FLIP ROBO

Image Scraping & Classification

Submitted by:

PRERNA SHARMA

ACKNOWLEDGMENT

I take this opportunity to acknowledge everyone who have helped me in every stage of this project.

Firstly, I am indebtedly grateful to my SME MR. Sajid Chowdary, MR. Keshav Bansal sir, Miss. Sapna Verma who helped me from beginning of my Projects. Am also thankful to my Mentor Shankar Gowda Sir and my whole Data Trained team, where I have learnt Analysing the datasets and building the models using Machine learning and making the projects. Finally, am so thankful to my Flip Robo Technologies team, as they provided me the opportunity to work as intern in their company.

I feel pleasure, to make project report on “Image Scarping & Classification”. It has been my privilege to have a team of project guide who have assisted me from the commencement of this project. The project is a result of my hard work, and determination put on by me with the help of Web-Scraping Techniques, searched on google for the information and watched youtube videos on deep learning model building of krish Naik.

INTRODUCTION

Business Problem Framing

Image classification is a fascinating deep learning project. Specifically, image classification comes under the computer vision project category.

The Image classification problem is to categorize all the pixels of a digital image into one of the defined classes. It is the most critical use case in digital image analysis. It is an application of both supervised classification and unsupervised classification.

Images are one of the major sources of data in the field of data science and AI. This field is making appropriate use of information that can be gathered through images by examining its features and details. In this I need to scrape the images from ecommerce website.

. The clothing categories used for scraping are:

- Sarees (women)
- Trousers (men)
- Jeans (men)

Conceptual Background of the Domain Problem

Image classification is a complex procedure which relies on different components. Here, some of the presented strategies, issues and additional prospects of image orders are addressed. The primary spotlight will be on cutting edge classification methods which are utilized for enhancing characterization precision.

Deep learning (DL) is a sub field to the machine learning, capable of learning through its own method of computing. A deep learning model is introduced to persistently break down information with a homogeneous structure like how a human would make determinations. To accomplish this, deep learning utilizes a layered structure of several algorithms expressed as an artificial neural system (ANN). The architecture of an ANN is simulated with the help of the biological neural network of the human brain. This makes the deep learning most capable than the standard machine learning models. In deep neural networks every node decides its basic inputs by itself and sends it to the next tier on behalf of the previous tier. We train the data in the networks by giving an input image and conveying the network about its output. Neural networks are expressed in terms of number of layers involved for producing the inputs and outputs and the depth of the neural network.

Review of Literature

Classification is a systematic arrangement in groups and categories based on its features. Image classification came into existence for decreasing the gap between the computer vision and human vision by training the computer with the data. The image classification is achieved by differentiating the image into the prescribed category based on the content of the vision. We use Deep Learning to accomplish the task of image Classification. In this project we have to classify whether is image is of a Jeans, a trouser or a saree.

Motivation for the Problem Undertaken

I have worked on various Machine learning projects and NLP projects but I have ever worked on deep learning models. So, in order to implement end to end projects in the field of Data science. I choose image classification project to implement deep learning models.

Analytical Problem Framing

Mathematical/ Analytical Modelling of the Problem

I have scraped different images from amazon of 3 classes they are men jeans, men trousers and women sarees and built our model by training it on the image data. I have used RESNET50 model for getting better results.

Data Sources and their formats

Data has been scraped from amazon.com using a python script which with selenium. All the data is in the .jpg image format. I have over 300 images per class.

men jeans
men jeans_1
men jeans_10
men jeans_1
ıen jeans_102
men jeans_103
men jeans_104
men jeans_1(

sarees_1
sarees_2
sarees_10
sarees_20
sarees_21
sarees_22
sarees_112
sarees_113
sarees_114

Data Pre-processing Done

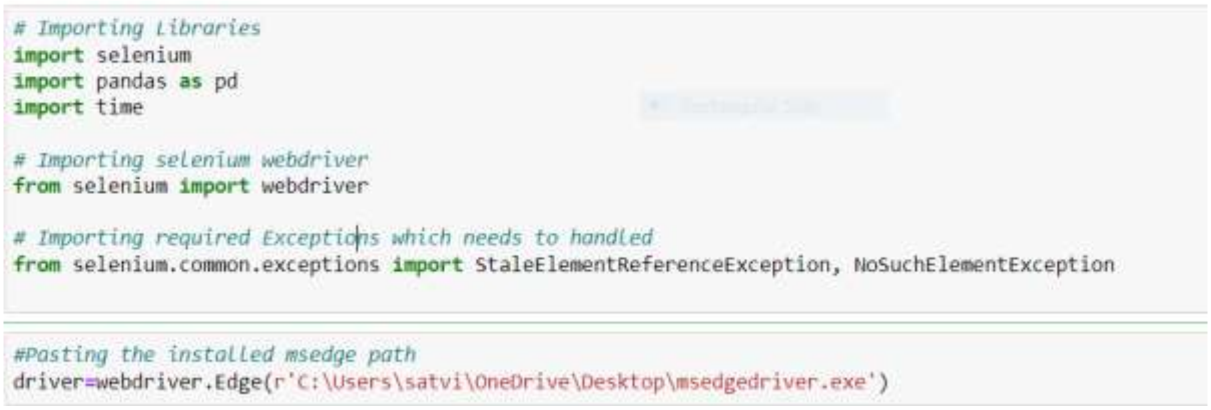
I have first labelled the image data and then manually removed irrelevant images that were downloading with the script and removed the duplicate images. I have also performed multiple data augmentation techniques in order to train the model and get multiple angles and orientation of the same image.

