

Sprint -4

| |
|--|
| Date 17 November 2022 |
| Team ID PNT2022TMID33736 |
| Project Name Project - AI-Powered Nutrition Analyzer for Fitness Enthusiasts |

Model Creation

Importing libraries

```
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
```

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

Initializing the Model

```
model = Sequential()
```

Adding CNN Layers

```
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())
```

Adding Dense Layers

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

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

```
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 (MaxPooling2D) | (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
=====

Configure the Learning Process

Compiling the CNN

```
# categorical_crossentropy for more than 2
```

```
classifier.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

Train 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))
```

```
Model Building.ipynb ☆
File Edit View Insert Runtime Tools Help All changes saved

Reconnect Editing

***Entry point for launching an IPython kernel.
Epoch 1/20
826/826 [=====] - 1315s 2s/step - loss: 0.6087 - accuracy: 0.7599 - val_loss: 0.5782 - val_accuracy: 0.7847
Epoch 2/20
826/826 [=====] - 47s 56ms/step - loss: 0.4062 - accuracy: 0.8454 - val_loss: 0.4071 - val_accuracy: 0.8558
Epoch 3/20
826/826 [=====] - 46s 50ms/step - loss: 0.3754 - accuracy: 0.8607 - val_loss: 0.4110 - val_accuracy: 0.8471
Epoch 4/20
826/826 [=====] - 45s 55ms/step - loss: 0.3381 - accuracy: 0.8728 - val_loss: 0.4668 - val_accuracy: 0.8170
Epoch 5/20
826/826 [=====] - 49s 59ms/step - loss: 0.3282 - accuracy: 0.8769 - val_loss: 0.4054 - val_accuracy: 0.8493
Epoch 6/20
826/826 [=====] - 47s 56ms/step - loss: 0.3081 - accuracy: 0.8854 - val_loss: 0.4247 - val_accuracy: 0.8418
Epoch 7/20
826/826 [=====] - 45s 54ms/step - loss: 0.2897 - accuracy: 0.8900 - val_loss: 0.4057 - val_accuracy: 0.8590
Epoch 8/20
826/826 [=====] - 46s 56ms/step - loss: 0.2746 - accuracy: 0.8932 - val_loss: 0.4180 - val_accuracy: 0.8741
Epoch 9/20
826/826 [=====] - 45s 54ms/step - loss: 0.2689 - accuracy: 0.8995 - val_loss: 0.4639 - val_accuracy: 0.8418
Epoch 10/20
826/826 [=====] - 46s 55ms/step - loss: 0.2456 - accuracy: 0.9092 - val_loss: 0.3555 - val_accuracy: 0.8773
Epoch 11/20
826/826 [=====] - 46s 56ms/step - loss: 0.2278 - accuracy: 0.9104 - val_loss: 0.3919 - val_accuracy: 0.8622
Epoch 12/20
826/826 [=====] - 46s 56ms/step - loss: 0.2104 - accuracy: 0.9213 - val_loss: 0.3689 - val_accuracy: 0.8751
Epoch 13/20
826/826 [=====] - 46s 56ms/step - loss: 0.2100 - accuracy: 0.9191 - val_loss: 0.3579 - val_accuracy: 0.8827
Epoch 14/20
826/826 [=====] - 47s 57ms/step - loss: 0.1906 - accuracy: 0.9319 - val_loss: 0.4280 - val_accuracy: 0.8611
Epoch 15/20
826/826 [=====] - 46s 55ms/step - loss: 0.1827 - accuracy: 0.9329 - val_loss: 0.3347 - val_accuracy: 0.9011
Epoch 16/20
826/826 [=====] - 43s 52ms/step - loss: 0.1636 - accuracy: 0.9394 - val_loss: 0.4189 - val_accuracy: 0.8579
Epoch 17/20
826/826 [=====] - 47s 57ms/step - loss: 0.1609 - accuracy: 0.9397 - val_loss: 0.3509 - val_accuracy: 0.8967
Epoch 18/20
826/826 [=====] - 46s 56ms/step - loss: 0.1363 - accuracy: 0.9479 - val_loss: 0.3901 - val_accuracy: 0.8924
Epoch 19/20
826/826 [=====] - 46s 55ms/step - loss: 0.1139 - accuracy: 0.9537 - val_loss: 0.4557 - val_accuracy: 0.8730
Epoch 20/20
826/826 [=====] - 45s 55ms/step - loss: 0.1179 - accuracy: 0.9566 - val_loss: 0.3902 - val_accuracy: 0.9042
<keras.callbacks.History at 0x7fc096200250>

