Text Analysis

Estimated Time: 25 minutes

Objectives

After completing this reading, you will be able to:

- Identify text analysis techniques and applications
- Explore the bag of words (BoW) model
- Explore Python libraries that specialize in text analysis, such as NLKT and Genism

An Overview of Text Analysis, Techniques, and Applications

The internet hosts a wealth of unstructured data, encompassing tweets and text created by individuals.

This unstructured data lacks the organization of structured data found in databases, creating difficulties in analysis and insight extraction.

For instance, while structured data neatly fits into tables and columns, unstructured data may include text from social media posts, articles, or comments, which requires different processing techniques.

Consequently, there is an urgent demand for developing techniques and tools to extract valuable information, discern patterns, and reveal hidden insights within this expansive expanse of unstructured data. These initiatives are vital for businesses, researchers, and organizations to utilize the vast knowledge and opportunities available in the digital domain.

Text analysis is the process of examining and extracting meaningful information from unstructured text data. Text analysis addresses the challenges posed by unstructured data by employing methods such as natural language processing (NLP), machine learning, and statistical analysis to process and make sense of textual information.

Techniques of Text Analysis:

Let's explore the text analysis stages through a use case of a fictional company, **TechTrend**, which manufactures and sells smartphones.

> **TechTrend wants to analyze customer reviews of their latest flagship smartphone model, TechTrend X1, to gain insights into consumer sentiment, identify areas for product improvement, and inform marketing strategies.**

Now let's apply the **Text Analysis** techniques given below to solve this use case:

1. Tokenization: Tokenization involves breaking down text into smaller units, typically words or phrases known as tokens, to facilitate subsequent analysis. This helps standardize the text data and facilitates subsequent processing steps.

TechTrend gathers customer reviews from various online platforms, such as e-commerce websites and social media. They tokenize each review, breaking them into individual words or phrases for analysis.

The three methods of Text tokenization are:

- **Word Tokenization**: Splits text into individual words. For example, a customer review might be tokenized into words like **love, battery, camera, performance**, and so on.
- **Phrase Tokenization**: Splits text into meaningful phrases or chunks. For example, the following sentence: **The TechTrend X1 boasts a sleek design and intuitive user interface, but its battery life could be improved**. can be tokenized into the following tokens: **The TechTrend**, **sleek design**,**intuitive user**, and so on.
- **Sentence Tokenization**: Splits text into sentences. For example the following paragraph:

  The TechTrend X1 boasts a sleek design and intuitive user interface. Its exceptional camera quality captures stunning photos even in low light conditions. However, some users have reported issues with battery life, suggesting room for improvement. can be split into the following sentences:

  - The TechTrend X1 boasts a sleek design and intuitive user interface.
  - Its exceptional camera quality captures stunning photos even in low light conditions.
  - However, some users have reported issues with battery life, suggesting room for improvement.

2. Stopword Removal: Stopwords are common words like the, and, is, and so on, that occur frequently in text but typically do not convey significant meaning. Removing stopwords can help reduce noise and improve the quality of analysis results.

For instance, removing stopwords leaves us with words like love, battery, and camera highlighting the key aspects of the review.

3. Stemming and Lemmatization: Stemming and lemmatization are techniques used to normalize words by reducing them to their base or root forms. This helps in collapsing variations of words to a common form, thereby reducing vocabulary size and improving analysis accuracy.

In the TechTrend example, we can demonstrate stemming and lemmatization using the phrase **capturing stunning photos**.
Stemming simplifies words to their root or base form. In this case, the stem of **capturing** is **capture** and the stem of **photos** is **photo**.

Lemmatization reduces words to their base or dictionary form, considering grammatical rules. In this example, **capturing** would be lemmatized to **capture** and **photos** would remain unchanged.

Here is how the phrase would appear after stemming and lemmatization:

Stemmed: **capture stunning photo**
Lemmatized: **capture stunning photos**

4. Word Frequency Analysis: Word frequency analysis comprises the process of counting the occurrences of individual words within a document or corpus.

It aids in recognizing the most prevalent words and terms, and offering insights into the primary topics or themes found within the text data.

Concerning the Techtrend use case, we can analyze the frequency of words in the reviews to identify recurring topics, such as **camera, battery, performance, design, and so on**. This helps understand which aspects of the phone customers discuss most frequently.