Data Inputs- Logic- Output Relationships

I have given raw cleaned images to the machine learning model and the output produced is a label of the image on which the model is predicted.

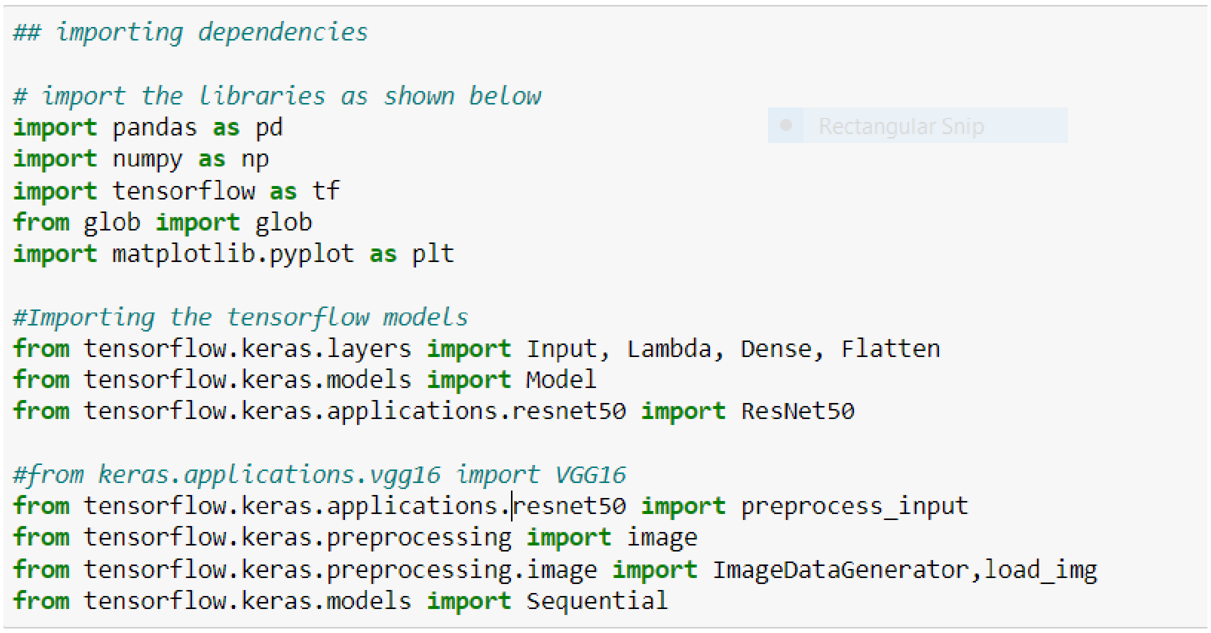
Hardware and Software Requirements and Tools Used

I have used my laptop, wed server, micro-soft edge, Jupiter Notebook which is having GUI interface. Imported necessary libraries from python such as pandas, NumPy, seaborn, matplotlib, then imported the required deep learning models using tensorflow, Keras.



```
# Importing Libraries
import selenium
import pandas as pd
import time

# Importing selenium webdriver
from selenium import webdriver

# Importing required Exceptions which needs to handled
from selenium.common.exceptions import StaleElementReferenceException, NoSuchElementException
```

```
#Pasting the installed msedge path
driver=webdriver.Edge(r'C:\Users\satvi\OneDrive\Desktop\msedgedriver.exe')
```

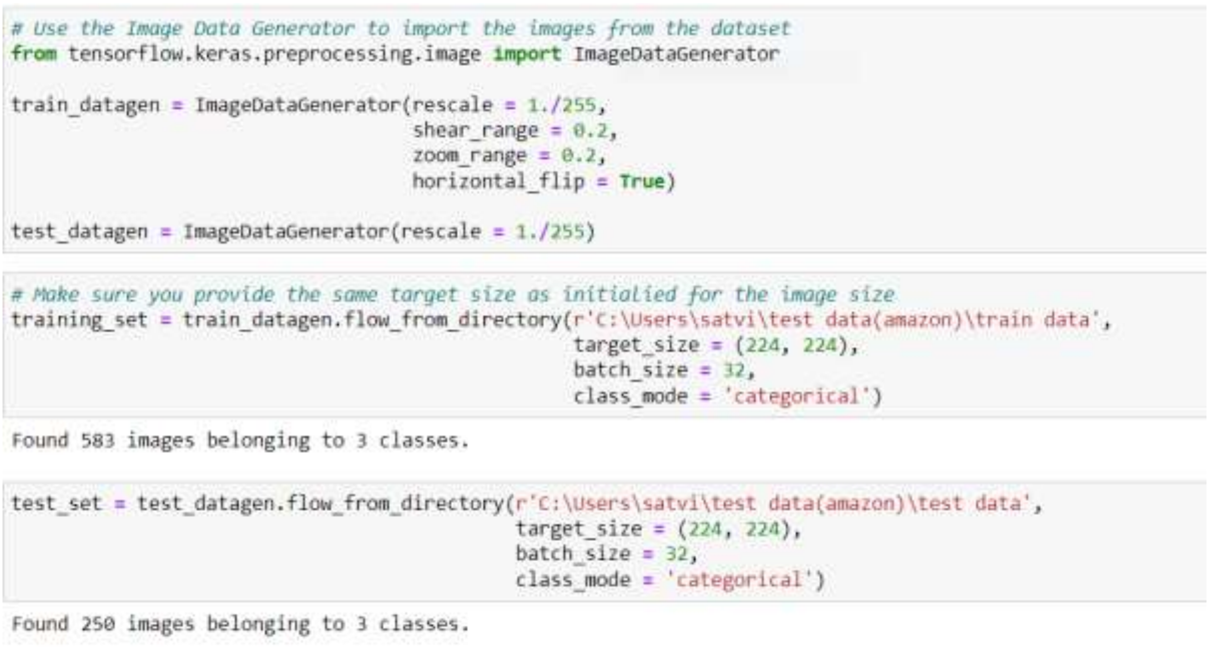


```python
## importing dependencies

# import the libraries as shown below
import pandas as pd
import numpy as np
import tensorflow as tf
from glob import glob
import matplotlib.pyplot as plt

#Importing the tensorflow models
from tensorflow.keras.layers import Input, Lambda, Dense, Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.applications.resnet50 import ResNet50

#from keras.applications.vgg16 import VGG16
from tensorflow.keras.applications.resnet50 import preprocess_input
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.models import Sequential
```

