Name:Kaushik Kotian Roll No.:29 Div:D20B Batch:B

Artificial Intelligence and Data Science - 2
Experiment 7

Aim: To build a cognitive based application to acquire knowledge through image for a customer service application.

Theory:
Extracting text from images is a task called Optical Character Recognition (OCR). It is the conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example: from a television broadcast).

EasyOCR is implemented using Python and PyTorch. If you have a CUDA-capable GPU, the underlying PyTorch can speed up your text detection and OCR speed.

The OCR software is by no means one, a uniform application that serves one and the same purpose. The OCR applications are used to serve lots of different purposes. We can start with "reading" the printed page from a book or a random image with text (for instance, graffiti or advertisement), but we go on to reading street signs, car licence plates, and even captchas. OCR software takes into consideration the following factors and attributes:

1. Text density: On a printed page, the text is dense. However, given an image of a street with a single street sign, the text is sparse. The OCR software has to recognize both.

2. Text structure: Text on a page is usually structured, mostly in strict rows, while text in the wild may be scattered everywhere, in different rotations, shapes, fonts, and sizes.

3. Font: While computer fonts are quite easy to recognize, handwriting fonts are much more inconsistent and, therefore, harder to read.

4. Artifacts: There are almost none of them on a perfectly scanned page, but what about outdoor pictures? In short, this is a completely different story, and you have to keep that in mind when using OCR.

Use cases for text extraction from image:

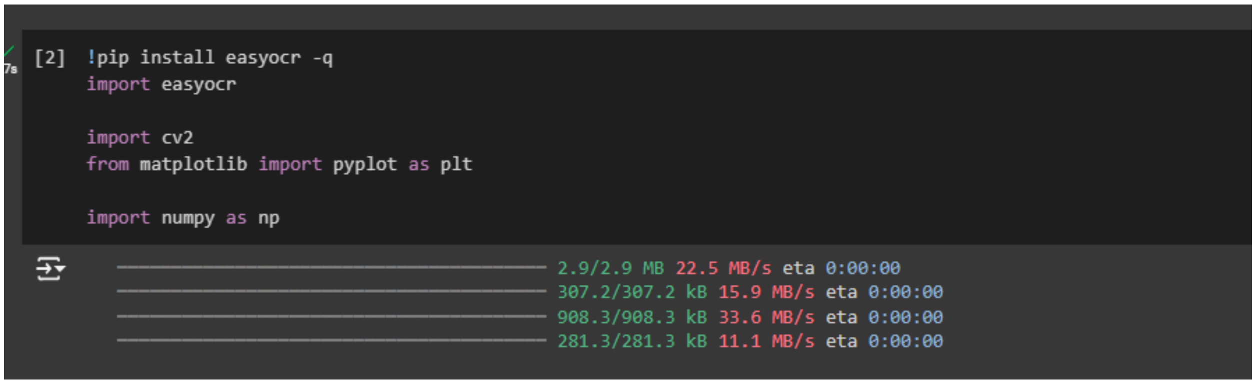
Social media monitoring: Your company can use text extraction from images to follow social media conversations to better understand customers, improve products, or take quick action to avoid a PR crisis. Text extraction from images may offer specific examples of what people on social media are saying about your business.

Client service with text extraction: Text extraction from images allows customer service staff to automate the process of tagging tickets, saving dozens of hours that might be spent on real-world problem-solving.

Business intelligence and text extraction from images: Text extraction from images can also be effective in business intelligence (BI) applications such as market research and competition analysis. You may also get information from a variety of sources, including product reviews and social media, and participate in discussions on topics of interest.

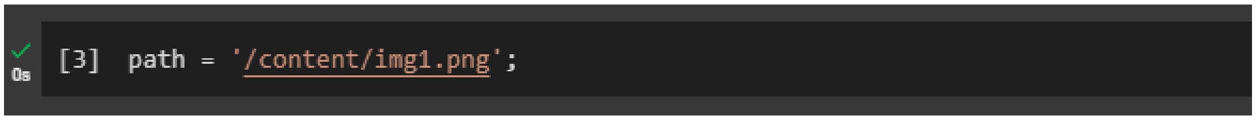
Code and Output:

Extracting text from images and analyzing the sentiment.



