How to Work with Unstructured Data in Python

By **James Keenan** on **February 17, 2023**

Read more about author James Keenan .

All our online actions generate data. Even if we don't write posts, comment, or upload other content, we leave our traces by being silent observers. This leads to predictable results – according to Statista , the amount of data generated globally is expected to surpass 180 zettabytes in 2025. On the one hand, having many resources to make data-based decisions is brilliant. What's a bit limiting: Most generated data is unstructured data, and such datasets have no predetermined model.

For better or for worse, by 2025, 80% of all data will be unstructured, according to IDC predictions . And that's the key reason we need to learn how to work with unstructured datasets.

Dealing with Unstructured Data

Why is it hard to work with unstructured data? Well, such datasets don't comply with a predefined format, making it hard to analyze or find use cases for direct usage. Yet, unstructured data can provide valuable insights and help to formulate data-driven strategies.

Manually analyzing unstructured data is time-consuming and expensive; hence, such a process is more prone to human error and bias. Plus, it's not scalable, which is a big no-no for businesses focusing on growth. Luckily, there are ways to transform unstructured data into a feasible format.

While it's relatively easy to manage structured data using everyday tools like Excel, Google Sheets, and relational databases , unstructured data management requires more advanced tools, complex rules, Python libraries, and techniques to transform it into quantifiable data.

Steps to Structure Unstructured Data

Unstructured data processing is more complex; however, the process can be less frustrating if you follow some exact steps. They can differ depending on the initial goal of analysis, the desired outcome, the software, and other resources.

1. Find Where to Store Your Data

Everything starts with the question: Where to store the data? The choice is either public or in-house storage hardware. The latter offers complete control over data and its security; however, it requires

The public clouds, on the other hand, enable remote collaboration and are cost-effective and more scalable: If you need more space, you can upgrade the plan. Therefore, it’s an excellent option for startups and small companies with limited IT resources, time, or funds to build internal storage systems.

2. Clean Your Data

By its nature, unstructured data is messy and sometimes includes typos, HTML tags, punctuation, hashtags, special characters, banner ads, and whatnot. Thus, it’s necessary to perform data pre-processing, commonly referred to as “data cleaning,” before jumping on the actual structuring process. Data cleaning entails various methods, such as reducing noise, removing irrelevant data, and splitting data into more understandable pieces. You can perform data cleaning with Excel, Python, and other programming languages or with special data cleaning tools.

3. Categorize the Collected Data

Another step in the data organizing process is to define relationships between various units in the dataset. Sorting the entities into categories helps measure which data is essential for your analysis. You can classify your data based on content, context, or a user according to your needs. For example, if you’re scraping used vehicle sites, you might need to differentiate which elements are comments and which are technical information. If your datasets are incredibly complex, you’ll need a professional data scientist to help structure everything correctly. For non-complex datasets, you can classify data using Python.

4. Design a Pre-annotator

After classifying data, complete the annotation part. This process of labeling data helps machines better understand the context and patterns behind the data to provide relevant results. Such a process can be handled by hand, making it time-consuming and fallible. You can automate this process by designing a pre-annotator with the help of Python dictionaries.

Setting a Dictionary and Rules

Python dictionaries can also help you retrieve the required values from the dataset. Setting a dictionary will create arrays of already grouped data units. In other words, dictionaries help you develop keys for data values. For example, when the keys are matched with particular values, the annotator can recognize that the mentioned word “Ford” is a car (in this case, “car” is a key, and “Ford”

To avoid mistakes in the structuration process, define the rules to prevent random associations. For example, whenever the annotator spots the car name, it should identify the serial number next to it. Thus, an annotation tool should mark the number next to a vehicle's name as its serial number.

5. Sort Data with Python

After finishing the previous step, you need to sort out and match certain pieces of information while removing irrelevant content. This can be done with the help of Python regular expressions – sequences of characters that can group and extract patterns in the text.

Tokenize Data

The following process is to split a large chunk of text into words or sentences. You may use a Natural Language Toolkit (NLTK) to deal with it. For that, you need to install this Python library and perform word or sentence tokenization, depending on your preferences.

Process Data Using Stemming and Lemmatization

Another step in natural language processing (NLP) coding is stemming and lemmatization. Simply put, they both shape words according to their root. The first one is simpler and faster – it just cuts down the stem; for example, "cooking" becomes "cook." Lemmatization is a bit slower and more sophisticated process. It composes the world's inflected forms into a single entity for analysis. In this case, the word "went" would be grouped with "go" even though they don't share the same root.

Those two processes are not only part of natural language processing but machine learning as well. Therefore, stemming and lemmatization are the text pre-processing techniques that help analysis tools understand and process text data at scale, later transforming the results into valuable insights.

6. Visualize the Received Results

The final and most important step in structuring data is convenient visualization. Concise data representation helps transform mundane spreadsheets into charts, reports, or graphs. All this can be done in Python using libraries like Matplotlib, Seaborn, and others, depending on the databases and visualization preferences.

Use Cases of Structuring Data

Not sure how data structuring can be helpful for your business? Here are some ideas:

ahead requires processing more data, which is mostly unstructured.

- **Document clustering:** Organize documents and retrieve and filter information automatically. In the long term, it helps to make the search process faster, more efficient, and cost-effective.
- **Information retrieval:** Map documents to prevent the loss of important information.

In a Nutshell

Working with unstructured data is not easy; however, investing in it as early as possible is essential. Luckily, Python can be actively used during the process and help automate the integral parts.

Share on: