```
!pip3 install autokeras
```

```
Collecting autokeras
  Downloading autokeras-1.0.16.post1-py3-none-any.whl (166 kB)
     |████████████████████████████████| 166 kB 18.2 MB/s
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages (f
Collecting keras-tuner<1.1,>=1.0.2
  Downloading keras_tuner-1.0.4-py3-none-any.whl (97 kB)
     |████████████████████████████████| 97 kB 5.6 MB/s
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.7/dist-packa
Collecting tensorflow<2.6,>=2.3.0
  Downloading tensorflow-2.5.2-cp37-cp37m-manylinux2010_x86_64.whl (454.4 MB)
     |████████████████████████████████| 454.4 MB 22 kB/s
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: tensorboard in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: ipython in /usr/local/lib/python3.7/dist-packages (
Collecting kt-legacy
  Downloading kt_legacy-1.0.4-py3-none-any.whl (9.6 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (fr
Collecting grpcio~=1.34.0
  Downloading grpcio-1.34.1-cp37-cp37m-manylinux2014_x86_64.whl (4.0 MB)
     |████████████████████████████████| 4.0 MB 27.8 MB/s
Requirement already satisfied: keras-preprocessing~=1.1.2 in /usr/local/lib/python
Requirement already satisfied: h5py~=3.1.0 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: termcolor~=1.1.0 in /usr/local/lib/python3.7/dist-p
Collecting tensorflow-estimator<2.6.0,>=2.5.0
  Downloading tensorflow_estimator-2.5.0-py2.py3-none-any.whl (462 kB)
     |████████████████████████████████| 462 kB 48.2 MB/s
Collecting typing-extensions~=3.7.4
  Downloading typing_extensions-3.7.4.3-py3-none-any.whl (22 kB)
Collecting keras-nightly~=2.5.0.dev
  Downloading keras_nightly-2.5.0.dev2021032900-py2.py3-none-any.whl (1.2 MB)
     |████████████████████████████████| 1.2 MB 51.3 MB/s
Requirement already satisfied: opt-einsum~=3.3.0 in /usr/local/lib/python3.7/dist-
Collecting flatbuffers~=1.12.0
  Downloading flatbuffers-1.12-py2.py3-none-any.whl (15 kB)
Requirement already satisfied: six~=1.15.0 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: absl-py~=0.10 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: astunparse~=1.6.3 in /usr/local/lib/python3.7/dist-
Collecting wrapt~=1.12.1
  Downloading wrapt-1.12.1.tar.gz (27 kB)
Requirement already satisfied: wheel~=0.35 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: google-pasta~=0.2 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: protobuf>=3.9.2 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: gast==0.4.0 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: setuptools>=41.0.0 in /usr/local/lib/python3.7/dist
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/pyt
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.7/d
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/d
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.7/
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-pack
```

```
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
from numpy import mean
from numpy import std
import numpy as np
from matplotlib import pyplot
from sklearn.model_selection import KFold
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Flatten
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import BatchNormalization
import tensorflow as tf
from tensorflow.keras.models import Model
from tensorflow.keras import backend as K
import matplotlib.pyplot as plt
import sklearn
from sklearn.metrics import confusion_matrix
from scipy.io import loadmat
import numpy as np
import PIL
import cv2
import os
from sklearn.model_selection import train_test_split
import autokeras as ak
```

```
# x = list()
# data = list()
# z = 0
# ##Class-1 images##
# folder_path_class1 = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet_
# #folder_path_class2 = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet
# #folder_path_class3 = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet
# #folder_path_class4 = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet
# #folder_path_class5 = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet

# folder_path_class1b = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet
# #folder_path_class2b = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physione
# #folder_path_class3b = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physione
# #folder_path_class4b = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physione
# #folder_path_class5b = ('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physione

# paths = [folder_path_class1,folder_path_class1b]
```

```