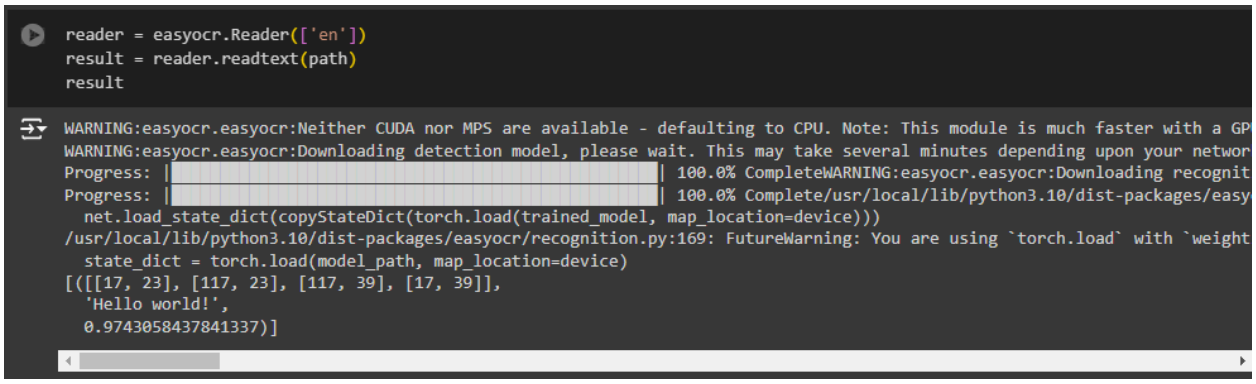
```
[2] !pip install easyocr -q
    import easyocr

    import cv2
    from matplotlib import pyplot as plt

    import numpy as np

    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.9/2.9 MB 22.5 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 307.2/307.2 kB 15.9 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 908.3/908.3 kB 33.6 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 281.3/281.3 kB 11.1 MB/s eta 0:00:00
```

Reading the image and Optical character recognition

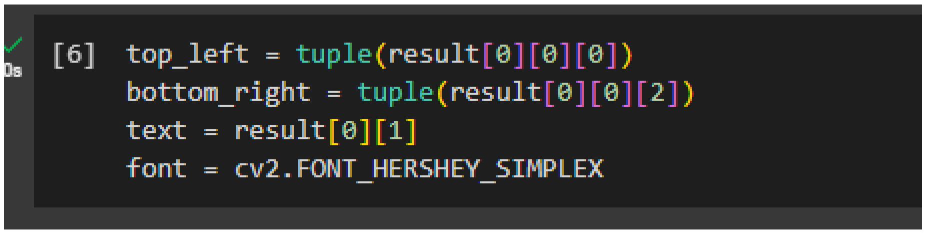


```
[3] path = '/content/img1.png';
```

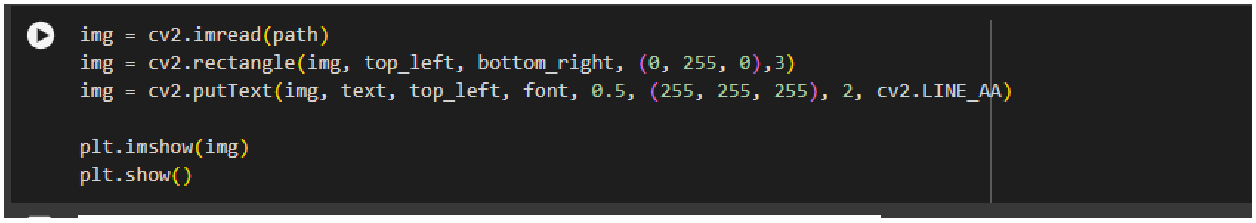
The result gives where the text is in our image and the text which has been recognized and lastly the confidence. Now visualizing where the text is in the image.