Model/s Development and Evaluation

Testing of Identified Approaches (Algorithms)

I have first scraped the data from amazon and then cleaned the data manually to train the model and then split the data into train and test set for model evaluation. I used ResNet50, for model building and performed multiple data augmentation techniques on the images for better generalization of our model and checked, identity an optimal batch size and number of epochs to train the model using trial and error method also keeping the epoch loss for both training set and test set in mind.



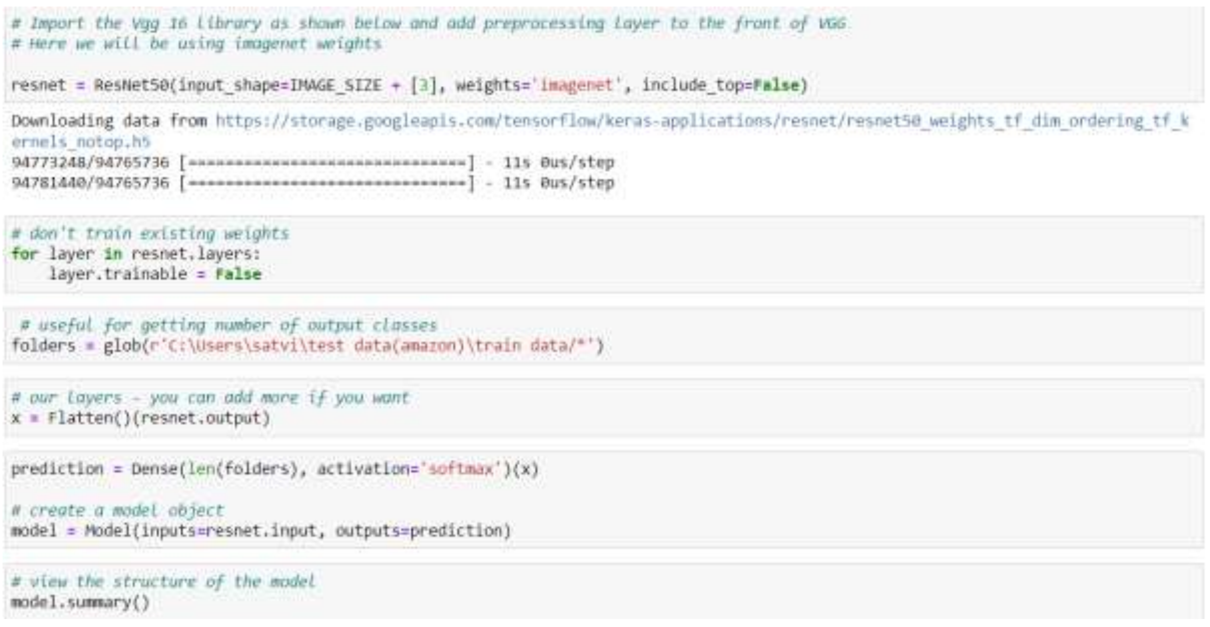
```
# Use the Image Data Generator to import the images from the dataset
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

test_datagen = ImageDataGenerator(rescale = 1./255)
```

```
# Make sure you provide the same target size as initialied for the image size
training_set = train_datagen.flow_from_directory(r'C:\Users\satvi\test data(amazon)\train data',
                                                 target_size = (224, 224),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
```

```
Found 583 images belonging to 3 classes.
```

```
test_set = test_datagen.flow_from_directory(r'C:\Users\satvi\test data(amazon)\test data',
                                            target_size = (224, 224),
                                            batch_size = 32,
                                            class_mode = 'categorical')
```

```
Found 250 images belonging to 3 classes.
```



```
# Import the Vgg 16 library as shown below and add preprocessing layer to the front of VGG
# Here we will be using imagenet weights

resnet = ResNet50(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weights_tf_dim_ordering_tf_k
ernels_notop.h5
94773248/94765736 [==============================] - 11s 0us/step
94781440/94765736 [==============================] - 11s 0us/step
```

```
# don't train existing weights
for layer in resnet.layers:
    layer.trainable = False
```

```
 # useful for getting number of output classes
folders = glob(r'C:\Users\satvi\test data(amazon)\train data/*')
```

```
# our layers - you can add more if you want
x = Flatten()(resnet.output)
```

