NAME : SAI CHARAN . P

ROLL:NO : 2403A52343

LAB : 5

Building Robust spaCy Pipelines for Noisy Social Media Data

Dataset: Twitter US Airline Sentiment (Kaggle)

Step 1: Install required libraries and load the spaCy English model.

```
!pip install pandas spacy matplotlib seaborn emoji
!python -m spacy download en_core_web_sm

import re
import pandas as pd
import spacy
from spacy.language import Language
from spacy.tokens import Doc
from collections import Counter
import matplotlib.pyplot as plt
import seaborn as sns
import emoji

nlp = spacy.load("en_core_web_sm")
```

```
Collecting emoji
  Downloading emoji-2.15.0-py3-none-any.whl.metadata (5.7 kB)
Requirement already satisfied: numpy>=1.26.0 in /usr/local/lib/python3.12/di
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/pyth
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dis
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/d
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/p
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/pytho
Requirement already satisfied: thinc<8.4.0,>=8.3.4 in /usr/local/lib/python3
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/pyt
Requirement already satisfied: weasel<0.5.0,>=0.4.2 in /usr/local/lib/python
Requirement already satisfied: typer-slim<1.0.0,>=0.3.0 in /usr/local/lib/py
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/pyt
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist
```

```
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/pyth
Requirement already satisfied: pydantic-core==2.41.4 in /usr/local/lib/pytho
Requirement already satisfied: typing-extensions>=4.14.1 in /usr/local/lib/p
Requirement already satisfied: typing-inspection>=0.4.2 in /usr/local/lib/py
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/py
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dis
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3
Requirement already satisfied: blis<1.4.0,>=1.3.0 in /usr/local/lib/python3
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/py
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.12/dis
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/py
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packa
Downloading emoji-2.15.0-py3-none-any.whl (608 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 608.4/608.4 kB 13.8 MB/s eta 0:0
Installing collected packages: emoji
Successfully installed emoji-2.15.0
Collecting en-core-web-sm==3.8.0
  Using cached https://github.com/explosion/spacy-models/releases/download/e
✓ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
⚠ Restart to reload dependencies
If you are in a Jupyter or Colab notebook, you may need to restart Python in
order to load all the package's dependencies. You can do this by selecting t
'Restart kernel' or 'Restart runtime' option.
```

Step 2: Load the Twitter US Airline Sentiment dataset.

```
df = pd.read_csv("Tweets.csv")
df.head()
```

	tweet_id	airline_sentiment	airline_sentiment_confidence	negat
0	570306133677760513	neutral	1.0000	
1	570301130888122368	positive	0.3486	
2	570301083672813571	neutral	0.6837	
3	570301031407624196	negative	1.0000	
4	570300817074462722	negative	1.0000	

Next steps: Generate code with df | New interactive sheet

Step 3: Select tweet text and sentiment columns and remove missing values.

```
df = df[["text", "airline_sentiment"]]
df.dropna(inplace=True)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14640 entries, 0 to 14639
Data columns (total 2 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   text               14640 non-null  object
 1   airline_sentiment  14640 non-null  object
dtypes: object(2)
memory usage: 228.9+ KB
```

Step 4: Clean tweets by removing URLs, mentions, emojis, special characters, and converting text to lowercase.

```
def clean_tweet(text):
    text = text.lower()
    text = re.sub(r"http\S+|www\S+", "", text)        # URLs
    text = re.sub(r"@\w+", "", text)                  # Mentions
    text = re.sub(r"#\w+", "", text)                  # Hashtags (text only)
    text = emoji.replace_emoji(text, replace="")     # Emojis
```

```
    text = re.sub(r"[^a-z\s]", "", text)              # Special characters
    text = re.sub(r"\s+", " ", text).strip()
    return text
```

Step 5: Create a cleaned tweet corpus after preprocessing.

```
df['cleaned_text'] = df['text'].apply(clean_tweet)
display(df.head())
```

	text	airline_sentiment	cleaned_text
0	@VirginAmerica What @dhepburn said.	neutral	what said
1	@VirginAmerica plus you've added commercials t...	positive	plus youve added commercials to the experience...
2	@VirginAmerica I didn't today... Must mean I n...	neutral	i didnt today must mean i need to take another...