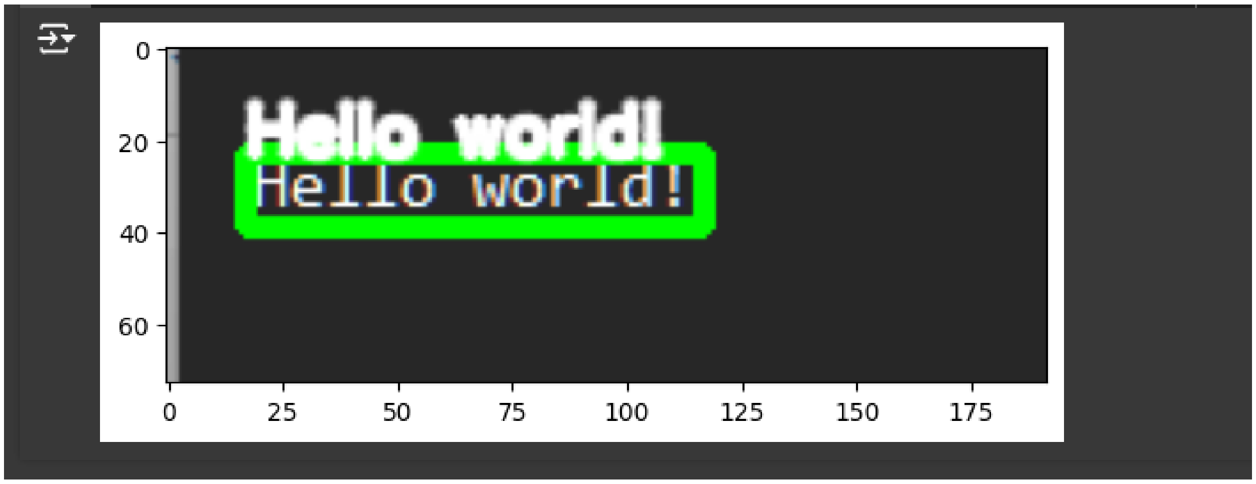
```
reader = easyocr.Reader(['en'])
result = reader.readtext(path)
result

WARNING:easyocr.easyocr:Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GP
WARNING:easyocr.easyocr:Downloading detection model, please wait. This may take several minutes depending upon your networ
Progress: |██████████████████████████████████████████████████| 100.0% CompleteWARNING:easyocr.easyocr:Downloading recognit
Progress: |██████████████████████████████████████████████████| 100.0% Complete/usr/local/lib/python3.10/dist-packages/easy
  net.load_state_dict(copyStateDict(torch.load(trained_model, map_location=device)))
/usr/local/lib/python3.10/dist-packages/easyocr/recognition.py:169: FutureWarning: You are using `torch.load` with `weight
  state_dict = torch.load(model_path, map_location=device)
[([[17, 23], [117, 23], [117, 39], [17, 39]],
  'Hello world!',
  0.9743058437841337)]
```

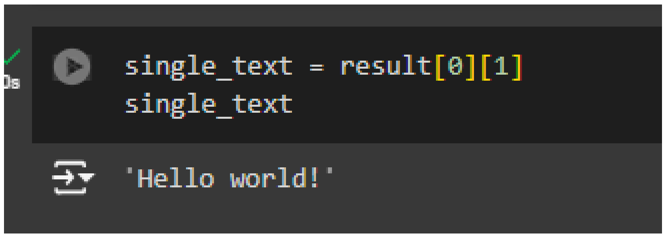


```
[6] top_left = tuple(result[0][0][0])
    bottom_right = tuple(result[0][0][2])
    text = result[0][1]
    font = cv2.FONT_HERSHEY_SIMPLEX
```



```
img = cv2.imread(path)
img = cv2.rectangle(img, top_left, bottom_right, (0, 255, 0),3)
img = cv2.putText(img, text, top_left, font, 0.5, (255, 255, 255), 2, cv2.LINE_AA)

plt.imshow(img)
plt.show()
```

