```
##installing modules for upgrading !pip install --upgrade module_name



import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
img_size=224
batch_size=32


train_datagen=ImageDataGenerator(rescale=1./255,validation_split=0.2)
train_generator=train_datagen.flow_from_directory(
      r'/content/drive/MyDrive/Train-20240627T045845Z-001',
    target_size=(img_size,img_size),
    batch_size=32,
    class_mode='binary',
    subset='training'
)

val_generator=train_datagen.flow_from_directory(
    r'/content/drive/MyDrive/Train-20240627T045845Z-001',
    target_size=(img_size,img_size),
    batch_size=32,
    class_mode='binary',
    subset='validation'

)
```

```
Found 2408 images belonging to 1 classes.
Found 602 images belonging to 1 classes.
```

```
model=keras.Sequential([
    layers.Conv2D(32,(3,3),activation='relu',input_shape=(img_size,img_size,3)),
    layers.MaxPooling2D(2,2),
    layers.Conv2D(64,(3,3),activation='relu'),
    layers.MaxPooling2D(2,2),
    layers.Conv2D(128,(3,3),activation='relu'),
    layers.MaxPooling2D(2,2),
    layers.Flatten(),
    layers.Dense(128,activation='relu'),
    layers.Dense(1,activation='sigmoid')
])


model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])


model.fit(train_generator,validation_data=val_generator,epochs=5)
```

```
Epoch 1/5
76/76 [==============================] - 468s 6s/step - loss: 0.0092 - accuracy: 0.9946 - val_loss: 0.0000e+00 - val_a
Epoch 2/5
76/76 [==============================] - 347s 5s/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - v
Epoch 3/5
76/76 [==============================] - 307s 4s/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - v
Epoch 4/5
76/76 [==============================] - 313s 4s/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - v
Epoch 5/5
76/76 [==============================] - 303s 4s/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - v
<keras.src.callbacks.History at 0x7e7ef86d07f0>
```

```
model.save("Model.h5")
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as a
  saving_api.save_model(
```

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np


model=load_model(r'/content/Model.h5')
print("model loaded")
```

```
model loaded
```

```
test_image_path=r"/content/WhatsApp Image 2024-06-27 at 12.44.01.jpeg"
img=image.load_img(test_image_path,target_size=(224,224))
img_array=image.img_to_array(img)
img_array=np.expand_dims(img_array,axis=0)


img_array /=225
prediction=model.predict(img_array)
print(prediction)
```

```
1/1 [==============================] - 0s 121ms/step
[[0.]]
```

```
test_image_path=r"/content/y123.jpg"
img=image.load_img(test_image_path,target_size=(224,224))
img_array=image.img_to_array(img)
img_array=np.expand_dims(img_array,axis=0)


img_array /=225
prediction=model.predict(img_array)
print(prediction)
```

```
1/1 [==============================] - 0s 54ms/step
[[5.8442634e-12]]
```

```
test_image_path=r"/content/y125.jpg"
img=image.load_img(test_image_path,target_size=(224,224))
img_array=image.img_to_array(img)
img_array=np.expand_dims(img_array,axis=0)


img_array /=225
prediction=model.predict(img_array)
print(prediction)
```

```
1/1 [==============================] - 0s 69ms/step
[[0.]]
```

KNN ALGORITHM : K NEAEST NEIGHBOURS

```
!pip install -q scikit-learn
import numpy as np
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
```

```
data={
    'BP':[120,130,140,150,160,170,180,190,200,210],
    'Cholestrol':[200,220,240,260,280,300,320,340,360,380],
    'HeartRisk':[0,0,0,0,1,1,1,1,1,1]
}


df=pd.DataFrame(data)


k=3
knn=KNeighborsClassifier(n_neighbors=k)


x=df[['BP','Cholestrol']]
y=df['HeartRisk']


knn.fit(x,y)
```

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=3)
```

```
new_data=np.array([[210,250]])
prediction=knn.predict(new_data)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but KNe
  warnings.warn(
```

```
if prediction==0:
  print("no risk")
else:
  print("ar risk")
```

```
ar risk
```