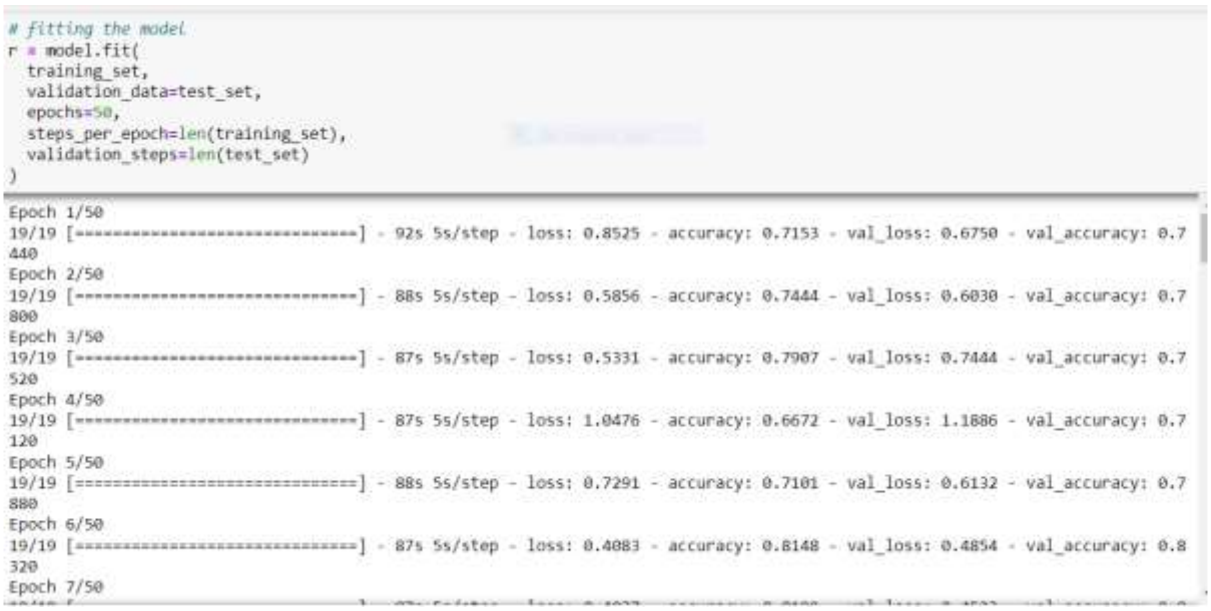
```
prediction = Dense(len(folders), activation='softmax')(x)

# create a model object
model = Model(inputs=resnet.input, outputs=prediction)
```

```
# view the structure of the model
model.summary()
```

```
# view the structure of the model
model.summary()
```

```
Model: "model"

__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_1 (InputLayer)           [(None, 224, 224, 3  0           []
                                )]

 conv1_pad (ZeroPadding2D)      (None, 230, 230, 3)  0           ['input_1[0][0]']

 conv1_conv (Conv2D)            (None, 112, 112, 64  9472        ['conv1_pad[0][0]']
                                )

 conv1_bn (BatchNormalization)  (None, 112, 112, 64  256         ['conv1_conv[0][0]']
                                )

 conv1_relu (Activation)        (None, 112, 112, 64  0           ['conv1_bn[0][0]']
                                )

 pool1_pad (ZeroPadding2D)      (None, 114, 114, 64  0           ['conv1_relu[0][0]']
```

Run and Evaluate selected models



```
# fitting the model
r = model.fit(
  training_set,
  validation_data=test_set,
  epochs=50,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
```

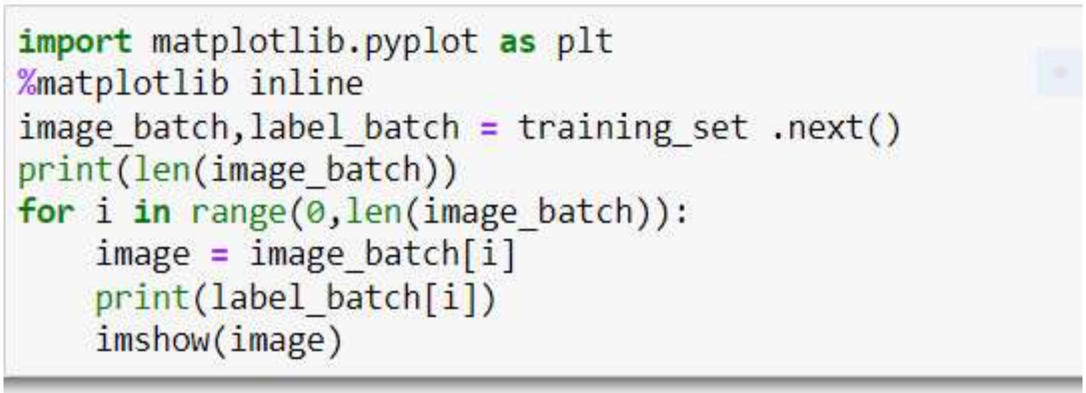
```
Epoch 1/50
19/19 [==============================] - 92s 5s/step - loss: 0.8525 - accuracy: 0.7153 - val_loss: 0.6750 - val_accuracy: 0.7
440
Epoch 2/50
19/19 [==============================] - 88s 5s/step - loss: 0.5856 - accuracy: 0.7444 - val_loss: 0.6030 - val_accuracy: 0.7
800
Epoch 3/50
19/19 [==============================] - 87s 5s/step - loss: 0.5331 - accuracy: 0.7907 - val_loss: 0.7444 - val_accuracy: 0.7
520
Epoch 4/50
19/19 [==============================] - 87s 5s/step - loss: 1.0476 - accuracy: 0.6672 - val_loss: 1.1886 - val_accuracy: 0.7
120
Epoch 5/50
19/19 [==============================] - 88s 5s/step - loss: 0.7291 - accuracy: 0.7101 - val_loss: 0.6132 - val_accuracy: 0.7
880
Epoch 6/50
19/19 [==============================] - 87s 5s/step - loss: 0.4083 - accuracy: 0.8148 - val_loss: 0.4854 - val_accuracy: 0.8
320
Epoch 7/50
```

