Assessment

In [50]:
```
from tensorflow import keras

base_model = keras.applications.VGG16(
    weights='imagenet',
    input_shape=(224, 224, 3),
    include_top=False)
```

In [51]:
```
base_model.summary()
```

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_10 (InputLayer)        [(None, 224, 224, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
```

Freeze Base Model

In [52]:

```
# Freeze base model
base_model.trainable = False
```

Add Layers to Model

In [53]:

```
# Create inputs with correct shape
inputs = keras.Input(shape=(224, 224, 3))

x = base_model(inputs, training=False)
```

```
# Add pooling layer or flatten layer
x = keras.layers.GlobalAveragePooling2D()(x)

# Add final dense layer
outputs = keras.layers.Dense(6, activation = 'softmax')(x)

# Combine inputs and outputs to create model
model = keras.Model(inputs, outputs)
```

In [54]:

```
model.summary()
```

```
Model: "model_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_11 (InputLayer)        [(None, 224, 224, 3)]     0
_________________________________________________________________
vgg16 (Model)                (None, 7, 7, 512)         14714688
_________________________________________________________________
global_average_pooling2d_6 ( (None, 512)               0
_________________________________________________________________
dense_5 (Dense)              (None, 6)                 3078
=================================================================
Total params: 14,717,766
Trainable params: 3,078
Non-trainable params: 14,714,688
_________________________________________________________________
```

Compile Model

In [55]:

```
model.compile(loss=keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=[keras.metrics.BinaryAccuracy()])
```

Augment the Data

In [56]:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
transformation_ratio = 0.05
datagen = ImageDataGenerator(rescale=1. / 255,
                             validation_split = 0.2,
                             rotation_range=transformation_ratio,
                             shear_range=transformation_ratio,
                             zoom_range=transformation_ratio,
                             cval=transformation_ratio,
                             horizontal_flip=True,
                             vertical_flip=True)
```

Load Dataset

In [57]:

```
# load and iterate training dataset
train_it = datagen.flow_from_directory('data/fruits/train/',
                                       target_size=(224, 224),
                                       color_mode='rgb',
                                       class_mode="categorical")
# load and iterate validation dataset
```

```
valid_it = datagen.flow_from_directory('data/fruits/valid/',
                                       target_size=(224, 224),
                                       color_mode='rgb',
                                       class_mode="categorical")
```

```
Found 1182 images belonging to 6 classes.
Found 329 images belonging to 6 classes.
```

Train the Model

In [58]:

```
model.fit(train_it,
          validation_data=valid_it,
          steps_per_epoch=train_it.samples/train_it.batch_size,
          validation_steps=valid_it.samples/valid_it.batch_size,
          epochs=16)
```

```
Epoch 1/16
37/36 [==============================] - 19s 501ms/step - loss: 0.7465 - binary_acc
uracy: 0.8376 - val_loss: 0.7406 - val_binary_accuracy: 0.8399
Epoch 2/16
37/36 [==============================] - 18s 499ms/step - loss: 0.7371 - binary_acc
uracy: 0.8481 - val_loss: 0.7311 - val_binary_accuracy: 0.8566
Epoch 3/16
37/36 [==============================] - 19s 502ms/step - loss: 0.7293 - binary_acc
uracy: 0.8589 - val_loss: 0.7247 - val_binary_accuracy: 0.8708
Epoch 4/16
37/36 [==============================] - 18s 496ms/step - loss: 0.7221 - binary_acc
uracy: 0.8766 - val_loss: 0.7180 - val_binary_accuracy: 0.8779
Epoch 5/16
37/36 [==============================] - 19s 501ms/step - loss: 0.7166 - binary_acc
uracy: 0.8837 - val_loss: 0.7113 - val_binary_accuracy: 0.8911
Epoch 6/16
37/36 [==============================] - 18s 498ms/step - loss: 0.7108 - binary_acc
uracy: 0.8921 - val_loss: 0.7065 - val_binary_accuracy: 0.8982
Epoch 7/16
37/36 [==============================] - 18s 496ms/step - loss: 0.7067 - binary_acc
uracy: 0.8937 - val_loss: 0.7010 - val_binary_accuracy: 0.9032
Epoch 8/16
37/36 [==============================] - 18s 496ms/step - loss: 0.7017 - binary_acc
uracy: 0.9023 - val_loss: 0.7003 - val_binary_accuracy: 0.8982
Epoch 9/16
37/36 [==============================] - 18s 498ms/step - loss: 0.6981 - binary_acc
uracy: 0.9085 - val_loss: 0.6946 - val_binary_accuracy: 0.9134
Epoch 10/16
37/36 [==============================] - 18s 499ms/step - loss: 0.6942 - binary_acc
uracy: 0.9172 - val_loss: 0.6945 - val_binary_accuracy: 0.9119
Epoch 11/16
37/36 [==============================] - 18s 490ms/step - loss: 0.6915 - binary_acc
uracy: 0.9220 - val_loss: 0.6892 - val_binary_accuracy: 0.9205
Epoch 12/16
37/36 [==============================] - 18s 499ms/step - loss: 0.6886 - binary_acc
uracy: 0.9288 - val_loss: 0.6876 - val_binary_accuracy: 0.9245
Epoch 13/16
37/36 [==============================] - 19s 500ms/step - loss: 0.6855 - binary_acc
uracy: 0.9349 - val_loss: 0.6847 - val_binary_accuracy: 0.9331
Epoch 14/16
37/36 [==============================] - 18s 499ms/step - loss: 0.6834 - binary_acc
uracy: 0.9392 - val_loss: 0.6832 - val_binary_accuracy: 0.9372
Epoch 15/16
37/36 [==============================] - 19s 502ms/step - loss: 0.6813 - binary_acc
uracy: 0.9423 - val_loss: 0.6818 - val_binary_accuracy: 0.9417
Epoch 16/16
37/36 [==============================] - 18s 499ms/step - loss: 0.6796 - binary_acc
uracy: 0.9459 - val_loss: 0.6775 - val_binary_accuracy: 0.9504
```