5. Sentiment Analysis: Sentiment analysis is used to determine the sentiment expressed in text, such as positive, negative, or neutral. This technique is widely used for tasks like brand monitoring, customer feedback analysis, and social media sentiment analysis.

Using sentiment analysis, TechTrend determines the overall sentiment expressed in the reviews - whether they are positive, negative, or neutral. They look for keywords and phrases indicative of customer satisfaction or dissatisfaction with different features of the phone.

6. Named Entity Recognition (NER): NER is the process of identifying and categorizing named entities such as people, organizations, locations, dates, etc., mentioned in text. This technique is valuable for tasks like information retrieval, data extraction, and entity linking.

TechTrend employs NER to identify and categorize named entities mentioned in the reviews, such as the brand name **TechTrend,** specific phone models, or mentions of competitors like **Apple** or **Samsung.**

7. Topic Modeling: Topic modeling is a statistical technique used to identify latent topics or themes within a collection of documents. It helps in organizing and summarizing large volumes of text data and can be used for tasks like content recommendation, trend analysis, and document clustering.

Applying topic modeling techniques, TechTrend uncovers latent topics within the reviews, such as **camera quality, battery life, software performance, and so on**. This helps in organizing the reviews and understanding what customers are discussing most frequently.

8. Text Classification:

Text classification involves categorizing text documents into predefined categories or labels based on their content. This technique is useful for tasks like spam detection, sentiment analysis, document tagging, and content moderation.

They classify reviews into predefined categories based on their content, such as **camera performance, battery life, design, and so on.** This enables them to quickly identify areas where the TechTrend X1 excels or needs improvement.

9. Document Summarization:

Document summarization aims to automatically generate concise summaries of lengthy text documents. It helps in extracting the main ideas and key information from documents, enabling users to quickly grasp their contents.

Finally, TechTrend generates concise summaries of the reviews, highlighting the most frequently mentioned features, common complaints, and overall sentiment. These summaries provide actionable insights for product development, marketing, and customer service teams.

Applications of Text Analysis

Here are a few examples showcasing the diverse applications of Text Analysis:

- **Understanding Customer Feedback**: Analyzing customer reviews, survey responses, and social media comments to grasp sentiment, spot trends, and refine products or services.
- **Market Insight**: Extracting insights from market reports, industry publications, and consumer forums to track emerging trends, competitor strategies, and consumer preferences.
- **Social Media Monitoring**: Observing mentions, sentiment, and engagement on social media platforms to manage brand image, identify influencers, and gauge public opinion.
- **Enhancing Customer Support**: Automating responses to customer inquiries, categorizing support tickets, and pinpointing common issues to boost response efficiency and satisfaction.
- **Tailoring Content and Recommendations**: Analyzing user behavior and preferences to deliver personalized content suggestions, product recommendations, and targeted advertising.
- **Financial Analysis**: Reviewing news articles, earnings reports, and market trends to inform investment decisions, forecast stock movements, and evaluate economic indicators.
- **Healthcare Analytics**: Scrutinizing electronic health records, clinical notes, and medical literature to uncover patterns, enhance patient outcomes, and guide clinical decisions.
- **Legal Document Review**: Scanning contracts, court rulings, and legal documents to identify key terms, obligations, and risks in legal proceedings and contract negotiations.
- **Academic Research**: Examining research papers, scholarly articles, and citation networks to identify emerging subjects, evaluate research impact, and support literature reviews.

Exploring Bag of Words: A Comprehensive Review

The bag of words (BoW) model is a fundamental concept in natural language processing (NLP) that represents text as a collection of words without considering grammar or word order. In BoW, a document is represented as a **bag** (unordered collection) of words, where each word's presence or frequency is used as a feature for further analysis.

Machine learning algorithms primarily work with numerical data rather than text data.

BoW provides a simple yet effective way to represent text data numerically, making it suitable for machine learning algorithms by transforming the variable-length text into fixed-length inputs.

BoW allows us to extract relevant features from text data, such as the presence or absence of certain words or their frequencies. We can then use these features to train machine learning models to perform tasks like document classification, topic modelling, and information retrieval.