0s completed at 10:58 AM
```

Save the Model

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

Test the Model

#Predict the results

```
from tensorflow.keras.models import load_model
```

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

```
keras_preprocessing.image import load_img
```

```
model = load_model("ainutrition.h5")
```

```
from tensorflow.keras.utils import img_to_array
```

```
#loading of the image
```

```
img = load_img(r'/content/drive/MyDrive/DataSet-IBM/TEST_SET/ORANGE/n07749192_1251.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 107ms/step  
array([2])
```

```
index=['APPLES', 'BANANA', 'ORANGE','PINEAPPLE','WATERMELON']
```

```
result=str(index[classes_x[0]])
```

```
result
```

```
↳ 'ORANGE'
```



```
print(result)
```

```
if result == 'APPLES':
```

```
    print("One serving, or one medium apple, provides about 95 calories, 0 gram fat, 1 gram protein, 25 grams carbohydrate, 19 grams sugar (naturally occurring), and 3 grams fiber.")
```

```
elif result == 'BANANA':
```

```
    print("One serving, or one medium ripe banana, provides about 110 calories, 0 gram fat, 1 gram protein, 28 grams carbohydrate, 15 grams sugar (naturally occurring), 3 grams fiber, and 450 mg potassium.")
```

```
elif result == 'ORANGE':
```

```
    print("60 calories, No fat or sodium, 3 grams of fiber, 12 grams of sugar, 1 gram of protein, 14 micrograms of vitamin A, 70 milligrams of vitamin C, 6% of your daily recommended amount of calcium.")
```