Out[58]: `<tensorflow.python.keras.callbacks.History at 0x7efda00c8240>`

Unfreeze Model for Fine Tuning

In [59]:

```
# Unfreeze the base model
base_model.trainable = True

# Compile the model with a low learning rate
model.compile(optimizer=keras.optimizers.RMSprop(learning_rate = .00001),  # Very
              loss=keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=[keras.metrics.BinaryAccuracy()])
```

In [61]:
```
model.fit(train_it,
          validation_data=valid_it,
          steps_per_epoch=train_it.samples/train_it.batch_size,
          validation_steps=valid_it.samples/valid_it.batch_size,
          epochs=12)
```

```
Epoch 1/12
37/36 [==============================] - 20s 533ms/step - loss: 0.6325 - binary_acc
uracy: 0.9955 - val_loss: 0.6337 - val_binary_accuracy: 0.9939
Epoch 2/12
37/36 [==============================] - 20s 534ms/step - loss: 0.6320 - binary_acc
uracy: 0.9966 - val_loss: 0.6338 - val_binary_accuracy: 0.9919
Epoch 3/12
37/36 [==============================] - 20s 529ms/step - loss: 0.6318 - binary_acc
uracy: 0.9966 - val_loss: 0.6339 - val_binary_accuracy: 0.9939
Epoch 4/12
37/36 [==============================] - 20s 545ms/step - loss: 0.6315 - binary_acc
uracy: 0.9969 - val_loss: 0.6332 - val_binary_accuracy: 0.9949
Epoch 5/12
37/36 [==============================] - 20s 527ms/step - loss: 0.6308 - binary_acc
uracy: 0.9983 - val_loss: 0.6335 - val_binary_accuracy: 0.9929
Epoch 6/12
37/36 [==============================] - 20s 531ms/step - loss: 0.6317 - binary_acc
uracy: 0.9968 - val_loss: 0.6330 - val_binary_accuracy: 0.9949
Epoch 7/12
37/36 [==============================] - 20s 542ms/step - loss: 0.6300 - binary_acc
uracy: 0.9997 - val_loss: 0.6336 - val_binary_accuracy: 0.9919
Epoch 8/12
37/36 [==============================] - 20s 529ms/step - loss: 0.6316 - binary_acc
uracy: 0.9969 - val_loss: 0.6320 - val_binary_accuracy: 0.9980
Epoch 9/12
37/36 [==============================] - 19s 526ms/step - loss: 0.6309 - binary_acc
uracy: 0.9980 - val_loss: 0.6344 - val_binary_accuracy: 0.9889
Epoch 10/12
37/36 [==============================] - 20s 532ms/step - loss: 0.6308 - binary_acc
uracy: 0.9989 - val_loss: 0.6367 - val_binary_accuracy: 0.9868
Epoch 11/12
37/36 [==============================] - 20s 536ms/step - loss: 0.6304 - binary_acc
uracy: 0.9992 - val_loss: 0.6330 - val_binary_accuracy: 0.9949
Epoch 12/12
37/36 [==============================] - 20s 544ms/step - loss: 0.6307 - binary_acc
uracy: 0.9986 - val_loss: 0.6352 - val_binary_accuracy: 0.9909
```

Out[61]:
```
<tensorflow.python.keras.callbacks.History at 0x7efec1175710>
```

In [62]:
```
model.evaluate(valid_it, steps=valid_it.samples/valid_it.batch_size)
```

```
11/10 [==============================] - 4s 323ms/step - loss: 0.6368 - binary_ac
curacy: 0.9868
```

Out[62]:
```
[0.6368132829666138, 0.9868288636207581]
```

Run the Assessment

In [63]:
```
from run_assessment import run_assessment
```

In [64]:
```
run_assessment(model, valid_it)
```

```
Evaluating model 5 times to obtain average accuracy...

11/10 [==============================] - 4s 326ms/step - loss: 0.6352 - binary_ac
curacy: 0.9883
11/10 [==============================] - 4s 327ms/step - loss: 0.6365 - binary_ac
curacy: 0.9873
11/10 [==============================] - 4s 362ms/step - loss: 0.6357 - binary_ac
curacy: 0.9889
11/10 [==============================] - 4s 320ms/step - loss: 0.6362 - binary_ac
curacy: 0.9894
11/10 [==============================] - 4s 336ms/step - loss: 0.6360 - binary_ac
curacy: 0.9899

Accuracy required to pass the assessment is 0.92 or greater.
Your average accuracy is 0.9888.

Congratulations! You passed the assessment!
See instructions below to generate a certificate.
```