Step 6: Initialize the spaCy NLP pipeline.

```
nlp = spacy.load("en_core_web_sm")
```

Step 7: Create and add a custom spaCy pipeline component to detect hashtags.

```
# Define custom extension
Doc.set_extension("hashtags", default=[], force=True)

@Language.component("hashtag_detector")
def hashtag_detector(doc):
    hashtags = [token.text for token in doc if token.text.startswith("#")]
    doc._.hashtags = hashtags
    return doc

# Add component to pipeline, only if it doesn't already exist
if "hashtag_detector" not in nlp.pipe_names:
    nlp.add_pipe("hashtag_detector", last=True)

nlp.pipe_names
```

```
['tok2vec',
 'tagger',
 'parser',
 'attribute_ruler',
 'lemmatizer',
 'ner',
 'hashtag_detector']
```

Step 8: Process the cleaned tweets using the customized spaCy pipeline.

```
df['cleaned_text'] = df['text'].apply(clean_tweet)
docs = list(nlp.pipe(df["cleaned_text"]))
```

Step 9: Extract lemmas and part-of-speech tags from processed tweets.

```
lemmatized_pos = []

for doc in docs:
    tokens = [(token.lemma_, token.pos_)
              for token in doc
              if not token.is_stop and not token.is_punct]
    lemmatized_pos.append(tokens)

lemmatized_pos[:2]
```

```
[[('say', 'VERB')],
 [('plus', 'CCONJ'),
  ('ve', 'AUX'),
  ('add', 'VERB'),
  ('commercial', 'NOUN'),
  ('experience', 'NOUN'),
  ('tacky', 'ADV')]]
```

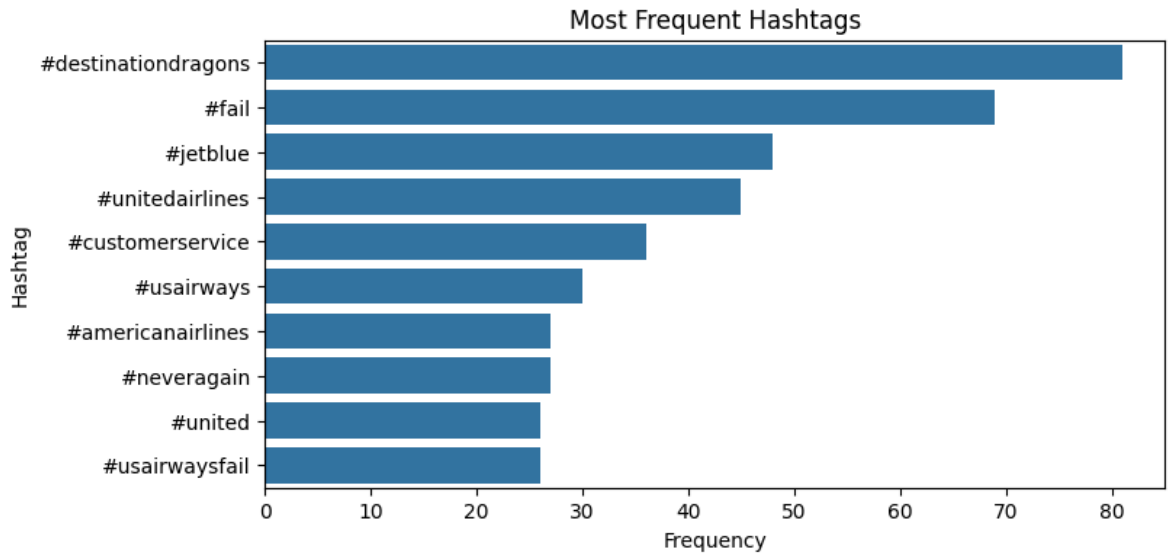
Step 10: Extract hashtags from original tweets and compute their frequencies.

```
hashtag_docs = list(nlp.pipe(df["text"]))

all_hashtags = []
for doc in hashtag_docs:
    all_hashtags.extend(doc._.hashtags)

hashtag_freq = Counter(all_hashtags)
hashtag_freq.most_common(10)
```

```
[('#destinationdragons', 81),
 ('#fail', 69),
 ('#jetblue', 48),
 ('#unitedairlines', 45),
 ('#customerservice', 36),
 ('#usairways', 30),
 ('#americanairlines', 27),
 ('#neveragain', 27),
 ('#united', 26),
 ('#usairwaysfail', 26)]
```