Let's now apply the below outlined steps of Bag of Words technique to the following two reviews obtained from TechTrend data:

> **Review 1: The TechTrend X1 camera captures stunning photos, but the battery life could be better. I'm very impressed with the camera quality.**
> **Review 2: I'm disappointed with the TechTrend X1 battery life, although the camera quality is exceptional. However, the camera features are lacking.**

- **Tokenization**: The text is divided into individual words or tokens. This process involves removing punctuation, splitting text into words, and converting them to lowercase for consistency.

  Tokenized Words for Review 1: [“the”, “techtrend”, “x1”, “camera”, “captures”, “stunning”, “photos”, “but”, “the”, “battery”, “life”, “could”, “be”, “better”, “i’m”, “very”, “impressed”, “with”, “the”, “camera”, “quality”]

  Tokenized Words for Review 2: [“i’m”, “disappointed”, “with”, “the”, “techtrend”, “x1”, “battery”, “life”, “although”, “the”, “camera”, “quality”, “is”, “exceptional”, “however”, “the”, “camera”, “features”, “are”, “lacking”]

  Vocabulary Creation: A unique vocabulary of words present in the entire corpus is established. This vocabulary comprises all distinct words encountered across all documents in the corpus, excluding common stop words that do not carry significant meaning.

  Unique Vocabulary with respect to the two reviews of TechTrend will be: [“the”, “techtrend”, “x1”, “camera”, “captures”, “stunning”, “photos”, “but”, “battery”, “life”, “could”, “be”, “better”, “i’m”, “very”, “impressed”, “with”, “quality”, “is”, “exceptional”, “however”, “features”, “are”, “lacking”]

  Vectorization: Each document is represented as a numerical vector based on the vocabulary created in the previous step.

  The two main types of vectorization are:
  Binary Bag of Words (BoW) Vectorization: In Binary BoW, the vectorization process involves representing each document as a binary vector, where each element indicates the presence (1) or absence (0) of a word from the vocabulary in the document.

  The Binary BOW for both the TechTrend reviews are:

  Review 1 Binary BoW Vector:

  “the”: 1
  “techtrend”: 1
  “x1”: 1
  “camera”: 1
  “captures”: 1
  “stunning”: 1
  “photos”: 1
  “but”: 1
  “battery”: 1
  “life”: 1
  “could”: 1
  “be”: 1
  “better”: 1
  “i’m”: 1
  “very”: 1
  “impressed”: 1
  “with”: 1
  “quality”: 1
  “is”: 0
  “exceptional”: 0
  “however”: 0
  “features”: 0
  “are”: 0
  “lacking”: 0

  Review 2 Binary BoW Vector:

  “the”: 1
  “techtrend”: 1
  “x1”: 1
  “camera”: 1
  “captures”: 0
  “stunning”: 0
  “photos”: 0
  “but”: 1
  “battery”: 1
  “life”: 1
  “could”: 0
  “be”: 0
  “better”: 0
  “i’m”: 1
  “very”: 0
  “impressed”: 1
  “with”: 1
  “quality”: 1
  “is”: 1
  “exceptional”: 1
  “however”: 1
  “features”: 1
  “are”: 1
  “lacking”: 1

Vector representation of these two reviews will be as follows:

Review 1 Binary BoW Vector: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
Review 2 Binary BoW Vector: [1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Frequency Bag of Words (BoW) Vectorization:
In Frequency BoW, the vectorization process involves representing each document as a numerical vector, where each element represents the frequency of a word from the vocabulary in the document.

BoW Vector for Review 1:
TechTrend: 1

X1: 1
camera: 2
captures: 1
stunning: 1
photos: 1
battery: 1
life: 1
could: 1
be: 1
better: 1
I'm: 1
disappointed: 0
with: 1
quality: 1
is: 0
exceptional: 0
very: 1
impressed: 1
however: 0
features: 0
are: 0
lacking: 0

BoW Vector for Review 2:

TechTrend: 1
X1: 1
camera: 1
captures: 0
stunning: 0
photos: 0
battery: 1
life: 1
could: 0
be: 0
better: 0
I'm: 1
disappointed: 1
with: 1
quality: 1
is: 1
exceptional: 1
very: 0
impressed: 0
however: 1
features: 1
are: 1
lacking: 1

Review 1 BoW Vector: [2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0]
Review 2 BoW Vector: [2, 1, 1, 2, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1]

Normalization: Normalization techniques such as TF-IDF can be applied to the BoW vectors to adjust for the varying lengths of documents and the importance of words across the corpus.

