# Automated Text Extraction and Data Analytics Assistant - by Yeriko Vargas

October 10, 2023

## 0.1 Automated Text Extraction and Data Analytics Assistant - by Yeriko Vargas

### 0.1.1 Summary

The project I developed focuses on automating the process of textual data extraction from images, utilizing Optical Character Recognition (OCR). This technology streamlines data management and documentation tasks, enabling seamless data conversion and analytics.

### 0.1.2 Problem It Addresses

The primary challenge it resolves is providing an automated, accurate solution for converting receipt images into a digital, machine-readable format. This is particularly beneficial for tasks that require merging data from various documents into a single, cohesive report.

### 0.1.3 How It Functions

1. **Preprocessing and Communication:** Initially, the image goes through a preprocessing stage using OpenCV, analogous to conducting comprehensive research to collect relevant data.
2. **Data Extraction:** Tesseract OCR engine is then used for text extraction, similar to extracting valuable insights using data mining techniques.
3. **Data Management:** Extracted text can be easily converted and stored in data frames or other formats, showcasing proficiency in Excel-like data management tasks.

### 0.1.4 Diagram

## 0.2 IMAGE PREPROCESSING and OCR test

[ ]:
```
#                       Diagram

#                 +----------------------+
#                 |  Image Preprocessing|
#                 +----------------------+
#                          |
#                          V
#                 +----------------------+
#                 |   Text Extraction   |
#                 +----------------------+
# Libraries
import cv2
import pytesseract
from PIL import Image

# Function to preprocess image
def preprocess_image(image_path):
    img = cv2.imread(image_path)
    img = cv2.resize(img, None, fx=1.2, fy=1.2, interpolation=cv2.INTER_CUBIC)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    img = cv2.adaptiveThreshold(cv2.GaussianBlur(img, (5, 5), 0), 255,
                                cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.
 ↪THRESH_BINARY, 31, 2)
    return img

# Function to perform OCR
def perform_ocr(img):
    img_new = Image.open(img)
    return pytesseract.image_to_string(img_new, lang='eng')

# Main Function
if __name__ == '__main__':
    pytesseract.pytesseract.tesseract_cmd = 'C:/Program Files/Tesseract-OCR/
 ↪tesseract/tesseract.exe'

    img = preprocess_image('rece.jpg')
    cv2.imwrite('preprocessed_rece.jpg', img)

    text = perform_ocr('preprocessed_rece.jpg')
    print(text)
```

## 1 OCR_Receipt_Data_Extractor

Image Preprocessing: OpenCV is used to preprocess the receipt images, making them more suitable for OCR. Text Extraction: Tesseract OCR engine is used to extract text from the preprocessed images. Data Storage: The extracted text can be stored in a DataFrame or a text file for future use

[ ]:
```
#                           Diagram

#                  +----------------------+
#                  |  Image Preprocessing|
#                  +----------------------+
#                          |
#                          V
#                  +----------------------+
#                  |    Text Extraction   |
#                  +----------------------+
#                          |
#                          V
#                  +----------------------+
#                  |   Data Storage       |
#                  +----------------------+

# Libraries
import cv2
import pytesseract
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import pandas as pd

# Function to preprocess image
def preprocess_image(image_path):
    img = cv2.imread(image_path)
    img = cv2.resize(img, None, fx=1.2, fy=1.2, interpolation=cv2.INTER_CUBIC)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    kernel = np.ones((1, 1), np.uint8)
    img = cv2.dilate(img, kernel, iterations=1)
    img = cv2.erode(img, kernel, iterations=1)
    return img

# Function to show image
def show_image(img):
    plt.figure(figsize=(12,16))
    plt.imshow(img)
    plt.show()

# Function to perform OCR
def perform_ocr(img):
    return pytesseract.image_to_string(img, lang='eng')

# Main code
if __name__ == "__main__":
    # Configuration
    pytesseract.pytesseract.tesseract_cmd = 'C:/Program Files/Tesseract-OCR/
 ↪tesseract/tesseract.exe'

    # Preprocess
    img = preprocess_image('example.jpg')

    # Show preprocessed image
    show_image(img)
    # Perform OCR
    text = perform_ocr(img)

    # Display OCR result
    print(text)
```

[1]:
```
# end
```