In [2]:

```
import keras
from keras.preprocessing.image import ImageDataGenerator
```

In [18]:

```
train_data=ImageDataGenerator(rescale=1./255,shear_range=0.3,zoom_range=0.2,rotation_range=
    vertical_flip=True)
test_data=ImageDataGenerator(rescale=1./255)
validation_data=ImageDataGenerator(rescale=1./255)
```

In [28]:

```
train_path=train_data.flow_from_directory(directory=r'C:\image\train',target_size=(256, 256
test_path=test_data.flow_from_directory(directory=r'C:\image\test')
validation_path=validation_data.flow_from_directory(directory=r'C:\image\train')
```

```
Found 23 images belonging to 3 classes.
Found 10 images belonging to 3 classes.
Found 23 images belonging to 3 classes.
```

In [48]:

```
train_path.class_indices
```

Out[48]:

```
{'huny': 0, 'khushbu': 1, 'rinku': 2}
```

In [29]:

```
from keras.layers import Dense,Flatten,Conv2D,MaxPool2D,Dropout,MaxPooling2D
from keras.models import Sequential
```

In [33]:

```
model=Sequential()
model.add(Conv2D(input_shape=(256,256,3), kernel_size=(5,5),padding='same',filters=16,strid
model.add(MaxPooling2D(pool_size=(2,2),strides=2))
model.add(Conv2D(kernel_size=(5,5), padding='same',filters=32,strides=1,activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2),strides=2))
model.add(Flatten())
model.add(Dense(units=2000,activation='relu'))
# model.add(Dropout(0.30))
model.add(Dense(units=1000,activation='relu'))
model.add(Dense(units=400,activation='relu'))
# model.add(Dropout(0.30))
model.add(Dense(units=3,activation='softmax'))

model.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_8 (Conv2D)           (None, 256, 256, 16)      1216

 max_pooling2d_8 (MaxPooling  (None, 128, 128, 16)     0
 2D)

 conv2d_9 (Conv2D)           (None, 128, 128, 32)      12832

 max_pooling2d_9 (MaxPooling  (None, 64, 64, 32)       0
 2D)

 flatten_4 (Flatten)         (None, 131072)            0

 dense_16 (Dense)            (None, 2000)              262146000

 dense_17 (Dense)            (None, 1000)              2001000

 dense_18 (Dense)            (None, 400)               400400

 dense_19 (Dense)            (None, 3)                 1203

=================================================================
Total params: 264,562,651
Trainable params: 264,562,651
Non-trainable params: 0
_________________________________________________________________
```

In [34]:

```
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics='accuracy')
```

In [35]:

```
model_training=model.fit(x=train_path,batch_size=32,epochs=10,validation_data=validation_pa
```

```
Epoch 1/10
1/1 [==============================] - 10s 10s/step - loss: 1.1049 - accurac
y: 0.3043 - val_loss: 50.0544 - val_accuracy: 0.5652
Epoch 2/10
1/1 [==============================] - 9s 9s/step - loss: 47.4810 - accurac
y: 0.5652 - val_loss: 6.3484 - val_accuracy: 0.2174
Epoch 3/10
1/1 [==============================] - 7s 7s/step - loss: 6.1345 - accuracy:
0.2174 - val_loss: 6.0669 - val_accuracy: 0.2174
Epoch 4/10
1/1 [==============================] - 6s 6s/step - loss: 5.7044 - accuracy:
0.2174 - val_loss: 4.0487 - val_accuracy: 0.5652
Epoch 5/10
1/1 [==============================] - 6s 6s/step - loss: 3.6411 - accuracy:
0.5652 - val_loss: 2.2782 - val_accuracy: 0.5652
Epoch 6/10
1/1 [==============================] - 6s 6s/step - loss: 2.0495 - accuracy:
0.5652 - val_loss: 1.1352 - val_accuracy: 0.5652
Epoch 7/10
1/1 [==============================] - 5s 5s/step - loss: 1.1018 - accuracy:
0.5652 - val_loss: 0.9290 - val_accuracy: 0.5652
Epoch 8/10
1/1 [==============================] - 4s 4s/step - loss: 0.9307 - accuracy:
0.5652 - val_loss: 0.9348 - val_accuracy: 0.6087
Epoch 9/10
1/1 [==============================] - 4s 4s/step - loss: 0.9789 - accuracy:
0.5652 - val_loss: 0.9516 - val_accuracy: 0.6957
Epoch 10/10
1/1 [==============================] - 5s 5s/step - loss: 0.9899 - accuracy:
0.6522 - val_loss: 0.9491 - val_accuracy: 0.6957
```

In [38]:

```
model.save('multiclassmodel')
```

```
INFO:tensorflow:Assets written to: multiclassmodel\assets
```

In [39]:

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

In [49]:

```
test_image=image.load_img(r'C:\data\test\khushbu\k1.jpg',target_size=(256,256))
test_image=image.img_to_array(test_image)
test_image=np.expand_dims(test_image,axis=0)

model=load_model('multiclassmodel')
result=model.predict(test_image)

if result[0][0]==1:
    print("huny")
elif result[0][1]==1:
    print("khushbu")
else:
    print("rinku")
```

```
khushbu
```