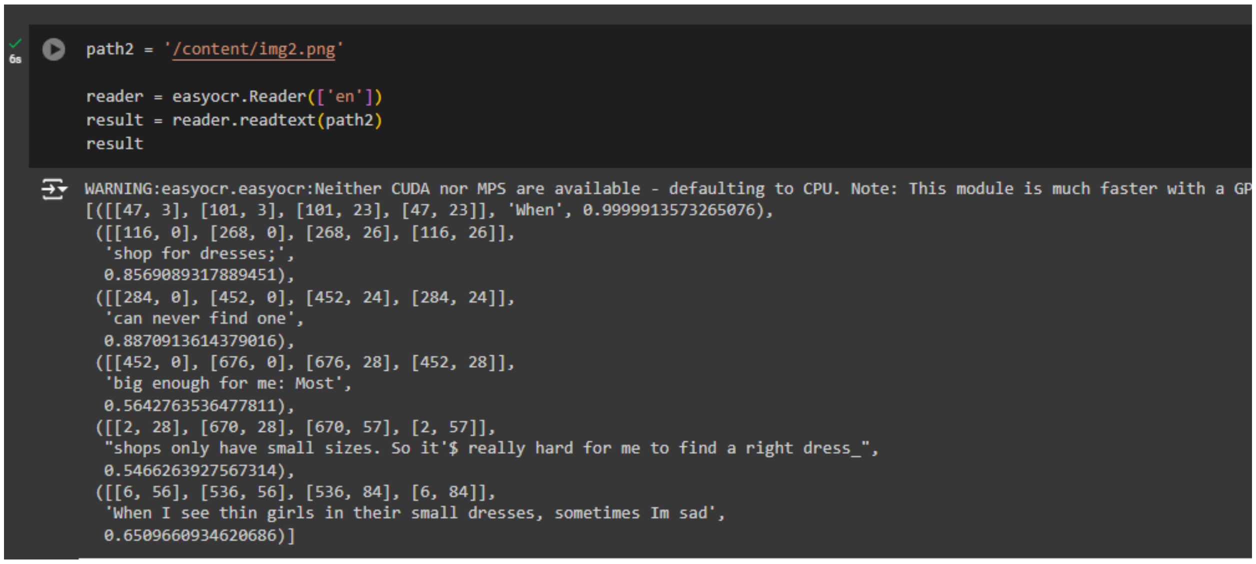




```
single_text = result[0][1]
single_text
```

```
'Hello world!'
```

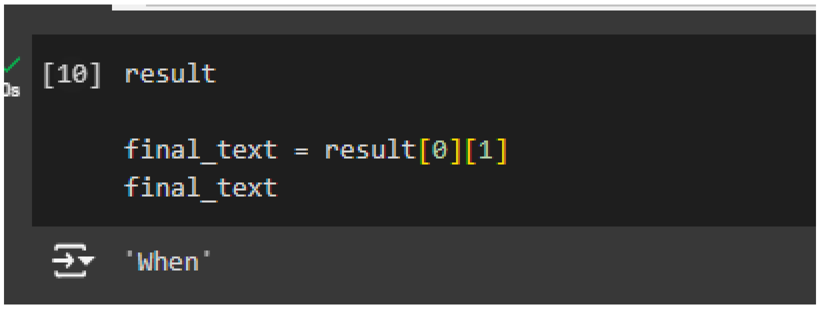
Finally reading an image with multiple lines of text.



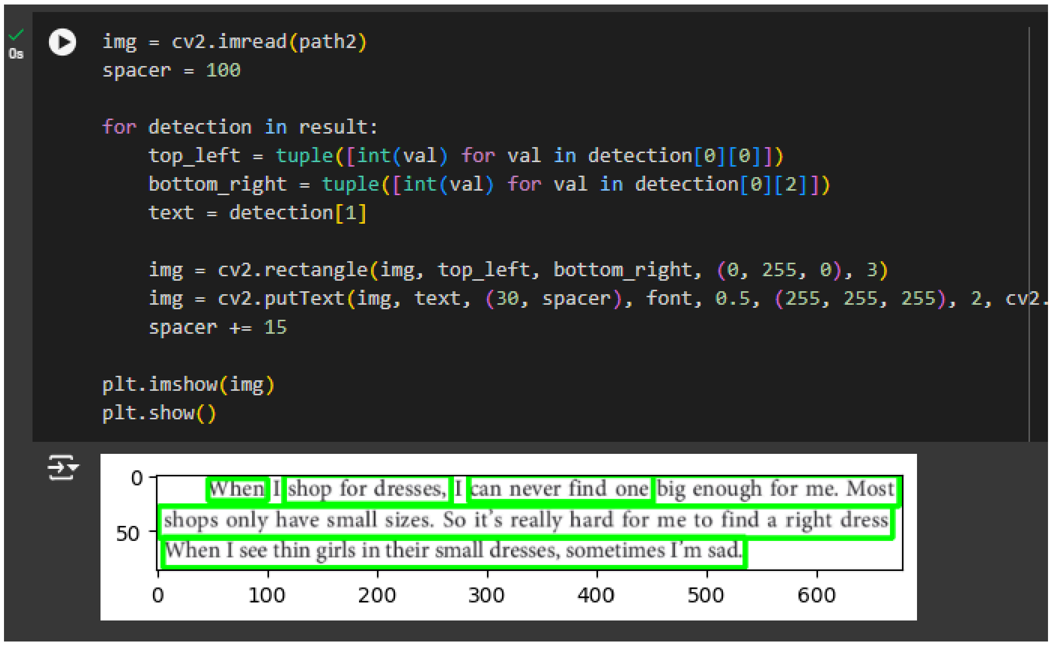
```
path2 = '/content/img2.png'

reader = easyocr.Reader(['en'])
result = reader.readtext(path2)
result
```

```
WARNING:easyocr.easyocr:Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GP
[([[47, 3], [101, 3], [101, 23], [47, 23]], 'When', 0.9999913573265076),
 ([[116, 0], [268, 0], [268, 26], [116, 26]],
  'shop for dresses;',
  0.8569089317889451),
 ([[284, 0], [452, 0], [452, 24], [284, 24]],
  'can never find one',
  0.8870913614379016),
 ([[452, 0], [676, 0], [676, 28], [452, 28]],
  'big enough for me: Most',
  0.5642763536477811),
 ([[2, 28], [670, 28], [670, 57], [2, 57]],
  "shops only have small sizes. So it'$ really hard for me to find a right dress_",
  0.5466263927567314),
 ([[6, 56], [536, 56], [536, 84], [6, 84]],
  'When I see thin girls in their small dresses, sometimes Im sad',
  0.6509660934620686)]
```

