```
!git clone https://github.com/OTN-Rajapaksha/Simulating-an-NLP-Pipeline-for-Complaint-Classification-02.git

Cloning into 'Simulating-an-NLP-Pipeline-for-Complaint-Classification-02'...
remote: Enumerating objects: 3, done.ote: Counting objects: 100% (3/3), done.ote: Compressing objects: 100% (2/2), done.ote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
```

Part A – Text Preprocessing Let's process the review: "I ordered a dress and it arrived late. The quality is terrible and I can’t return it!"

```
import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer

# Download necessary resources
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

review = "I ordered a dress and it arrived late. The quality is terrible and I can’t return it!"

# 1. Convert to lowercase
review_lower = review.lower()
print("Lowercase:", review_lower)

# 2. Tokenize sentences and words
sentences = sent_tokenize(review_lower)
print("Sentences:", sentences)

words = word_tokenize(review_lower)
print("Words:", words)

# 3. Remove stopwords
stop_words = set(stopwords.words('english'))
words_nostop = [word for word in words if word.isalpha() and word not in stop_words]
print("Without stopwords:", words_nostop)

# 4. Lemmatization
lemmatizer = WordNetLemmatizer()
lemmatized = [lemmatizer.lemmatize(word) for word in words_nostop]
print("Lemmatized:", lemmatized)

[nltk_data] Downloading package punkt to
[nltk_data]     /Users/oshanrajapaksha/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/oshanrajapaksha/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     /Users/oshanrajapaksha/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!

Lowercase: i ordered a dress and it arrived late. the quality is
terrible and i can’t return it!
Sentences: ['i ordered a dress and it arrived late.', 'the quality is
terrible and i can’t return it!']
Words: ['i', 'ordered', 'a', 'dress', 'and', 'it', 'arrived', 'late',
'.', 'the', 'quality', 'is', 'terrible', 'and', 'i', 'can', '’', 't',
'return', 'it', '!']
Without stopwords: ['ordered', 'dress', 'arrived', 'late', 'quality',
'terrible', 'return']
Lemmatized: ['ordered', 'dress', 'arrived', 'late', 'quality',
'terrible', 'return']
```

Part B – Feature Engineering

```
from collections import Counter

bow = Counter(lemmatized)
print("Bag-of-Words:", bow)

Bag-of-Words: Counter({'ordered': 1, 'dress': 1, 'arrived': 1, 'late':
1, 'quality': 1, 'terrible': 1, 'return': 1})
```

Part C – Mock Classification

Sentiment: Negative Justification:

Words such as "late", "terrible", and "return" indicate dissatisfaction. The reviewer complains about late delivery, poor quality, and inability to return the item, all of which are negative experiences.

```
Part D – Reflection on Evaluation

Evaluation Metrics:

Accuracy: Proportion of correct predictions out of total predictions.
Precision: Proportion of true positives out of all predicted
positives.
Recall: Proportion of true positives out of all actual positives.
F1-Score: Harmonic mean of precision and recall.

Most Important Metric:

For customer complaint classification, Recall is often most important.
```

```
Missing a genuine complaint (false negative) can lead to unresolved
customer issues, so it is crucial to identify all actual complaints,
even if it means occasionally flagging non-complaints by mistake.
```

Part E – Additional Review

Let's process:

"Loved the product! It arrived earlier than expected and fits perfectly."

```
review2 = "Loved the product! It arrived earlier than expected and
fits perfectly."

# Preprocessing
review2_lower = review2.lower()
sentences2 = sent_tokenize(review2_lower)
words2 = word_tokenize(review2_lower)
words2_nostop = [word for word in words2 if word.isalpha() and word
not in stop_words]
lemmatized2 = [lemmatizer.lemmatize(word) for word in words2_nostop]

# Bag-of-Words
bow2 = Counter(lemmatized2)

print("Lemmatized:", lemmatized2)
print("Bag-of-Words:", bow2)
```

```
Lemmatized: ['loved', 'product', 'arrived', 'earlier', 'expected',
'fit', 'perfectly']
Bag-of-Words: Counter({'loved': 1, 'product': 1, 'arrived': 1,
'earlier': 1, 'expected': 1, 'fit': 1, 'perfectly': 1})
```