# for p in paths:
#   #print(p)
#   for image in os.walk(p):
#     data.append(image[2])
#     #print(image[2])

#   for i in range(len(data[0])):
#     str_complete = p + data[0][i]
#     #print(str_complete)
#     img = cv2.imread(str_complete)
#     img = cv2.resize(img, (128, 128))
#     x.append(img)#Ensure all images are loaded
#   data = []

# data_x = np.asarray(x)
# np.save('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet_Database_Evalu



# y = np.zeros(1000)
# y[500:] = 1
# np.save('/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet_Database_Evalu


x  = np.load("/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet_Database_Ev
y = np.load("/content/drive/MyDrive/PCG_signal_time_frequency_image/Physionet_Database_Eva


x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.1, random_state=1)
x_train, x_val, y_train, y_val = train_test_split(x_train, y_train, test_size=1/8, random_

y_tr_one_hot  = np.zeros((np.array(y_train).shape[0],2))
for i in range(np.array(y_train).shape[0]):
  label = y_train[i]
  y_tr_one_hot[i][int(label)] = 1

y_te_one_hot  = np.zeros((np.array(y_test).shape[0],2))
for i in range(np.array(y_test).shape[0]):
  label = y_test[i]
  y_te_one_hot[i][int(label)] = 1

y_val_one_hot  = np.zeros((np.array(y_val).shape[0],2))
for i in range(np.array(y_val).shape[0]):
  label = y_val[i]
  y_val_one_hot[i][int(label)] = 1


##Define model
def model_define():
  model=Sequential()
  model.add(Conv2D(60, kernel_size=(3, 3), activation='relu',input_shape=(128,128,3)))
  model.add(BatchNormalization())
  model.add(MaxPooling2D((2, 2)))
  model.add(Conv2D(100, (5, 5), activation='relu'))
  model.add(BatchNormalization())
```

```
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(120, (5, 5), activation='relu'))
    #model.add(BatchNormalization())
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(120, (3, 3), activation='relu'))
    #model.add(BatchNormalization())
    model.add(MaxPooling2D((2, 2)))
    model.add(Flatten())
    model.add(Dense(700, activation='relu'))
    model.add(Dropout(0.3))
    model.add(Dense(200, activation='relu'))
    model.add(Dropout(0.2))
    model.add(Dense(100,activation='relu'))
    #model.add(Dropout(0.2))
    model.add(Dense(50,activation='relu'))
    model.add(Dense(2,activation='softmax'))
    return model

K.clear_session()
#model = model_define()
#print(modelf.summary())

model = tf.keras.applications.VGG19(include_top=False,input_shape=(128,128,3))
# mark loaded layers as not trainable
for layer in model.layers:
    layer.trainable = False
# add new classifier layers
flat1 = Flatten()(model.layers[-1].output)
#class1 = Dense(2048, activation='relu')(flat1)
class2 = Dense(1024, activation='relu')(flat1)
class3 = Dense(512, activation='relu')(class2)
class4 = Dense(256,activation='relu')(class3)
class5 = Dense(128,activation='relu')(class4)
output = Dense(2, activation='softmax')(class5)
model = Model(inputs=model.inputs, outputs=output)

optimizer = tf.keras.optimizers.Adam(lr=0.000001)
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(np.array(x_train), y_tr_one_hot, validation_data=(np.array(x_val), y_val_one_hot
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications
80142336/80134624 [==============================] - 1s 0us/step
80150528/80134624 [==============================] - 1s 0us/step
Epoch 1/25
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarning
  super(Adam, self).__init__(name, **kwargs)
2/2 [==============================] - 26s 12s/step - loss: 16.7274 - accuracy: 0.
Epoch 2/25
2/2 [==============================] - 1s 294ms/step - loss: 9.3120 - accuracy: 0.
Epoch 3/25
2/2 [==============================] - 1s 290ms/step - loss: 1.7403 - accuracy: 0.
Epoch 4/25
2/2 [==============================] - 1s 288ms/step - loss: 1.9375 - accuracy: 0.
Epoch 5/25
2/2 [==============================] - 1s 289ms/step - loss: 0.7778 - accuracy: 0.
Epoch 6/25
```

```
2/2 [==============================] - 1s 288ms/step - loss: 1.3940 - accuracy: 0.
Epoch 7/25
2/2 [==============================] - 1s 288ms/step - loss: 0.9130 - accuracy: 0.
Epoch 8/25
2/2 [==============================] - 1s 290ms/step - loss: 0.9389 - accuracy: 0.
Epoch 9/25
2/2 [==============================] - 1s 288ms/step - loss: 1.0227 - accuracy: 0.
Epoch 10/25
2/2 [==============================] - 1s 286ms/step - loss: 0.7237 - accuracy: 0.
Epoch 11/25
2/2 [==============================] - 1s 286ms/step - loss: 0.6277 - accuracy: 0.
Epoch 12/25
2/2 [==============================] - 1s 289ms/step - loss: 0.5730 - accuracy: 0.
Epoch 13/25
2/2 [==============================] - 1s 296ms/step - loss: 0.7508 - accuracy: 0.
Epoch 14/25
2/2 [==============================] - 1s 287ms/step - loss: 0.6751 - accuracy: 0.
Epoch 15/25
2/2 [==============================] - 1s 290ms/step - loss: 0.6215 - accuracy: 0.
Epoch 16/25
2/2 [==============================] - 1s 289ms/step - loss: 0.5354 - accuracy: 0.