```
elif result == 'PINEAPPLE':
```

```
    print("Calories: 83, Fat: 1.7 grams, Protein: 1 gram, Carbs: 21.6 grams, Fiber: 2.3 grams, Vitamin C: 88% of the
```

Daily Value (DV), Manganese: 109% of the DV, Vitamin B6: 11% of the DV.")

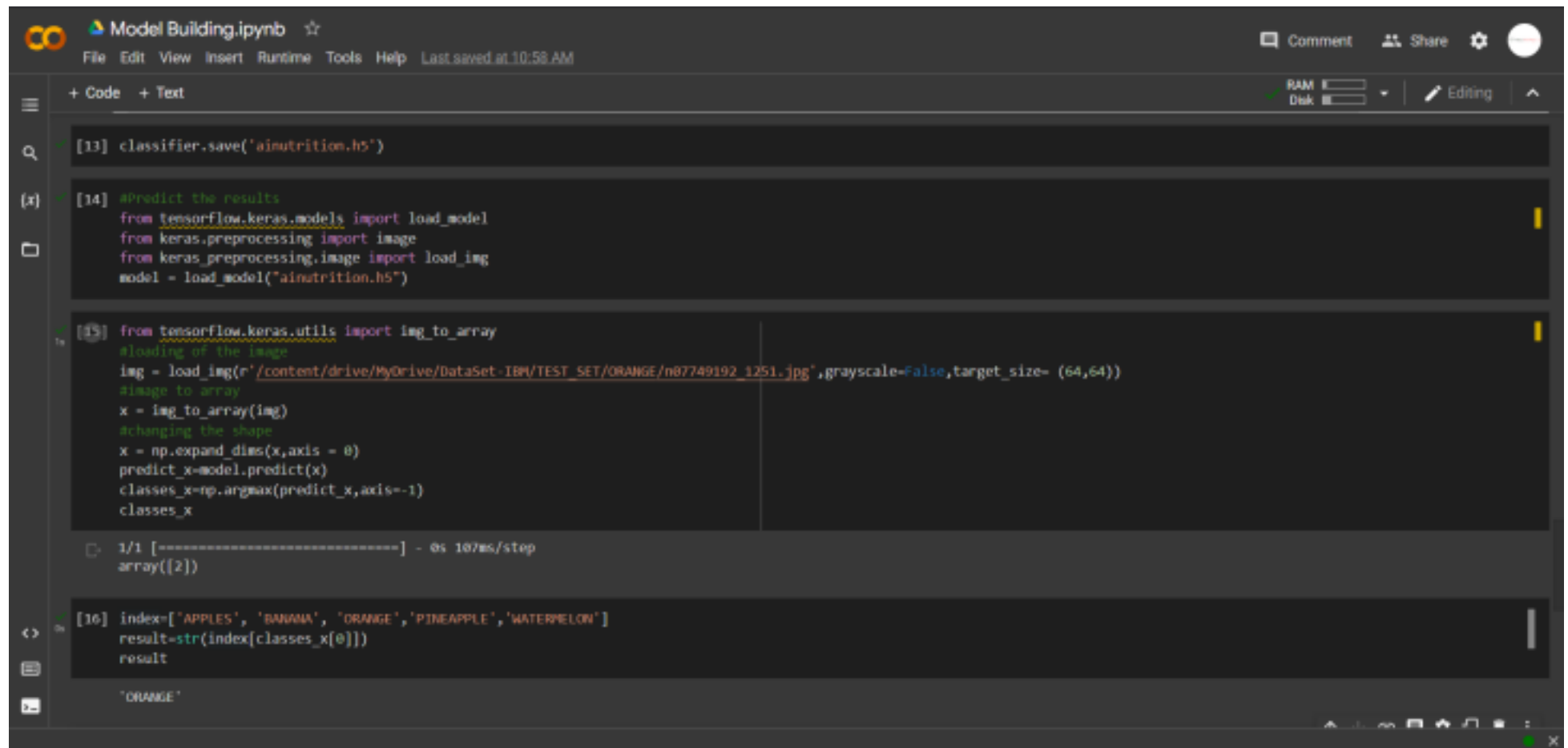
elif result == 'WATERMELON':

print("Calories: 46, Carbs: 11.5 grams, Fiber: 0.6 grams, Sugar: 9.4 grams, Protein: 0.9 grams, Fat: 0.2 grams, Vitamin A: 5% of the Daily Value (DV), Vitamin C: 14% of the DV.")

A screenshot of a terminal window with a dark background. The text is white and shows the output of a program for an orange. The first line is "ORANGE" in all caps. The second line is a single line of text: "60 calories, No fat or sodium, 3 grams of fiber, 12 grams of sugar, 1 gram of protein, 14 micrograms of vitamin A, 70 milligrams of vitamin C, 6% of your daily recommended amount". Below the text is a horizontal progress bar with a white fill and a small white square at the end. The number "3" is visible at the far right end of the bar.

```
ORANGE
60 calories, No fat or sodium, 3 grams of fiber, 12 grams of sugar, 1 gram of protein, 14 micrograms of vitamin A, 70 milligrams of vitamin C, 6% of your daily recommended amount
```

Model Building



The screenshot shows a Jupyter Notebook interface with the title "Model Building.ipynb". The top bar includes a menu (File, Edit, View, Insert, Runtime, Tools, Help) and a status bar indicating "Last saved at 10:58 AM". The notebook has three cells:

- Cell [13]: `classifier.save('ainutrition.h5')`
- Cell [14]:

```
#Predict the results
from tensorflow.keras.models import load_model
from keras.preprocessing import image
from keras_preprocessing.image import load_img
model = load_model("ainutrition.h5")
```
- Cell [15]:

```
from tensorflow.keras.utils import img_to_array
#loading of the image
img = load_img(r'/content/drive/MyDrive/DataSet-IBM/TEST_SET/ORANGE/n07749192_1251.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
```

Below the code in cell [15], the output is displayed:

```
1/1 [=====] - 0s 107ms/step
array([2])
```
- Cell [16]:

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

The output of cell [16] is:

```
'ORANGE'
```

Webpage

Know Your Food Calorie

Know live food calories & nutrition information from a single food image

Choose File n07749192_1251.jpg

Submit

Instructions:

Limitations

- The image size must be under 1024KB.
- The image format must be in JPEG, JPG or PNG.

Do's

- Center the food on the picture.
- Upload squared images, meaning that height and width are the same.

Dont's

- Blurry images.
- Images that include multiple food items.



Fruit: ORANGE

Nutrition: 60 calories, No fat or sodium, 3 grams of fiber, 12 grams of sugar, 1 gram of protein, 14 micrograms of vitamin A, 70 milligrams of vitamin C, 6% of your daily recommended amount of calcium.