Project Development Phase
Model Performance Test

Date	17 November 2022
Team ID	PNT2022TMID26295
Project Name	Fertilizer Recommendation System For Disease Prediction
Maximum Marks	10 Marks

Model Performance Testing:

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

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Total params: 36,160,485 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 - NILL Confidence Score - NILL	NILL

SCREENSHOTS :

Veg:

In [1]:
```
from keras.preprocessing.image import ImageDataGenerator
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1)
```

In [2]:
```
x_train=train_datagen.flow_from_directory(r'C:\Users\princ\OneDrive\Desktop\Dataset Plant Disease\Veg-dataset\Veg-dataset\train_set',target_size=(128,
x_test=test_datagen.flow_from_directory(r'C:\Users\princ\OneDrive\Desktop\Dataset Plant Disease\Veg-dataset\Veg-dataset\test_set',target_size=(128,128
```

```
Found 11386 images belonging to 9 classes.
Found 3416 images belonging to 9 classes.
```

In [3]:
```
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
```

In [4]:
```
model=Sequential()
```

In [5]:
```
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
```

In [6]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

In [7]:
```
model.add(Flatten())
```

In [8]:
```
model.add(Dense(units=300,kernel_initializer='uniform',activation='relu'))
```

In [8]:
```
model.add(Dense(units=300,kernel_initializer='uniform',activation='relu'))
```

In [9]:
```
model.add(Dense(units=150,kernel_initializer='uniform',activation='relu'))
```

In [10]:
```
model.add(Dense(units=75,kernel_initializer='uniform',activation='relu'))
```

In [11]:
```
model.add(Dense(units=9,kernel_initializer='uniform',activation='softmax'))
```

In [12]:
```
model.compile(loss='categorical_crossentropy',optimizer="adam",metrics=["accuracy"])
```

In [13]:
```
model.fit(x_train,steps_per_epoch=89,epochs=20,validation_data=x_test,validation_steps=27)
```

```
Epoch 1/20
89/89 [==============================] - 37s 407ms/step - loss: 1.8913 - accuracy: 0.2963 - val_loss: 201.2470 - val_accuracy: 0.3634
Epoch 2/20
89/89 [==============================] - 34s 386ms/step - loss: 1.3260 - accuracy: 0.5105 - val_loss: 144.9129 - val_accuracy: 0.5104
Epoch 3/20
89/89 [==============================] - 31s 348ms/step - loss: 1.0517 - accuracy: 0.6204 - val_loss: 452.8666 - val_accuracy: 0.2465
Epoch 4/20
89/89 [==============================] - 30s 340ms/step - loss: 0.9506 - accuracy: 0.6570 - val_loss: 1062.1256 - val_accuracy: 0.2801
Epoch 5/20
89/89 [==============================] - 33s 368ms/step - loss: 0.7732 - accuracy: 0.7268 - val_loss: 713.5864 - val_accuracy: 0.3264
Epoch 6/20
89/89 [==============================] - 33s 372ms/step - loss: 0.6780 - accuracy: 0.7574 - val_loss: 1175.1545 - val_accuracy: 0.2801
Epoch 7/20
89/89 [==============================] - 31s 352ms/step - loss: 0.6116 - accuracy: 0.7745 - val_loss: 1305.8286 - val_accuracy: 0.2743
Epoch 8/20
89/89 [==============================] - 29s 323ms/step - loss: 0.5763 - accuracy: 0.7903 - val_loss: 1302.3727 - val_accuracy: 0.2269
Epoch 9/20
89/89 [==============================] - 30s 342ms/step - loss: 0.5767 - accuracy: 0.7946 - val_loss: 1277.9509 - val_accuracy: 0.3056
```

```
Epoch 10/20
89/89 [==============================] - 31s 348ms/step - loss: 0.5556 - accuracy: 0.8013 - val_loss: 1235.4139 - val_accuracy: 0.3113
Epoch 11/20
89/89 [==============================] - 32s 355ms/step - loss: 0.4403 - accuracy: 0.8392 - val_loss: 1530.7563 - val_accuracy: 0.3461
Epoch 12/20
89/89 [==============================] - 31s 344ms/step - loss: 0.4323 - accuracy: 0.8522 - val_loss: 1574.8284 - val_accuracy: 0.3125
Epoch 13/20
89/89 [==============================] - 29s 320ms/step - loss: 0.4533 - accuracy: 0.8343 - val_loss: 1657.8656 - val_accuracy: 0.2627
Epoch 14/20
89/89 [==============================] - 29s 324ms/step - loss: 0.4202 - accuracy: 0.8498 - val_loss: 1603.1924 - val_accuracy: 0.3021
Epoch 15/20
89/89 [==============================] - 26s 289ms/step - loss: 0.3590 - accuracy: 0.8792 - val_loss: 1485.8334 - val_accuracy: 0.3623
Epoch 16/20
89/89 [==============================] - 25s 281ms/step - loss: 0.3680 - accuracy: 0.8694 - val_loss: 2185.7332 - val_accuracy: 0.2708
Epoch 17/20
89/89 [==============================] - 24s 270ms/step - loss: 0.3488 - accuracy: 0.8792 - val_loss: 2317.3721 - val_accuracy: 0.2870
Epoch 18/20
89/89 [==============================] - 24s 264ms/step - loss: 0.3258 - accuracy: 0.8902 - val_loss: 1627.2305 - val_accuracy: 0.3009
Epoch 19/20
89/89 [==============================] - 26s 293ms/step - loss: 0.3268 - accuracy: 0.8869 - val_loss: 1958.1587 - val_accuracy: 0.2546
Epoch 20/20
89/89 [==============================] - 28s 313ms/step - loss: 0.2834 - accuracy: 0.9031 - val_loss: 2081.9343 - val_accuracy: 0.3090
```