Key Metrics for success in solving problem under consideration

I have considered the model accuracy and loss for both the training and validation data

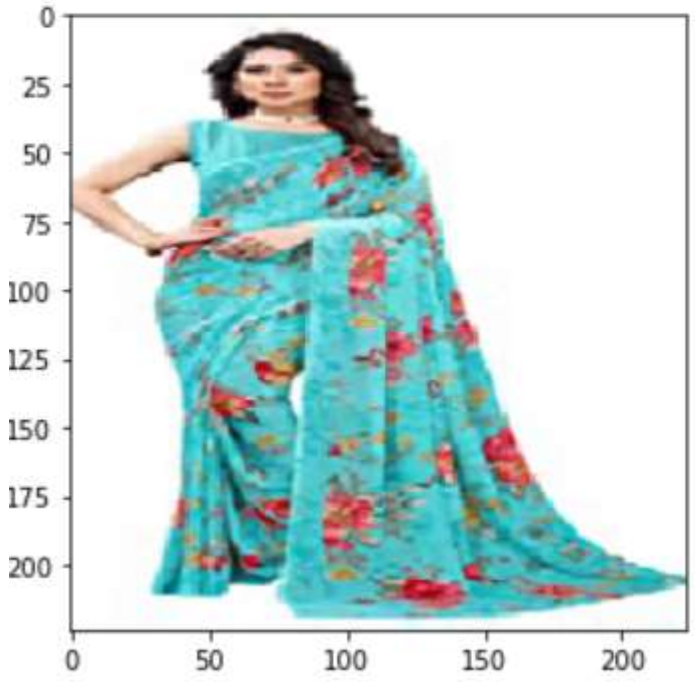
```
# tell the model what cost and optimization method to use
model.compile(
  loss='categorical_crossentropy',
  optimizer='adam',
  metrics=['accuracy']
)
```

Visualizations



```python
import matplotlib.pyplot as plt
%matplotlib inline
image_batch,label_batch = training_set .next()
print(len(image_batch))
for i in range(0,len(image_batch)):
    image = image_batch[i]
    print(label_batch[i])
    imshow(image)
```

