Max Liu

Prof. Hernandez

CISB 63

29 October 2023

N.O.L.D. identification project

The NOLD (Name, Organization, Location, Date) project uses multiple NLP methods to find and categorize Names, Locations, Dates, and Organizations in a sample text and highlight them. There are two sample texts included in the code, but if you want to test the code out, you can add your own sample texts. My program uses a flexible dataframe to fill in missing information about the sample text if it is not identified.

GITHUB LINK: https://github.com/NMOD-Max/NOLD

I used NLP methods such as:

1. NER (Named Entity Recognition): I used spaCy to perform Named Entity Recognition and identify entities like names (PERSON), locations (GPE), dates (DATE), and organizations (ORG).

```
# Process named entities
for token in doc:
    if token.ent_type_ == 'PERSON':
        # Format names
        formatted_tokens.append(name_format.format(token.text))
        data_dict['Name'].append(token.text)
    elif token.ent_type_ == 'GPE':
        # Highlight locations
        formatted_tokens.append(location_format.format(token.text))
```

```
        data_dict['Location'].append(token.text)
    elif token.ent_type_ == 'DATE':
        # Underline dates
        formatted_tokens.append(date_format.format(token.text))
        data_dict['Date'].append(token.text)
    elif token.ent_type_ == 'ORG':
        # Italicize organizations
        formatted_tokens.append(org_format.format(token.text))
        data_dict['Organization'].append(token.text)
    else:
        formatted_tokens.append(token.text)
```

2. WordCloud: I created a word cloud to visualize the frequency of words (including names and locations) in the text.

```
# Create a word cloud for names, locations, and organizations
all_text = ' '.join([name + " " + location for name, location, date,
org in zip(df['Name'], df['Location'], df['Date'],
df['Organization'])])
wordcloud = WordCloud(width=800, height=400,
background_color="white").generate(all_text)

# Display the word cloud using Matplotlib
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.title("Word Cloud for Names and locations")
plt.show()
```

3. Translation: I used the TextBlob library to translate the text to another language.

```
# Create a TextBlob object with the text
blob = TextBlob(text)

# Translate the text to Spanish
```

```
translated_blob = blob.translate('en', 'ja')

# Print the translated text
print(translated_blob)
```

4. Tokenization: Tokenization is the process of splitting text into individual words or tokens. In the code, tokenization is implicitly performed when iterating through the doc object (generated by spaCy) to process each word separately.

```
words = word_tokenize(text)
```

5. POS: Categorizing words in a text (corpus) in correspondence with a particular part of speech, depending on the definition of the word and its context.

```
nltk.pos_tag(words)
```