Project Development Phase
Model Performance Test

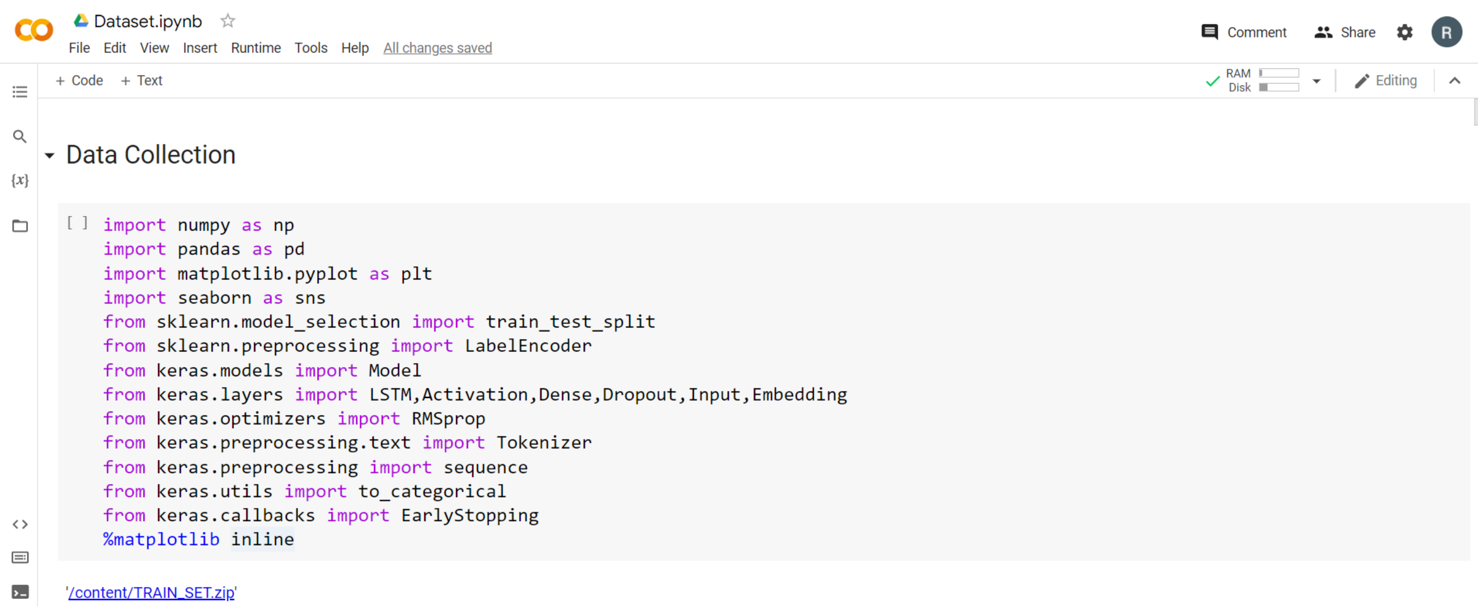
Date	17 November 2022
Team ID	PNT2022TMID4406
Project Name	AI-powered Nutrition Analyzer for Fitness Enthusiasts
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

SI.No.	Parameter	Values	Screenshot
1.	Model Summary	Total params: 21,885,485 Trainable params: 1,024,005 Non-trainable params: 20,861,480	Attached below
2.	Accuracy	Training Accuracy - 72% Validation Accuracy - 59%	Attached below
3.	Confidence Score (Only Yolo Projects)	Class Detected - NIL Confidence Score - NIL	NIL