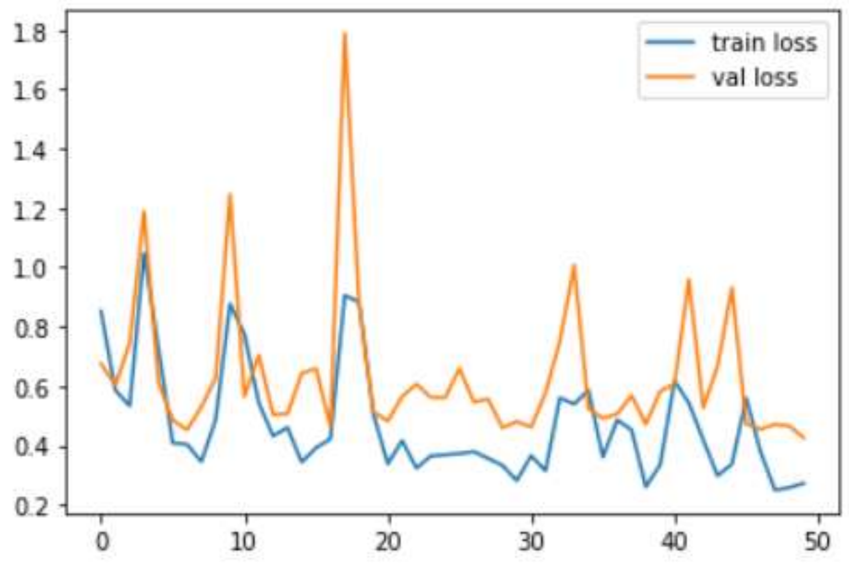
```
32
[1. 0. 0.]
```

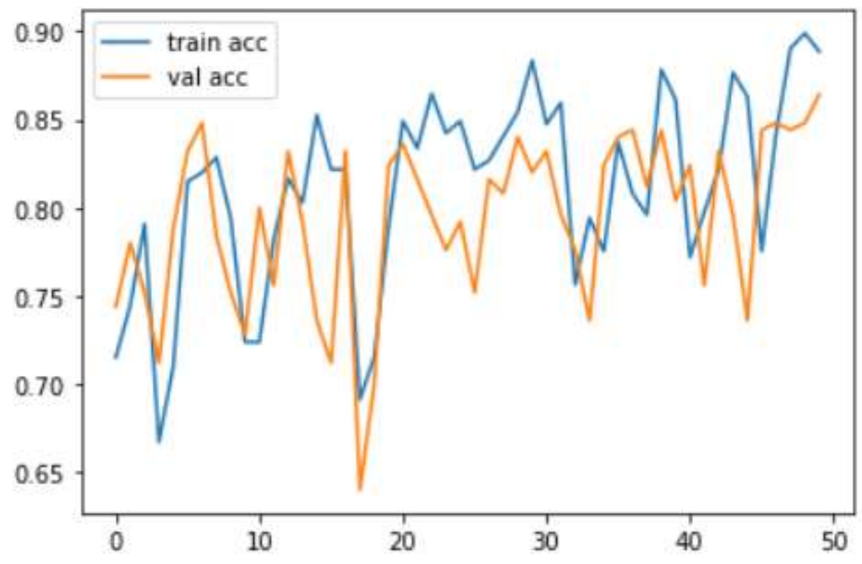


0
25
50
75
100
125
150
175
200
0
50
100
150
200

0
25
50
75
100
125
150
175
200
0
50
100
150
200

Interpretation of the Results

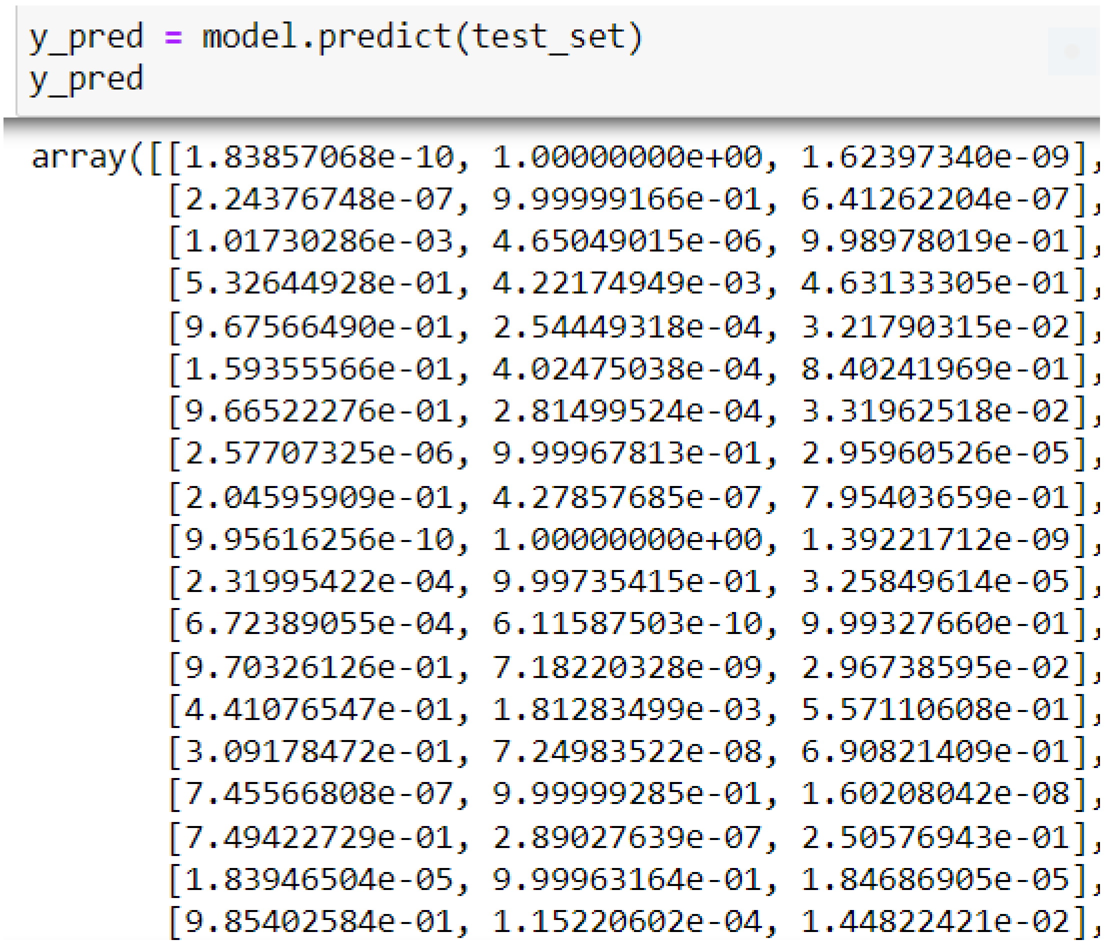




```
<Figure size 432x288 with 0 Axes>
```

CONCLUSIONS

Key Findings and Conclusions of the Study

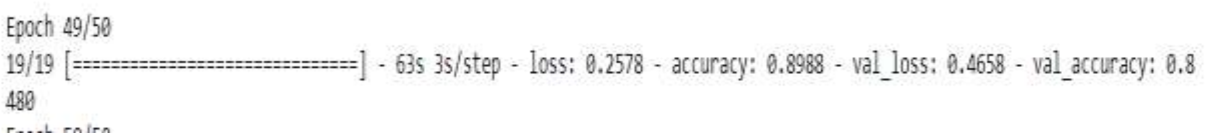


```
y_pred = model.predict(test_set)
y_pred
```