Getting the text extracted from the image.



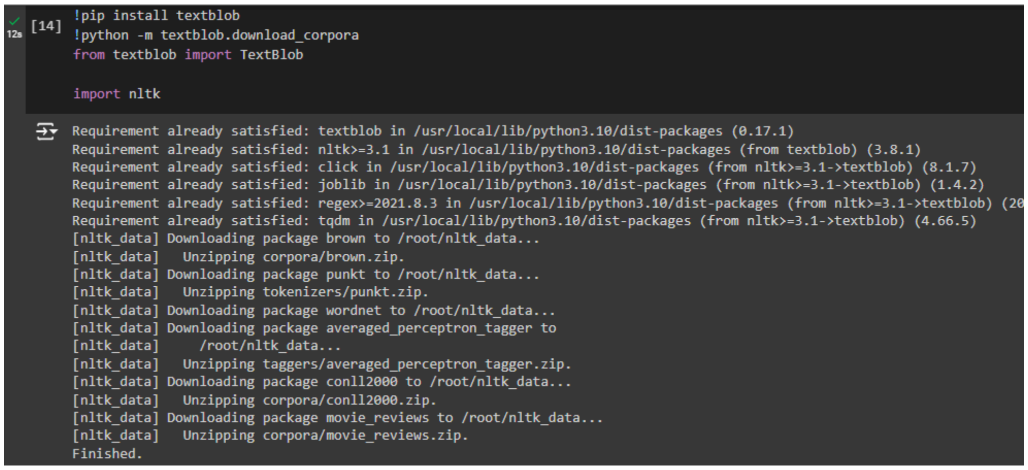
```
[10] result

     final_text = result[0][1]
     final_text

'When'
```



```
img = cv2.imread(path2)
spacer = 100

for detection in result:
    top_left = tuple([int(val) for val in detection[0][0]])
    bottom_right = tuple([int(val) for val in detection[0][2]])
    text = detection[1]

    img = cv2.rectangle(img, top_left, bottom_right, (0, 255, 0), 3)
    img = cv2.putText(img, text, (30, spacer), font, 0.5, (255, 255, 255), 2, cv2.
    spacer += 15

plt.imshow(img)
plt.show()
```