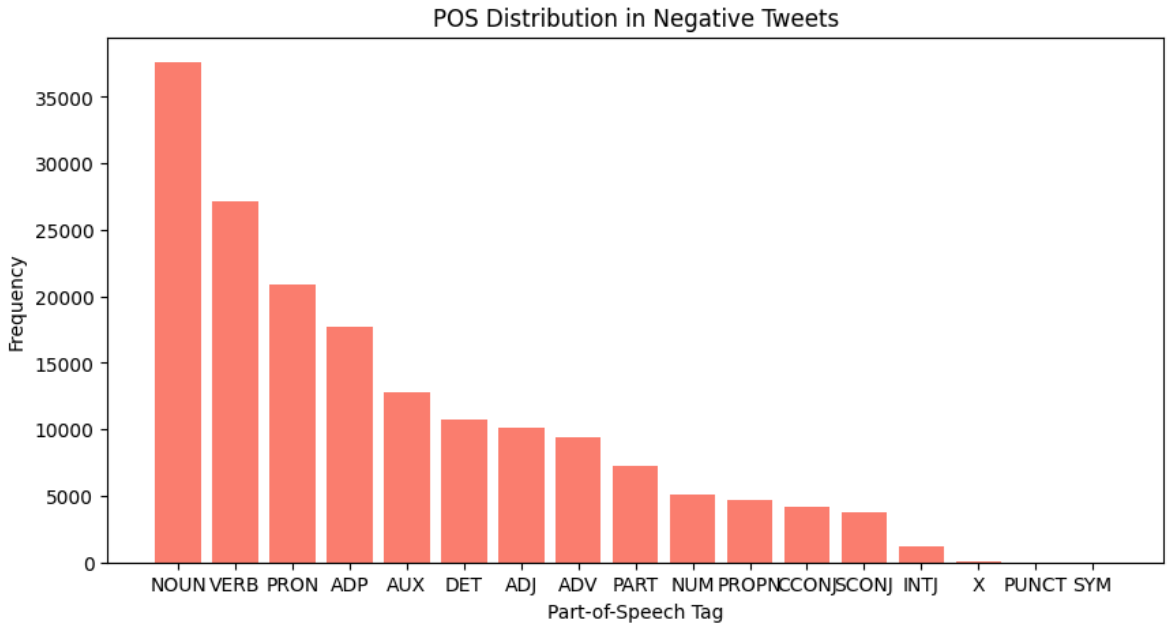
Step 11: Visualize the most frequent hashtags.

```
top_hashtags = hashtag_freq.most_common(10)
hashtags, counts = zip(*top_hashtags)

plt.figure(figsize=(8, 4))
sns.barplot(x=list(counts), y=list(hashtags))
plt.title("Most Frequent Hashtags")
plt.xlabel("Frequency")
plt.ylabel("Hashtag")
plt.show()
```



Step 12: Filter negative tweets and visualize their POS tag distribution.

```
# Filter negative tweets
df_neg = df[df['airline_sentiment'] == 'negative'].copy()
docs_neg = [docs[i] for i in df_neg.index]

# Count POS tags
pos_counter = Counter()
for doc in docs_neg:
    pos_counter.update([token.pos_ for token in doc])

# Plot POS distribution
pos_tags, counts = zip(*pos_counter.most_common())
plt.figure(figsize=(10,5))
plt.bar(pos_tags, counts, color='salmon')
plt.title("POS Distribution in Negative Tweets")
plt.xlabel("Part-of-Speech Tag")
plt.ylabel("Frequency")
plt.show()
```



Summary:

Data Analysis Key Findings

- The `Doc.set_extension` function for the 'hashtags' extension was modified by adding `force=True`.

Insights or Next Steps

- This modification enables the re-registration of the 'hashtags' extension on the `Doc` object, which is useful during iterative development or when resetting extensions.