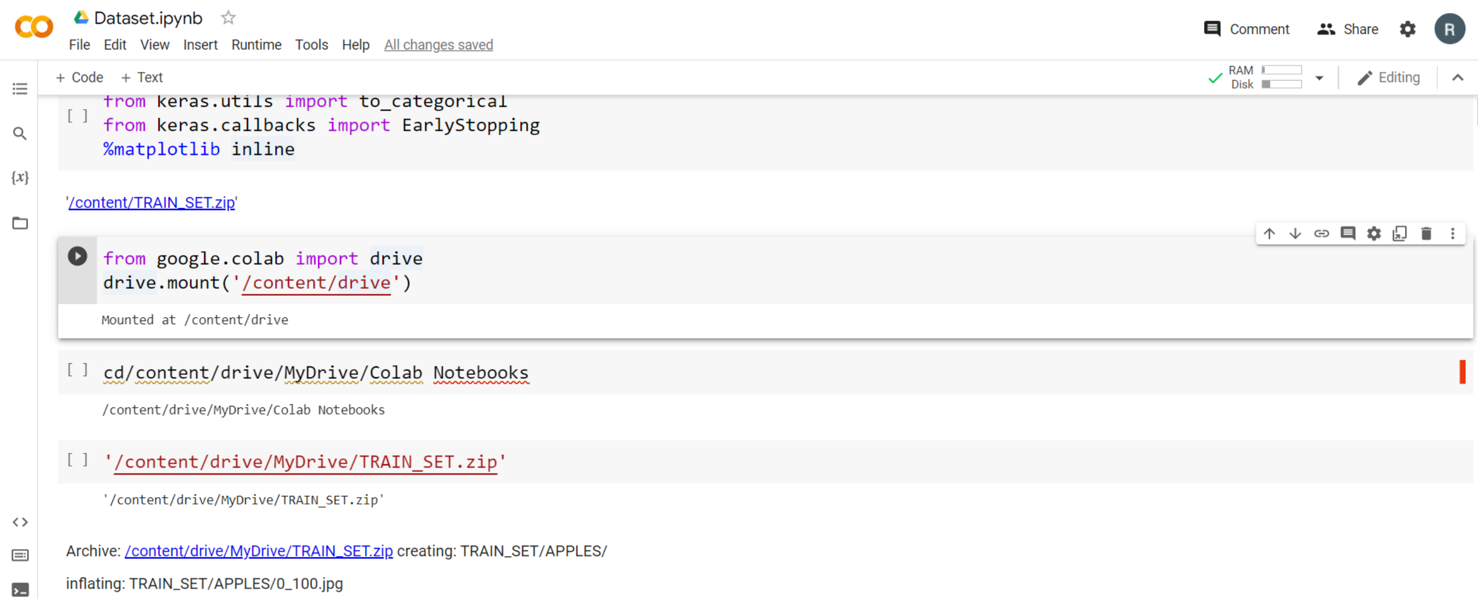
SCREENSHOTS :



Data Collection

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.models import Model
from keras.layers import LSTM,Activation,Dense,Dropout,Input,Embedding
from keras.optimizers import RMSprop
from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
%matplotlib inline
```

'/content/TRAIN_SET.zip'



```
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
%matplotlib inline
```

'/content/TRAIN_SET.zip'

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

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