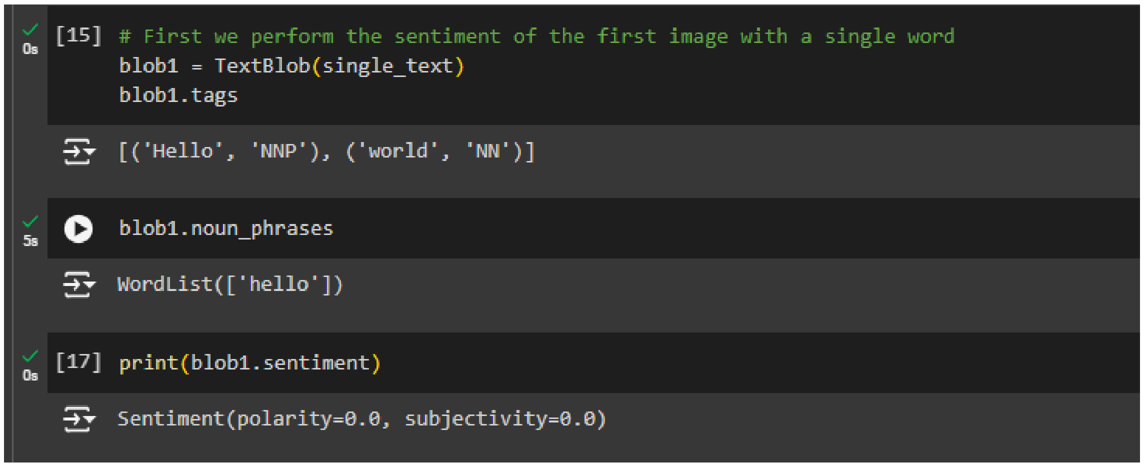
When I shop for dresses, I can never find one big enough for me. Most
shops only have small sizes. So it's really hard for me to find a right dress
When I see thin girls in their small dresses, sometimes I'm sad.

Sentiment Analysis on the extracted text.



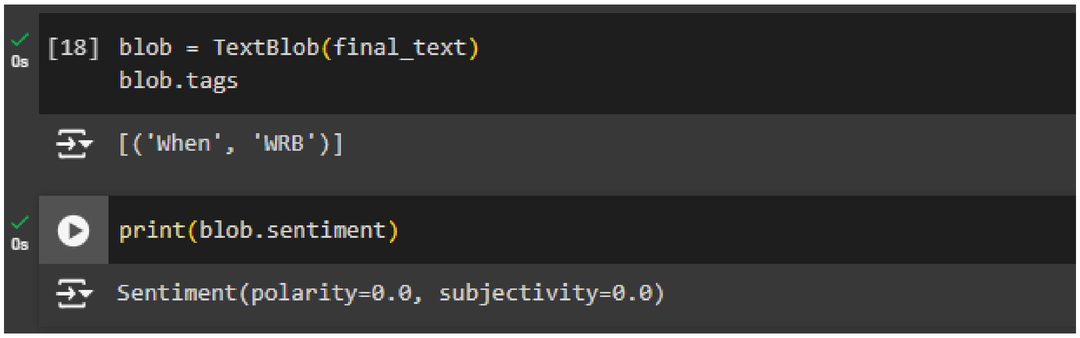
```
[14] !pip install textblob
     !python -m textblob.download_corpora
     from textblob import TextBlob

     import nltk

Requirement already satisfied: textblob in /usr/local/lib/python3.10/dist-packages (0.17.1)
Requirement already satisfied: nltk>=3.1 in /usr/local/lib/python3.10/dist-packages (from textblob) (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob) (20
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob) (4.66.5)
[nltk_data] Downloading package brown to /root/nltk_data...
[nltk_data]   Unzipping corpora/brown.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package conll2000 to /root/nltk_data...
[nltk_data]   Unzipping corpora/conll2000.zip.
[nltk_data] Downloading package movie_reviews to /root/nltk_data...
[nltk_data]   Unzipping corpora/movie_reviews.zip.
Finished.
```

First we perform the sentiment of the first image with a single word.



```
[15] # First we perform the sentiment of the first image with a single word
     blob1 = TextBlob(single_text)
     blob1.tags

[('Hello', 'NNP'), ('world', 'NN')]

blob1.noun_phrases

WordList(['hello'])

[17] print(blob1.sentiment)

Sentiment(polarity=0.0, subjectivity=0.0)
```

Finally we perform the sentiment of the second image with multiple words.



```
[18] blob = TextBlob(final_text)
     blob.tags

[('When', 'WRB')]

print(blob.sentiment)

Sentiment(polarity=0.0, subjectivity=0.0)
```

Conclusion:Thus we studied an overview of how to build a Cognitive based application to acquire knowledge through images for a customer service application.