```
#import keras libraries
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten


#image preprocessing(or) image augmentation
from keras.preprocessing.image import ImageDataGenerator


train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizonta
#rescale => rescaling pixel value from 0 to 255 to 0 to 1
#shear_range=> counter clock wise rotation(anti clock)



test_datagen = ImageDataGenerator(rescale=1./255)


x_train = train_datagen.flow_from_directory("/content/drive/MyDrive/DATA/TRAIN_SET",target
```

```
     Found 162 images belonging to 5 classes.
```

```
x_test = test_datagen.flow_from_directory("/content/drive/MyDrive/DATA/TEST_SET",target_si
```

```
     Found 1055 images belonging to 5 classes.
```

```
x_train.class_indices
```

```
     {'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}
```

```
#checking the number of classes
print(x_test.class_indices)
```

```
     {'APPLES': 0, 'BANANA': 1, 'ORANGE': 2, 'PINEAPPLE': 3, 'WATERMELON': 4}
```

```
from collections import Counter as c
c(x_train .labels)
```

```
     Counter({0: 162})
```

```
#Initializing the model
model = Sequential()


model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))
# 32 indicates => no of feature detectors
#(3,3)=> kernel size (feature detector size)
```

```
# add Maxpooling layer
model.add(MaxPooling2D(pool_size=(2,2)))
```

SECOND CONVOLUTION THEORY

```
model.add(Convolution2D(32,(3,3),activation='relu'))


model.add(MaxPooling2D(pool_size=(2,2)))


#Flattening the layers
model.add(Flatten())


model.add(Dense(units=128,activation='relu'))


model.add(Dense(units=5,activation='softmax'))
# add flatten layer => input to your ANN


model.add(Flatten())


model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 29, 29, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 flatten (Flatten)           (None, 6272)              0

 dense (Dense)               (None, 128)               802944

 dense_1 (Dense)             (None, 5)                 645

 flatten_1 (Flatten)         (None, 5)                 0

=================================================================
Total params: 813,733
Trainable params: 813,733
Non-trainable params: 0
_________________________________________________________________
```

```
model.add(Dense(units=300,kernel_initializer="random_uniform",activation="relu"))
model.add(Dense(units=200,kernel_initializer="random_uniform",activation="relu"))
```

Output Layer

```
model.add(Dense(units=4,kernel_initializer="random_uniform",activation="softmax"))
len(x_train)
```

```
6
```

```
#Ann starts so need to add dense layers
model.add(Dense(units=128,activation="relu",kernel_initializer="random_uniform"))
model.add(Dense(units=1,activation="sigmoid",kernel_initializer="random_uniform"))
```

```
#Compile the model
model.compile(loss="binary_crossentropy",optimizer="adam",metrics=['accuracy'])
```

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

```
#Prediction the result
from tensorflow.keras.models import load_model
from keras.preprocessing  import image
model =load_model("nutrition.h5")
```

```
import numpy as np
from tensorflow.keras.utils import load_img
from tensorflow.keras.utils import img_to_array
#loading of the image
img = load_img(r'/content/drive/MyDrive/DATA/TEST_SET/APPLES/151_100.jpg', grayscale=False
#image to array
x = img_to_array(img)
#changing the shape
x= np.expand_dims(x,axis = 0)
predict_x=model.predict(x)
classes_x=np.argmax(predict_x,axis = -1)
classes_x
```

```
1/1 [==============================] - 0s 20ms/step
array([0])
```

```
index=['APPLES', 'BANANA', 'ORANGE', 'PINEAPPLE', 'WATERMELON']
result=str(index[classes_x[0]])
result
```

```
'APPLES'
```

Colab paid products - Cancel contracts here

✓ 0s completed at 9:54 PM