Model Building

Pre-trained CNN Model as feature extractor:

MODEL BUILDING

Pre-Training CNN Model As A Feature Extractor

In [17]:
```
xception = Xception(input_shape = imageSize + [3],weights='imagenet',include_top = False)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/xception/xception_
weights_tf_dim_ordering_tf_kernels_notop.h5
83683744/83683744 [==============================] - 4s 0us/step
```

In [18]:
```
for layer in xception.layers:
  layer.trainable = False
```

In [19]:
```
x = Flatten()(xception.output)
```

Adding Dense Layers:

Adding Dense Layers

In [20]:
```
prediction = Dense( 5,activation ='softmax')(x)
```

In [21]:
```
model = Model(inputs=xception.input,outputs=prediction)
```

In [22]:
```
model.summary()
Model : "model"
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
=================================================================
 input_1 (InputLayer)        [(None, 299, 299, 3       0         []
                             )]

 block1_conv1 (Conv2D)       (None, 149, 149, 32       864       ['input_1[0][0]']
                             )

 block1_conv1_bn (BatchNormaliz  (None, 149, 149, 32   128       ['block1_conv1[0][0]']
 ation)                      )

 block1_conv1_act (Activation)  (None, 149, 149, 32    0         ['block1_conv1_bn[0][0]']
```

Configure the Learning Process:

Configuring The Learning Process

n [23]:

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

Train the Model:

Training The Model

In [24]:

```
# fit the model

r = model.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=30,
  steps_per_epoch=len (training_set)//32,
  validation_steps=len(test_set)//32
)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:8: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.
Epoch 1/30
3/3 [==============================] - 58s 17s/step - loss: 14.1287 - accuracy: 0.3438
Epoch 2/30
3/3 [==============================] - 48s 14s/step - loss: 7.1767 - accuracy: 0.5729
Epoch 3/30
3/3 [==============================] - 47s 14s/step - loss: 10.7616 - accuracy: 0.3125
Epoch 4/30
3/3 [==============================] - 40s 12s/step - loss: 7.0867 - accuracy: 0.4615
Epoch 5/30
3/3 [==============================] - 48s 15s/step - loss: 10.9142 - accuracy: 0.5729
Epoch 6/30
3/3 [==============================] - 50s 16s/step - loss: 6.9483 - accuracy: 0.6667
Epoch 7/30
3/3 [==============================] - 48s 14s/step - loss: 4.2671 - accuracy: 0.6562
Epoch 8/30
3/3 [==============================] - 48s 14s/step - loss: 10.7949 - accuracy: 0.4896
Epoch 9/30
3/3 [==============================] - 50s 16s/step - loss: 3.1253 - accuracy: 0.6875
Epoch 10/30
3/3 [==============================] - 49s 15s/step - loss: 5.1989 - accuracy: 0.6146
Epoch 11/30
3/3 [==============================] - 48s 14s/step - loss: 6.4308 - accuracy: 0.6771
Epoch 12/30
3/3 [==============================] - 47s 14s/step - loss: 3.4153 - accuracy: 0.7083
Epoch 13/30
3/3 [==============================] - 48s 15s/step - loss: 3.2298 - accuracy: 0.6771
Epoch 14/30
3/3 [==============================] - 50s 16s/step - loss: 3.7171 - accuracy: 0.6979
Epoch 15/30
3/3 [==============================] - 50s 15s/step - loss: 5.5327 - accuracy: 0.5938
Epoch 16/30
3/3 [==============================] - 47s 15s/step - loss: 3.1135 - accuracy: 0.7083
Epoch 17/30
3/3 [==============================] - 48s 14s/step - loss: 4.7249 - accuracy: 0.7188
Epoch 18/30
```

Save the Model:

Saving The Model

In [25]:
```
model.save("diabetic.h5")
```