Out[13]:

In [14]:
```
model.save('vegetabledata.h5')
```

```
Epoch 19/20
89/89 [==============================] - 26s 293ms/step - loss: 0.3268 - accuracy: 0.8869 - val_loss: 1958.1587 - val_accuracy: 0.2546
Epoch 20/20
89/89 [==============================] - 28s 313ms/step - loss: 0.2834 - accuracy: 0.9031 - val_loss: 2081.9343 - val_accuracy: 0.3090
```

Out[13]:

In [14]:
```
model.save('vegetabledata.h5')
```

In [15]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 126, 126, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 63, 63, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 127008)            0
_________________________________________________________________
dense (Dense)                (None, 300)               38102700
_________________________________________________________________
dense_1 (Dense)              (None, 150)               45150
_________________________________________________________________
dense_2 (Dense)              (None, 75)                11325
_________________________________________________________________
dense_3 (Dense)              (None, 9)                 684
=================================================================
Total params: 38,160,755
Trainable params: 38,160,755
Non-trainable params: 0
_________________________________________________________________
```

Fruit:

In [1]:
```
from keras.preprocessing.image import ImageDataGenerator
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1)
```

In [2]:
```
x_train=train_datagen.flow_from_directory(r'C:\Users\princ\OneDrive\Desktop\Dataset Plant Disease\fruit-dataset\fruit-dataset\train',target_size=(128,
x_test=test_datagen.flow_from_directory(r'C:\Users\princ\OneDrive\Desktop\Dataset Plant Disease\fruit-dataset\fruit-dataset\test',target_size=(128,128
```

```
Found 5384 images belonging to 6 classes.
Found 1686 images belonging to 6 classes.
```

In [3]:
```
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
```

In [4]:
```
model=Sequential()
```

In [5]:
```
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
```

In [6]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

In [7]:
```
model.add(Flatten())
```

In [8]:
```
model.add(Dense(units=40,kernel_initializer='uniform',activation='relu'))
model.add(Dense(units=70,kernel_initializer='random_uniform',activation='relu'))
model.add(Dense(units=6,kernel_initializer='random_uniform',activation='softmax'))
```

In [9]:
```
model.compile(loss='categorical_crossentropy',optimizer="adam",metrics=["accuracy"])
```

In [10]:
```
model.fit(x_train,steps_per_epoch=168,epochs=3,validation_data=x_test,validation_steps=52)
```

```
Epoch 1/3
168/168 [==============================] - 78s 460ms/step - loss: 0.8677 - accuracy: 0.6741 - val_loss: 62.5237 - val_accuracy: 0.8191
Epoch 2/3
168/168 [==============================] - 28s 165ms/step - loss: 0.3775 - accuracy: 0.8707 - val_loss: 86.6816 - val_accuracy: 0.8071
Epoch 3/3
168/168 [==============================] - 28s 166ms/step - loss: 0.2731 - accuracy: 0.9088 - val_loss: 231.7529 - val_accuracy: 0.7362
```

Out[10]:

In [11]:
```
model.save('fruitdata.h5')
```

In [12]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 126, 126, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 63, 63, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 127008)            0
_________________________________________________________________
dense (Dense)                (None, 40)                5080360
_________________________________________________________________
dense_1 (Dense)              (None, 70)                2870
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 426
=================================================================
Total params: 5,084,552
Trainable params: 5,084,552
Non-trainable params: 0
_________________________________________________________________
```