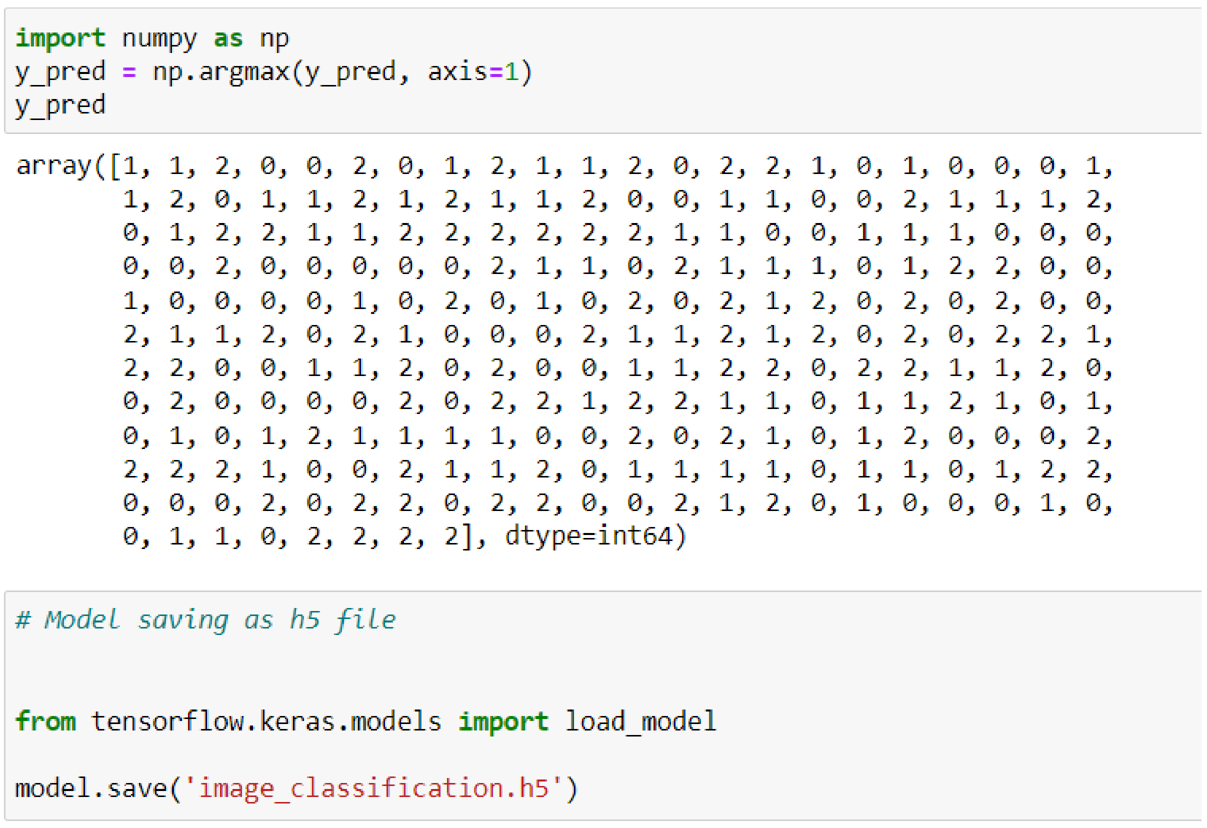
```
array([[1.83857068e-10, 1.00000000e+00, 1.62397340e-09],
       [2.24376748e-07, 9.99999166e-01, 6.41262204e-07],
       [1.01730286e-03, 4.65049015e-06, 9.98978019e-01],
       [5.32644928e-01, 4.22174949e-03, 4.63133305e-01],
       [9.67566490e-01, 2.54449318e-04, 3.21790315e-02],
       [1.59355566e-01, 4.02475038e-04, 8.40241969e-01],
       [9.66522276e-01, 2.81499524e-04, 3.31962518e-02],
       [2.57707325e-06, 9.99967813e-01, 2.95960526e-05],
       [2.04595909e-01, 4.27857685e-07, 7.95403659e-01],
       [9.95616256e-10, 1.00000000e+00, 1.39221712e-09],
       [2.31995422e-04, 9.99735415e-01, 3.25849614e-05],
       [6.72389055e-04, 6.11587503e-10, 9.99327660e-01],
       [9.70326126e-01, 7.18220328e-09, 2.96738595e-02],
       [4.41076547e-01, 1.81283499e-03, 5.57110608e-01],
       [3.09178472e-01, 7.24983522e-08, 6.90821409e-01],
       [7.45566808e-07, 9.99999285e-01, 1.60208042e-08],
       [7.49422729e-01, 2.89027639e-07, 2.50576943e-01],
       [1.83946504e-05, 9.99963164e-01, 1.84686905e-05],
       [9.85402584e-01, 1.15220602e-04, 1.44822421e-02],
```

Conclusions on our model building

Our model was successfully to classify the images with an accuracy of 89.8% Percent on a test set of around 250 images.



```
Epoch 49/50
19/19 [==============================] - 63s 3s/step - loss: 0.2578 - accuracy: 0.8988 - val_loss: 0.4658 - val_accuracy: 0.8
480
```



