

Sprint 1 – Model Building

Date	10-11-2022
Team ID	PNT2022TMID26433
Project Name	AI Powered Nutrition Analyzer for Fitness Enthusiasts

Data Set:

In our Dataset we have collected images of the five variety of fruits.

- Apple
- Orange
- Watermelon
- Muskmelon
- Banana

Image Pre-processing:

- ❖ Importing the imagedatagenerator library

```
From keras.preprocessing.image import  
imagedatagenerator
```

- ❖ Configuring imagedatageneratorclass

```
Train_datagen=imagedatagenerator(rescale =  
1./255,shear_range = 0.2, zoom_range = 0.2,  
horizontal_flip = true)
```

```
Test_datagen = imagedatagenerator(rescale = 1./255)
```

- ❖ Applying image datagenerator functionality to trainset and testset

```
X_train=train_datagen.flow_from_directory(r'/content/data  
set/TRAIN_SET', target_size = (64,64), batch_size = 5,  
colour_mode = 'rgb', class_mode = 'sparse')
```

```
X_train=train_datagen.flow_from_directory(r'/content/dataset/TEST_SET', target_size = (64,64), batch_size = 5, colour_mode = 'rgb', class_mode = 'sparse')
```

Data Collection:

```
# unzipping the dataset
```

```
! unzip'/content/dataset.zip'
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_288_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_289_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_28_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_290_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_291_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_292_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_293_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_294_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_295_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_296_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_297_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_298_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_299_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_29_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_2_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_300_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_301_100.jpg
```

```
Inflating:Dataset/TRAIN_SET/WATERMELON/r_302_100.jpg
```

Inflating:Dataset/TRAIN_SET/WATERMELON/r_303_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_304_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_305_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_306_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_307_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_308_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_309_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_30_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_310_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_311_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_312_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_313_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_314_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_315_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_31_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_32_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_33_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_34_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_35_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_36_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_37_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_38_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_39_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_3_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_40_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_41_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_42_100.jpg

Inflating:Dataset/TRAIN_SET/WATERMELON/r_43_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_44_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_45_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_46_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_4_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_50_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_57_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_5_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_67_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_81_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_8_100.jpg
Inflating:Dataset/TRAIN_SET/WATERMELON/r_9_100.jpg

Image Pre-processing:

Importing the imagedatagenerator library

From keras.preprocessing.image import imagedatagenerator

Image Data Augmentation:

Configuring imagedatageneratorclass

Train_datagen=imagedatagenerator(rescale = 1./255,shear_range
= 0.2, zoom_range = 0.2, horizontal_Test_datagen) =
imagedatagenerator(rescale = 1./255)

Applying image data generator functionality to trainset and testset:

Applying image data generator functionality to trainset
and testset

```
X_train=train_datagen.flow_from_directory(r'/content/dataset/TRAIN_SET', target_size = (64,64), batch_size = 5,
colour_mode = 'rgb', class_mode = 'sparse')
```

Applying image data generator functionality to testset

```
X_train=train_datagen.flow_from_directory(r'/content/dataset/TEST_SET', target_size = (64,64), batch_size = 5,
colour_mode = 'rgb', class_mode = 'sparse')
```

Found 4118 images belonging to the 5 classes

found 929 images belonging to 5 classes

Checking the number of classes

```
Print(x_train.class_indices)
```

```
{'APPLE': 0, 'BANANA': 1, 'ORANGE': 2, 'MUSKMELON': 3, 'WATERMELON': 4}
```

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

```
import numpy as np
```

```
import tensorflow
```

```
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

```
model = Sequential ()
```

3. Adding CNN Layers

```
# 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

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

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

```
# 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
2D)
(None, 14, 14, 32) 0
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

Compiling the CNN

```

# categorical_crossentropy for more than 2
classifier.compile(optimizer='adam',
loss='sparse_categorical_crossentropy', metrics=['acc

```