Epoch 17/25
2/2 [==============================] - 1s 289ms/step - loss: 0.5353 - accuracy: 0.
Epoch 18/25
2/2 [==============================] - 1s 289ms/step - loss: 0.5192 - accuracy: 0.
Epoch 19/25
2/2 [==============================] - 1s 290ms/step - loss: 0.4879 - accuracy: 0.
Epoch 20/25
2/2 [==============================] - 1s 295ms/step - loss: 0.4285 - accuracy: 0.
Epoch 21/25
2/2 [==============================] - 1s 287ms/step - loss: 0.4524 - accuracy: 0.
Epoch 22/25
2/2 [==============================] - 1s 288ms/step - loss: 0.4561 - accuracy: 0.
Epoch 23/25
2/2 [==============================] - 1s 288ms/step - loss: 0.4578 - accuracy: 0.
Epoch 24/25
2/2 [==============================] - 1s 289ms/step - loss: 0.4001 - accuracy: 0.
Epoch 25/25
2/2 [==============================] - 1s 291ms/step - loss: 0.3927 - accuracy: 0.
<keras.callbacks.History at 0x7f4d1691a210>
```

```
test_loss, test_acc = model.evaluate(np.array(x_test), np.array(y_te_one_hot), verbose=0)
print(test_acc)
##Evaluating Sensitivity, Accuracy and Kappa scores
y_prob = model.predict(x_test)
Y_pred = y_prob.argmax(axis=-1)

cm1 = confusion_matrix(y_test,Y_pred)
print("confusion matrix \n",cm1)
```

```
0.699999988079071
confusion matrix
 [[35 14]
 [16 35]]
```

Auto-Keras Code

```
input_node = ak.ImageInput()
output_node = ak.Normalization()(input_node)
output_node1 = ak.ConvBlock()(output_node)
output_node2 = ak.ConvBlock()(output_node1)
output_node = ak.ClassificationHead()(output_node)

auto_model = ak.AutoModel(
    inputs=input_node, outputs=output_node, overwrite=True,
)



print(x_train.shape)  # (60000, 28, 28)
print(y_tr_one_hot.shape)  # (60000,)
#print(y_train)  # array([7, 2, 1], dtype=uint8)

# Feed the AutoModel with training data.
auto_model.fit(x_train, y_train, epochs=50)
# Predict with the best model.
predicted_y = auto_model.predict(x_test)
# Evaluate the best model with testing data.
print(auto_model.evaluate(x_test, y_test))
```

```
Trial 17 Complete [00h 00m 52s]
val_loss: 0.921077311038971

Best val_loss So Far: 0.6870185732841492
Total elapsed time: 00h 07m 27s
INFO:tensorflow:Oracle triggered exit
Epoch 1/50
25/25 [==============================] - 1s 36ms/step - loss: 0.9506 - accuracy: 0
Epoch 2/50
25/25 [==============================] - 1s 36ms/step - loss: 0.8788 - accuracy: 0
Epoch 3/50
25/25 [==============================] - 1s 36ms/step - loss: 0.7865 - accuracy: 0
Epoch 4/50
25/25 [==============================] - 1s 36ms/step - loss: 0.6974 - accuracy: 0
Epoch 5/50
25/25 [==============================] - 1s 36ms/step - loss: 0.5806 - accuracy: 0
Epoch 6/50
25/25 [==============================] - 1s 37ms/step - loss: 0.5200 - accuracy: 0
Epoch 7/50
25/25 [==============================] - 1s 37ms/step - loss: 0.4476 - accuracy: 0
Epoch 8/50
25/25 [==============================] - 1s 36ms/step - loss: 0.3939 - accuracy: 0
Epoch 9/50
25/25 [==============================] - 1s 36ms/step - loss: 0.3538 - accuracy: 0
Epoch 10/50
25/25 [==============================] - 1s 36ms/step - loss: 0.3152 - accuracy: 0
Epoch 11/50
25/25 [==============================] - 1s 37ms/step - loss: 0.2846 - accuracy: 0
Epoch 12/50
25/25 [==============================] - 1s 37ms/step - loss: 0.2676 - accuracy: 0
Epoch 13/50
25/25 [==============================] - 1s 37ms/step - loss: 0.2341 - accuracy: 0
Epoch 14/50
25/25 [==============================] - 1s 37ms/step - loss: 0.2187 - accuracy: 0
```

```
Epoch 15/50
25/25 [==============================] - 1s 37ms/step - loss: 0.1950 - accuracy: 0
Epoch 16/50
25/25 [==============================] - 1s 38ms/step - loss: 0.1824 - accuracy: 0
Epoch 17/50
25/25 [==============================] - 1s 37ms/step - loss: 0.1820 - accuracy: 0