```
cd/content/drive/MyDrive/Colab Notebooks
```

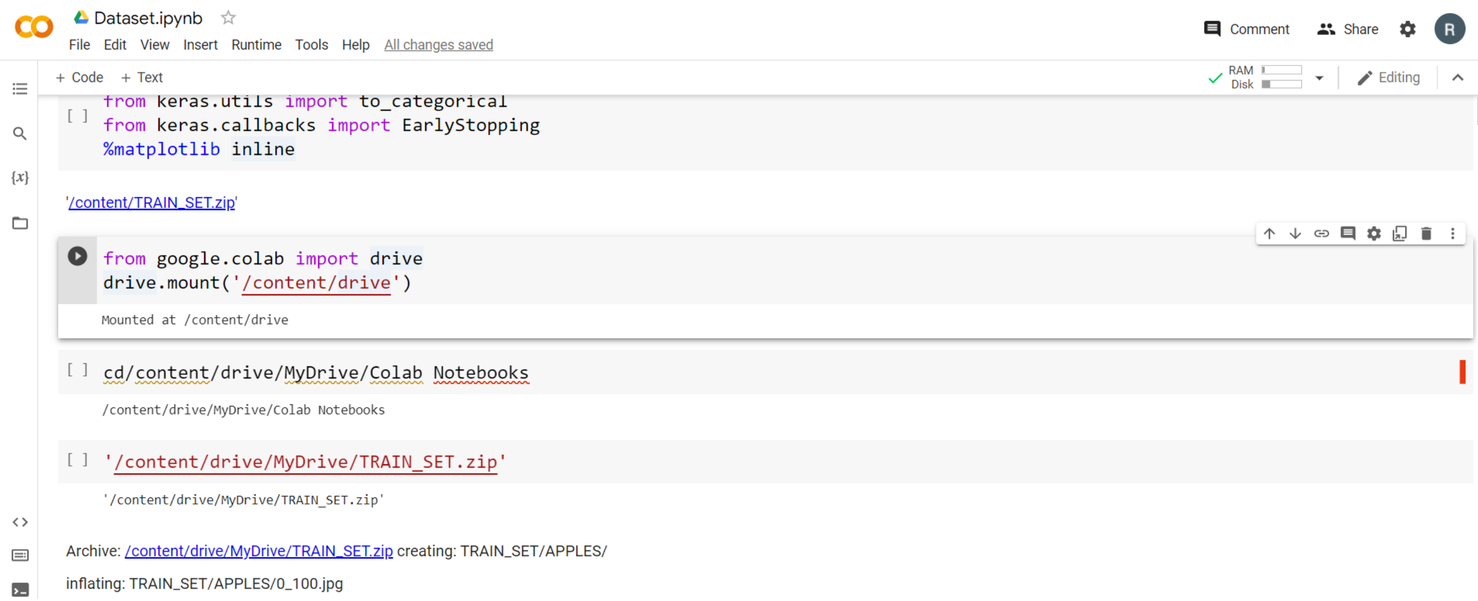
```
/content/drive/MyDrive/Colab Notebooks
```

```
'/content/drive/MyDrive/TRAIN_SET.zip'
```

```
'/content/drive/MyDrive/TRAIN_SET.zip'
```

Archive: /content/drive/MyDrive/TRAIN_SET.zip creating: TRAIN_SET/APPLES/

inflating: TRAIN_SET/APPLES/0_100.jpg



```
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
%matplotlib inline
```

'/content/TRAIN_SET.zip'

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

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

```
cd/content/drive/MyDrive/Colab Notebooks
```

```
/content/drive/MyDrive/Colab Notebooks
```

```
'/content/drive/MyDrive/TRAIN_SET.zip'
```

```
'/content/drive/MyDrive/TRAIN_SET.zip'
```

Archive: /content/drive/MyDrive/TRAIN_SET.zip creating: TRAIN_SET/APPLES/

inflating: TRAIN_SET/APPLES/0_100.jpg

In [1]:

```
import numpy as np#used for numerical analysis
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A layer consists of a tensor-in tensor-out computation function
#Dense layer is the regular deeply connected neural network layer
from tensorflow.keras.layers import Dense,Flatten
#Faltten-used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Dropout #Convolutional layer
#MaxPooling2D-for downsampling the image
from keras.preprocessing.image import ImageDataGenerator
```

In [2]:

```
#setting parameter for Image Data agumentation to the training data
train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
#Image Data agumentation to the testing data
test_datagen=ImageDataGenerator(rescale=1./255)
```

In [ ]:

```
#performing data agumentation to train data
x_train = train_datagen.flow_from_directory(
    r'C:\Users\Welcome-pc\Downloads\TRAIN_SET\TRAIN_SET',
    target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')
#performing data agumentation to test data
x_test = test_datagen.flow_from_directory(
    r'C:\Users\Welcome-pc\Downloads\TEST_SET-20221109T113651Z-001\TEST_SET',
    target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')
```

```
Found 2626 images belonging to 5 classes.