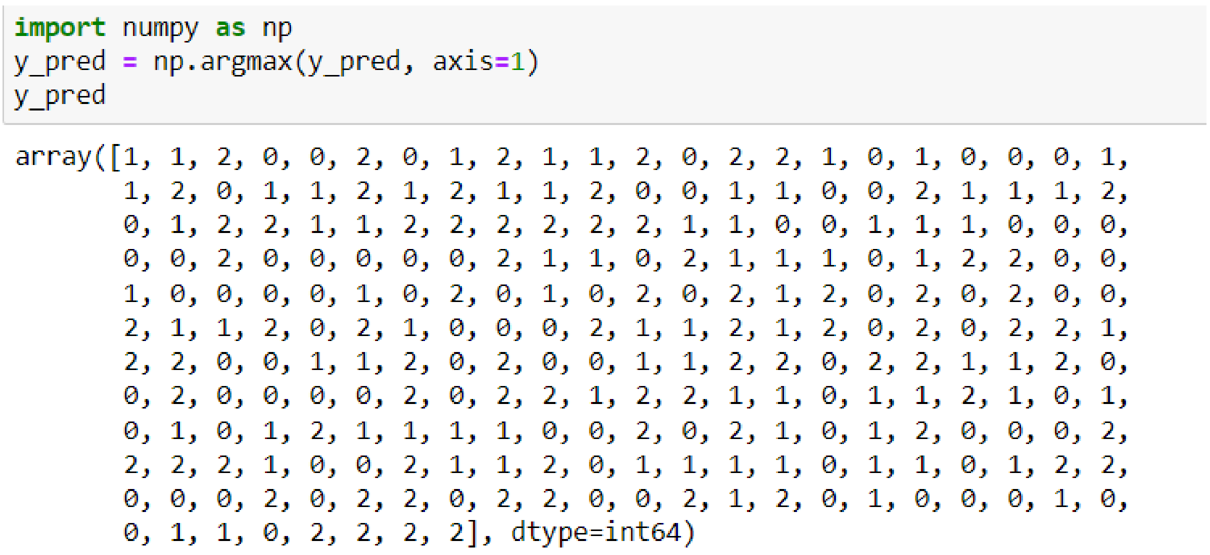
```python
import numpy as np
y_pred = np.argmax(y_pred, axis=1)
y_pred
```

```
array([1, 1, 2, 0, 0, 2, 0, 1, 2, 1, 1, 2, 0, 2, 2, 1, 0, 1, 0, 0, 0, 1,
       1, 2, 0, 1, 1, 2, 1, 2, 1, 1, 2, 0, 0, 1, 1, 0, 0, 2, 1, 1, 1, 2,
       0, 1, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0,
       0, 0, 2, 0, 0, 0, 0, 0, 2, 1, 1, 0, 2, 1, 1, 1, 0, 1, 2, 2, 0, 0,
       1, 0, 0, 0, 0, 1, 0, 2, 0, 1, 0, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 0,
       2, 1, 1, 2, 0, 2, 1, 0, 0, 0, 2, 1, 1, 2, 1, 2, 0, 2, 0, 2, 2, 1,
       2, 2, 0, 0, 1, 1, 2, 0, 2, 0, 0, 1, 1, 2, 2, 0, 2, 2, 1, 1, 2, 0,
       0, 2, 0, 0, 0, 0, 2, 0, 2, 2, 1, 2, 2, 1, 1, 0, 1, 1, 2, 1, 0, 1,
       0, 1, 0, 1, 2, 1, 1, 1, 1, 0, 0, 2, 0, 2, 1, 0, 1, 2, 0, 0, 0, 2,
       2, 2, 2, 1, 0, 0, 2, 1, 1, 2, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 2, 2,
       0, 0, 0, 2, 0, 2, 2, 0, 2, 2, 0, 0, 2, 1, 2, 0, 1, 0, 0, 0, 1, 0,
       0, 1, 1, 0, 2, 2, 2, 2], dtype=int64)
```

```python
# Model saving as h5 file


from tensorflow.keras.models import load_model

model.save('image_classification.h5')
```

Learning Outcomes of the Study in respect of Data Science

The larger the data the better the model could predict. Multi class prediction are somewhat relatively harder to train in comparison to a Binary class prediction. Data Augmentation is necessary where we have small datasets of images.



```
import numpy as np
y_pred = np.argmax(y_pred, axis=1)
y_pred
```

```
array([1, 1, 2, 0, 0, 2, 0, 1, 2, 1, 1, 2, 0, 2, 2, 1, 0, 1, 0, 0, 0, 1,
       1, 2, 0, 1, 1, 2, 1, 2, 1, 1, 2, 0, 0, 1, 1, 0, 0, 2, 1, 1, 1, 2,
       0, 1, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0,
       0, 0, 2, 0, 0, 0, 0, 0, 2, 1, 1, 0, 2, 1, 1, 1, 0, 1, 2, 2, 0, 0,
       1, 0, 0, 0, 0, 1, 0, 2, 0, 1, 0, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 0,
       2, 1, 1, 2, 0, 2, 1, 0, 0, 0, 2, 1, 1, 2, 1, 2, 0, 2, 0, 2, 2, 1,
       2, 2, 0, 0, 1, 1, 2, 0, 2, 0, 0, 1, 1, 2, 2, 0, 2, 2, 1, 1, 2, 0,
       0, 2, 0, 0, 0, 0, 2, 0, 2, 2, 1, 2, 2, 1, 1, 0, 1, 1, 2, 1, 0, 1,
       0, 1, 0, 1, 2, 1, 1, 1, 1, 0, 0, 2, 0, 2, 1, 0, 1, 2, 0, 0, 0, 2,
       2, 2, 2, 1, 0, 0, 2, 1, 1, 2, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 2, 2,
       0, 0, 0, 2, 0, 2, 2, 0, 2, 2, 0, 0, 2, 1, 2, 0, 1, 0, 0, 0, 1, 0,
       0, 1, 1, 0, 2, 2, 2, 2], dtype=int64)
```

Limitations of this work and Scope for Future Work

The larger the dataset, the better the model accuracy but in this project, we have scraped the images only from amazon with three different classes so the scraped images are very less. In future we may scrape the data from different ecommerce websites and collect huge data and get the better results.