TF-IDF, short for Term Frequency-Inverse Document Frequency, is a method used in text mining and information retrieval to assess the significance of a word within a document relative to a collection of documents (corpus).

TF-IDF is computed based on two main factors:

Term Frequency (TF) measures the frequency of a term (word) in a document. It's calculated as the number of times a word appears in a document divided by the total number of words in the document.

Inverse Document Frequency (IDF) measures the importance of a term across multiple documents in the corpus. It's calculated as the logarithm of the total number of documents divided by the number of documents containing the term. Terms that appear in many documents have a lower IDF, while terms that appear in few documents have a higher IDF.

The TF-IDF value for each term in the vector is calculated as TF * IDF. This value represents the importance of the term in the document relative to its frequency across the entire corpus. Terms with higher TF-IDF values are considered more important to the document.

Assuming a simplified scenario:

Total number of documents in the corpus (N): 1000
Number of documents containing the word camera (df_camera): 500
Number of documents containing the word quality (df_quality): 100

Calculating IDF:

For the word **camera**:

IDF_camera = log(N / df_camera) = log(1000 / 500) = log(2) ≈ 0.301
For the word **quality**:

IDF_quality = log(N / df_quality) = log(1000 / 100) = log(10) ≈ 1
Interpretation:

camera has a relatively low IDF value of approximately 0.301. This indicates that the word **camera** appears quite frequently across the documents in the corpus. Despite its high frequency, it may still have importance within individual documents due to its high TF (Term Frequency).

quality has a relatively high IDF value of approximately 1. This suggests that the word **quality** appears less frequently across the documents in the corpus compared to **camera.** As a result, **quality** is considered to be more important or informative when it appears in a document, as it is less common across the entire corpus.

In summary, **camera** has a low IDF value, indicating that it is a common term across the corpus, while **quality** has a high IDF value, indicating that it is a less common term and therefore more informative when it appears in a document.

This metric assigns higher weights to terms that are frequent within a document but infrequent across the corpus, aiding in tasks such as text classification and document ranking.

Exploring Python libraries specialized in text analysis, such as NLTK and Gensim

NLTK: The NLTK library, an abbreviation for Natural Language Toolkit, is a robust open-source Python library utilized for natural language processing (NLP) endeavors.

It furnishes an array of tools and resources for diverse NLP tasks, encompassing tokenization, stemming, lemmatization, part-of-speech tagging, named entity recognition, parsing, and beyond.

NLTK finds extensive application across research, education, and practical implementations in computational linguistics and artificial intelligence, facilitating the development and deployment of algorithms aimed at analyzing and comprehending human language.

Some of the important methods in the context of BOW model are:

word_tokenize(): This method in NLTK is a function used for tokenizing text into individual words or tokens.

Syntax:
`nltk.tokenize.word_tokenize(text, language='english', preserve_line=False)`

Parameters
text (str) – text to split into words

language (str) – the model name in the Punkt corpus

preserve_line (bool) – A flag to decide whether to sentence tokenize the text or not.

stopwords.words ('english'): This method in NLTK is used to access the stopwords for a specific language from the NLTK corpus. It returns a list of common stopwords for the specified language.

Syntax:

`nltk.corpus.stopwords.words(languagename)`

Parameters
languagename (str) - the specified name of the language for which the stop words is to be listed.

Genism: The `genism` library, primarily known for its robust capabilities in topic modeling and document similarity tasks, can also be utilized for Bag-of-Words (BoW) representation.

In BoW, Gensim provides functionalities to preprocess text data, build a vocabulary, and convert documents into numerical vectors based on word occurrences.

While Gensim's primary focus lies in more advanced NLP tasks like topic modeling, its BoW implementation can serve as a foundational step in preprocessing text data for various downstream tasks such as classification, clustering, and information retrieval.

Some of the important methods of the genism library used for BOW are as follows:

gensim.corpora.Dictionary(documents): This method allows you to build a dictionary from a collection of documents, where each unique word in the corpus is assigned a unique integer ID.

`gensim.corpora.dictionary.Dictionary(documents=None)`

Parameters

documents (iterable of iterable of str, optional): This refers to documents to be used to initialize the mapping and collect corpus statistics.

`token2id`is an instance attribute under `gensim.corpora.dictionary.Dictionary(documents=None)` class which is used to retrieve the mapping of tokens (words) to their corresponding integer IDs in the dictionary.

gensim.corpora.Dictionary.doc2bow(document): This method is used to convert a document represented as a list of words (or tokens) into a bag-of-words format.