Found 1055 images belonging to 5 classes.
```

In [1]:

```
import numpy as np#used for numerical analysis
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A layer consists of a tensor-in tensor-out computation function
#Dense layer is the regular deeply connected neural network layer
from tensorflow.keras.layers import Dense,Flatten
#Faltten-used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Dropout #Convolutional layer
#MaxPooling2D-for downsampling the image
from keras.preprocessing.image import ImageDataGenerator
```

In [2]:

```
#setting parameter for Image Data agumentation to the training data
train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
#Image Data agumentation to the testing data
test_datagen=ImageDataGenerator(rescale=1./255)
```

In [ ]:

```
#performing data agumentation to train data
x_train = train_datagen.flow_from_directory(
    r'C:\Users\Welcome-pc\Downloads\TRAIN_SET\TRAIN_SET',
    target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')
#performing data agumentation to test data
x_test = test_datagen.flow_from_directory(
    r'C:\Users\Welcome-pc\Downloads\TEST_SET-20221109T113651Z-001\TEST_SET',
    target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')
```

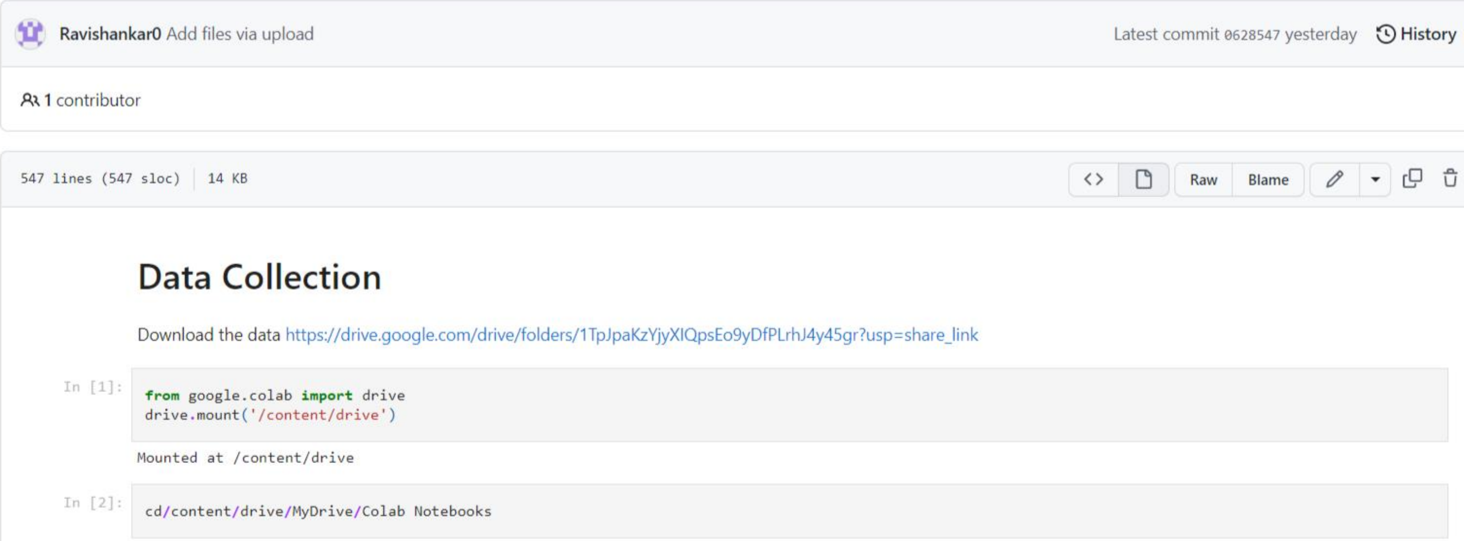
```
Found 2626 images belonging to 5 classes.