Epoch 18/50
25/25 [==============================] - 1s 37ms/step - loss: 0.1638 - accuracy: 0
Epoch 19/50
25/25 [==============================] - 1s 38ms/step - loss: 0.1627 - accuracy: 0
Epoch 20/50
25/25 [==============================] - 1s 38ms/step - loss: 0.1736 - accuracy: 0
Epoch 21/50
25/25 [==============================] - 1s 37ms/step - loss: 0.1874 - accuracy: 0
Epoch 22/50
25/25 [==============================] - 1s 37ms/step - loss: 0.1717 - accuracy: 0
Epoch 23/50
25/25 [==============================] - 1s 37ms/step - loss: 0.1504 - accuracy: 0
Epoch 24/50
25/25 [==============================] - 1s 37ms/step - loss: 0.1437 - accuracy: 0
Epoch 25/50
25/25 [==============================] - 1s 38ms/step - loss: 0.1290 - accuracy: 0
Epoch 26/50
```

```
##Define model
test_acc_list = []
K_cappa_list = []
precision_list = []
recall_list = []
F1_list = []

for iter in range(5):
  print("Trial Number : ", (iter+1))

  x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=1)
  #x_train, x_val, y_train, y_val = train_test_split(x_train, y_train, test_size=1/8, rand

  predicted_y = auto_model.predict(x_test)

  p_list = predicted_y.reshape(200).tolist()
  p_list_int = []
  for i in p_list:
    p_list_int.append(int(i))

  K_cappa = sklearn.metrics.cohen_kappa_score(y_test,p_list_int)
  print("cohen kappa scores:" ,K_cappa)
  K_cappa_list.append(K_cappa)

  cm1 = confusion_matrix(y_test,p_list_int)
  print("confusion matrix \n",cm1)

  precision = sklearn.metrics.precision_score(y_test,p_list_int,average='micro')
  print('precision : ', precision)
  precision_list.append(precision)
```

```
    recall = sklearn.metrics.recall_score(y_test,p_list_int,average='micro')
    print('recall : ', recall)
    recall_list.append(recall)

    F1 = sklearn.metrics.f1_score(y_test,p_list_int,average="micro")
    print("F1 : ", F1)
    F1_list.append(F1)

#  print('5-Trial Accuracy:',sum(test_acc_list)/len(test_acc_list))
  print("5-Trial cohen kappa scores:" ,sum(K_cappa_list)/len(K_cappa_list))
  print('5-Trial precision : ', sum(precision_list)/len(precision_list))
  print('5-Trial recall : ', sum(recall_list)/len(recall_list))
  print("5-Trial F1 : ", sum(F1_list)/len(F1_list))
```

```
Trial Number :  1
7/7 [==============================] - 0s 18ms/step
cohen kappa scores: 0.6794871794871795
confusion matrix
 [[80 16]
 [16 88]]
precision :  0.84
recall :  0.84
F1 :  0.8399999999999999
5-Trial cohen kappa scores: 0.6794871794871795
5-Trial precision :  0.84
5-Trial recall :  0.84
5-Trial F1 :  0.8399999999999999
Trial Number :  2
7/7 [==============================] - 0s 17ms/step
cohen kappa scores: 0.6794871794871795
confusion matrix
 [[80 16]
 [16 88]]
precision :  0.84
recall :  0.84
F1 :  0.8399999999999999
5-Trial cohen kappa scores: 0.6794871794871795
5-Trial precision :  0.84
5-Trial recall :  0.84
5-Trial F1 :  0.8399999999999999
Trial Number :  3
7/7 [==============================] - 0s 18ms/step
cohen kappa scores: 0.6794871794871795
confusion matrix
 [[80 16]
 [16 88]]
precision :  0.84
recall :  0.84
F1 :  0.8399999999999999
5-Trial cohen kappa scores: 0.6794871794871794
5-Trial precision :  0.84
5-Trial recall :  0.84
5-Trial F1 :  0.8399999999999999
Trial Number :  4
7/7 [==============================] - 0s 19ms/step
cohen kappa scores: 0.6794871794871795
confusion matrix
 [[80 16]
```

```
 [16 88]]
precision :  0.84
recall :  0.84
F1 :  0.8399999999999999
5-Trial cohen kappa scores: 0.6794871794871795
5-Trial precision :  0.84
5-Trial recall :  0.84
5-Trial F1 :  0.8399999999999999
Trial Number :  5
7/7 [==============================] - 0s 18ms/step
cohen kappa scores: 0.6794871794871795
confusion matrix
 [[80 16]
```