It is a part of the Gensim library, commonly used for topic modeling and text analysis tasks. This method takes a document as input and returns a list of tuples, where each tuple represents a word in the document along with its frequency (count) in the document.

`gensim.corpora.Dictionary.doc2bow(document)`

Parameter:

document: A list of words

Sample Python code of BOW model for TechTrend Reviews

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from gensim.corpora import Dictionary
nltk.download('punkt')
nltk.download('stopwords')
# Sample reviews
review1 = "The TechTrend X1 camera captures stunning photos, but the battery life could be better. I'm very impressed with the camera qualit
review2 = "I'm disappointed with the TechTrend X1 battery life, although the camera quality is exceptional. However, the camera features are
# Tokenization
tokens1 = word_tokenize(review1)
tokens2 = word_tokenize(review2)
# Stop word removal
```

```
stop_words = set(stopwords.words('english'))
filtered_tokens1 = [word for word in tokens1 if word.lower() not in stop_words]
filtered_tokens2 = [word for word in tokens2 if word.lower() not in stop_words]
# Create dictionary
documents = [filtered_tokens1, filtered_tokens2]
dictionary = Dictionary(documents)
# Generate bag-of-words vectors
bow_vector1 = dictionary.doc2bow(filtered_tokens1)
bow_vector2 = dictionary.doc2bow(filtered_tokens2)
# Print results
print("Filtered Tokens 1:", filtered_tokens1)
print("Filtered Tokens 2:", filtered_tokens2)
print("Dictionary:", dictionary.token2id)
print("BoW Vector 1:", bow_vector1)
print("BoW Vector 2:", bow_vector2)
```

Code output

```
Filtered Tokens 1: ['TechTrend', 'X1', 'camera', 'captures', 'stunning', 'photos', ',', 'battery', 'life', 'could',
"'m", 'impressed', 'camera', 'quality', '.']
Filtered Tokens 2: ["'m", 'disappointed', 'TechTrend', 'X1', 'battery', 'life', ',', 'although', 'camera', 'quality
'.', 'However', ',', 'camera', 'features', 'lacking', '.']
Dictionary: {"'m": 0, ',': 1, '.': 2, 'TechTrend': 3, 'X1': 4, 'battery': 5, 'better': 6, 'camera': 7, 'captures': 8
pressed': 10, 'life': 11, 'photos': 12, 'quality': 13, 'stunning': 14, 'However': 15, 'although': 16, 'disappointed
l': 18, 'features': 19, 'lacking': 20}
BoW Vector 1: [(0, 1), (1, 1), (2, 2), (3, 1), (4, 1), (5, 1), (6, 1), (7, 2), (8, 1), (9, 1), (10, 1), (11, 1), (12
4, 1)]
BoW Vector 2: [(0, 1), (1, 2), (2, 2), (3, 1), (4, 1), (5, 1), (7, 2), (11, 1), (13, 1), (15, 1), (16, 1), (17, 1),
(20, 1)]
```

The approach taken in the Python code provided above can be summarized as follows:

- Import the necessary libraries
- The commands `nltk.download('punkt')` and `nltk.download('stopwords')` are used to download the necessary resources for NLTK's word tokenizer (punkt) and the list of stopwords for various languages, respectively.
- There are two reviews assigned to two variables `review1` and `review2`.
- The `word_tokenize` function from NLTK is used to tokenize each review into a list of words or tokens.
- NLTK's list of English `stopwords` is used to remove common stopwords from the tokenized lists.
- The Gensim `Dictionary` class is then used to create a mapping between words and their integer IDs in the combined corpus.
- The `doc2bow` method from Gensim is used to convert each filtered token list into a bag-of-words representation. This representation counts the occurrences of each word in the document and stores them as tuples of (word_id, word_count).
- The filtered token lists, the generated dictionary, and the bag-of-words vectors for both reviews are printed to the console for inspection.

Conclusion:

In this reading, we gained a comprehensive understanding of text analysis, the Bag of Words model, and how these concepts are applied in real-world scenarios. We demonstrated their relevance through a practical use case and provided Python code to illustrate the implementation.

Author(s)

Lakshmi Holla

Other Contributors

Malika Singla