Found 1055 images belonging to 5 classes.
```

```python
# Flattening the layers
classifier.add(Flatten())

# Adding a fully connected layer
classifier.add(Dense(units=128, activation='relu'))
classifier.add(Dense(units=5, activation='softmax')) # softmax for more than 2
```

In [3]:

```python
classifier.summary()#summary of our model
```

```
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

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



Ravishankar0 Add files via upload

Latest commit 0628547 yesterday

1 contributor

547 lines (547 sloc) | 14 KB

Data Collection

Download the data https://drive.google.com/drive/folders/1TpJpaKzYjyXIQpsEo9yDfPLrhJ4y45gr?usp=share_link

In [1]:

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

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

In [2]:

```
cd/content/drive/MyDrive/Colab Notebooks
```

Data Collection

Download the data https://drive.google.com/drive/folders/1TpJpaKzYjyXIQpsEo9yDfPLrhJ4y45gr?usp=share_link

In [1]:

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

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

In [2]:

```
cd/content/drive/MyDrive/Colab Notebooks
```

```
/content/drive/MyDrive/Colab Notebooks
```

In [ ]:

```
# Unzipping the dataset
!unzip 'TRAIN_SET.zip'
```

Archive: Dataset.zip replace Dataset/TEST_SET/APPLES/n07740461_10011.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename:

IMAGE PROCESSING:

In [28]:

```
#Importing The ImageDataGenerator Library
from keras.preprocessing.image import ImageDataGenerator
```

IMAGE DATA AUGUMENTATION:

Applying Image DataGenerator Functionality To Trainset And Testset

In [ ]:

```
#Applying Image DataGenerator Functionality To Trainset And Testset
x_train = train_datagen.flow_from_directory(
    r'/content/drive/MyDrive/TRAIN_SET.zip',
    target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')
#Applying Image DataGenerator Functionality To Testset
x_test = test_datagen.flow_from_directory(
    r'/content/drive/MyDrive/TEST_SET-20221109T113651Z-001.zip',
    target_size=(64, 64),batch_size=5,color_mode='rgb',class_mode='sparse')
```

```
Found 4118 images belonging to 5 classes.

Found 929 images belonging to 5 classes.
```

In [ ]:

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

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

In [ ]:

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

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

In [ ]:

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

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

In [ ]:

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

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

In [ ]:

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

```
Counter({0: 995, 1: 1354, 2: 1019, 3: 275, 4: 475})
```

Model Building:

1] Importing The Model Building Libraries

In [ ]:

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense,Flatten
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Dropout
```

2] Initializing The Model

In [ ]:

```
model = Sequential()
```

3] Adding CNN Layers

In [ ]:

```
# Initializing the CNN
classifier = Sequential()

# First convolution layer and pooling
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Second convolution layer and pooling
classifier.add(Conv2D(32, (3, 3), activation='relu'))

# input_shape is going to be the pooled feature maps from the previous convolution layer
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Flattening the layers
classifier.add(Flatten())
```

4] Adding Dense Layers

In [ ]:

```
classifier.add(Dense(units=128, activation='relu'))
classifier.add(Dense(units=5, activation='softmax'))
```

In [38]:

```
#summary of our model
classifier.summary()
```

```
Model: "sequential_1"
```

In [ ]:

```
classifier.add(Dense(units=128, activation='relu'))
classifier.add(Dense(units=5, activation='softmax'))
```

In [38]:

```
#summary of our model
classifier.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 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

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

5] Configure The Learning Process

In [ ]:

```
# Compiling the CNN
# categorical_crossentropy for more than 2
classifier.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

6] Train The Model

In [ ]:

```
#Fitting the model
classifier.fit_generator(generator=x_train,steps_per_epoch = len(x_train),epochs=20, validation_data=x_test,validation_steps = len(x_test))
```

7] Saving The Model

In [30]:

```
classifier.save('nutrition.h5')
```

8] Testing The Model

In [31]:

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

In [ ]:

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model = load_model("nutrition.h5")
#loading of the image
img = image.load_img(r'/content/drive/MyDrive/Colab Notebooks/Sample_Images/Test_Image1.jpg',grayscale=False,target_size= (64,64))
#image to array
x = img_to_array(img)
```

In [ ]:

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model = load_model("nutrition.h5")
#loading of the image
img = image.load_img(r'/content/drive/MyDrive/Colab Notebooks/Sample_Images/Test_Image1.jpg',grayscale=False,target_size= (64,64))
#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 62ms/step
```

```
array([0])
```

In [ ]:

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

```
'APPLES'
```