.h5")

predict = loadmodel.predict(img_test)
prediction = np.argmax(predict)
```

CODE
EXPLANATION

```
i = 0
glau = 0
norm = 0
oth = 0
for i in predict:
    index = np.argmax(i)
    if index == 0:
        glau+=1
    elif index == 1:
        norm+=1
    elif index==2 :
        oth+=1

print(glau)
print(norm)
print(oth)
```

```
# change file type to jpeg

import imghdr
import cv2
import os
import glob


for file in glob.glob('dataset/test/other/*.jpg'):
    image = cv2.imread(file)
    file_type = imghdr.what(file)
    if file_type != 'jpeg':
        print(file + " - invalid - " + str(file_type))
        cv2.imwrite(file, image)
    print("finish jpg",image)
```

PROBLEM

GPU Error.

Some images can't crop.

The feature can't extract other images.

Reference

Machine Learning

[1] A. Issac, M. Parthasarthi and M. K. Dutta, "An adaptive threshold based algorithm for optic disc and cup segmentation in fundus images," 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN), Noida, India, 2015, pp. 143-147, doi: 10.1109/SPIN.2015.7095384.
(access Mar. 23th 2021)

[2] Ahmed Almazroa, Ritambhar Burman, Kaamran Raahemifar, Vasudevan Lakshminarayanan, "Optic Disc and Optic Cup Segmentation Methodologies for Glaucoma Image Detection: A Survey", Journal of Ophthalmology, vol. 2015, Article ID 180972, 28 pages, 2015. https://doi.org/10.1155/2015/180972
(access Mar. 23th 2021)

[3] Atheesan S. and Yashothara S., "Automatic glaucoma detection by using funduscopic images," 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai, India, 2016, pp. 813-817, doi: 10.1109/WiSPNET.2016.7566246.
(access Mar. 24th 2021)

[4]Ashish Issac, M. Parthasarthi, Malay Kishore Dutta, Flowchart for the segmentation of optic disc and cup, [Online]. Available: https://ieeexplore.ieee.org/document/7095384. (access Mar. 24th 2021)

Reference

Deep Learning

[1] Lorenzo Baraldi, "VGG-16 pre-trained model for Keras", 2015. [Online]. Available: https://gist.github.com/baraldilorenzo/07d7802847aaad0a35d3 (access May. 7th 2021)

[2] Siladittya Manna, "K-Fold Cross Validation for Deep Learning Models using Keras", Mar. 2020. [Online]. Available: https://medium.com/the-owl/k-fold-cross-validation-in-keras-3ec4a3a00538 (access May. 7th 2021)

[3] Machinecurve, "How to predict new samples with your TensorFlow / Keras model?", Feb. 2020. [Online]. Available: https://www.machinecurve.com/index.php/2020/02/21/how-to-predict-new-samples-with-your-keras-model (access May. 7th 2021)

[4] Patrick Kalkman, "Increase the Accuracy of Your CNN by Following These 5 Tips I Learned From the Kaggle Community", Feb. 2021. [Online]. Available: https://towardsdatascience.com/increase-the-accuracy-of-your-cnn-by-following-these-5-tips-i-learned-from-the-kaggle-community-27227ad39554 (access May. 8th 2021)

Reference

Deep Learning

[5] Brijesh Thumar, "How to use VGG model in TensorFlow Keras", May. 2019. [Online]. Available: https://androidkt.com/how-to-use-vgg-model-in-tensorflow-keras/ (access May. 9th 2021)

[6] JIACHENG DANG, "Detecting Open-angle Glaucoma Using a Two-parts Deep Learning Architecture", Mar. 2020. [Online]. Available: https://ysjournal.com/detecting-open-angle-glaucoma-using-a-two-parts-deep-learning-architecture/ (access May. 9th 2021)

[7]Ahn JM, Kim S, Ahn KS, Cho SH, Lee KB, et al. (2019) Correction: A deep learning model for the detection of both advanced and early glaucoma using fundus photography. PLOS ONE 14(1): e0211579. https://doi.org/10.1371/journal.pone.0211579 (access May. 9th 2021)