**What is text analysis?**

Text analysis is the process of using computer systems to read and understand human-written text for business insights.

Text analysis software can independently classify, sort, and extract information from text to identify patterns, relationships, sentiments, and other actionable knowledge.

We can use text analysis to efficiently and accurately process multiple text-based sources such as emails, documents, social media content, and product reviews, like a human would.

**Why is text analysis important?**

Businesses use text analysis to extract actionable insights from various unstructured data sources. They depend on feedback from sources like emails, social media, and customer survey responses to aid decision making.

With text analysis, we can get accurate information from the sources more quickly. The process is fully automated and consistent, and it displays data you can act on.

For example, using text analysis software allows you to immediately detect negative sentiment on social media posts so you can work to solve the problem

**Sentiment analysis**

Sentiment analysis or opinion mining uses text analysis methods to understand the opinion conveyed in a piece of text. We can use sentiment analysis of reviews, blogs, forums, and other online media to determine if your customers are happy with their purchases. Sentiment analysis helps you spot new trends, track sentiment changes, and tackle PR issues. By using sentiment analysis and identifying specific keywords, you can track changes in customer opinion and identify the root cause of the problem.

**Record management**

Text analysis leads to efficient management, categorization, and searches of documents. This includes automating patient record management, monitoring brand mentions, and detecting insurance fraud. For example, LexisNexis Legal & Professional uses text extraction to identify specific records among 200 million documents.

**Personalizing customer experience**

We can use text analysis software to process emails, reviews, chats, and other text-based correspondence. With insights about customers’ preferences, buying habits, and overall brand perception, you can tailor personalized experiences for different customer segments.

**­­**

**What is spaCy?**

spaCy is a free, open-source library for advance Natural Language Processing(NLP) in Python.

If you are working with a lot of text, you will eventually want to know more about it. For example, what’s it about? What do the words mean in context? Who is doing what to whom? What companies are products are mentioned? Which texts are similar to each other?

spaCy is designed specifically for production use and helps you build application that process and understand large volume of text. It can be used to build information extraction or natural language understanding systems or to pre-precess text for deep learning.

**spaCy Features & Text mining components**

|  |  |
| --- | --- |
| Name | Description |
| Tokenization | Segmenting text into words, punctuation marks etc. |
| Part-of-speech (POS) Tagging | Assigning word types to tokens like verb/noun/pro-noun |
| Dependency Parsing | Assigning syntactic dependency labels, describing the relations between individual tokens, like subject or object. |
| Lemmatization | Assigning the base form of words. For example, the lemma of “was” is “be”, and the lemma of “rats” is “rat” |
| Named Entity Recognition | Labelling names “real-word” objects like persons, companies or locations. |
| Similarity | Comparing words, text spans and documents and how similar they are to each other. |

**Installation:**

!pip install -U spacy

!python -m spacy download en\_core\_web\_sm

**Import Spacy in Python Code**

import spacy

text\_engine = spacy.load("en\_core\_web\_sm")

**Tokenization**

During processing, spaCy/text mining first tokenizes the text. i.e. segments it into words, punction and so on. This is done by applying rules specific to each language.

import spacy

text\_engine = spacy.load("en\_core\_web\_sm")

doc = text\_engine("Apple is looking to buying U.K. startup for $1 billion.")

for token in doc:

print(token.text)

**Part of Speech tags & dependencies**

After tokenization, spacy/text mining can parse and tag a given Doc. This is where the trained pipeline and it’s statistical model come in, which enable spaCy to make predictions of which tag or label most likely applies in this context.

A trained component includes binary data that is produced by showing a system enough examples for it to make predictions that generalize across the language – for example, a word following “the” in English is most likely a noun.

import spacy

text\_engine = spacy.load("en\_core\_web\_sm")

doc = text\_engine("Apple is looking to buying U.K. startup for $1 billion.")

for token in doc:

print(token.text, token.pos\_, token.dep\_, token.tag\_, token.is\_alpha, token.is\_stop , token.is\_shape, token.lemma\_)

POS: The simple Part of speech tag based on language

Dep: Syntactic dependency i.e relation between tokens

Tag: The detailed part-of-speech tag

Shape: The word shape – capitalization, punctuation, digits.

Is alpha : is the token an alpha character?

Is stop : is the token part of a stop work list.

Lemma: the base form of the word.

**Named Entities**

A named entity is a “read-world object that’s assigned a name – for example, a person, a country, a product or book title. spaCy can recognize various types of named entities in a document, by asking the model for prediction.

import spacy

text\_engine = spacy.load("en\_core\_web\_sm")

doc = text\_engine("Apple is looking to buying U.K. startup for $1 billion.")

for ent in doc.ents:

print(ent.text, ent.start\_char, ent.end\_char, ent.label\_)

Text: the original entity text

Start: Index of start of entity in the Doc

End: Index of end of the entity in the Doc

Label: Entity label i.e. type