DATA AND
ARTIFICIAL INTELLIGENCE
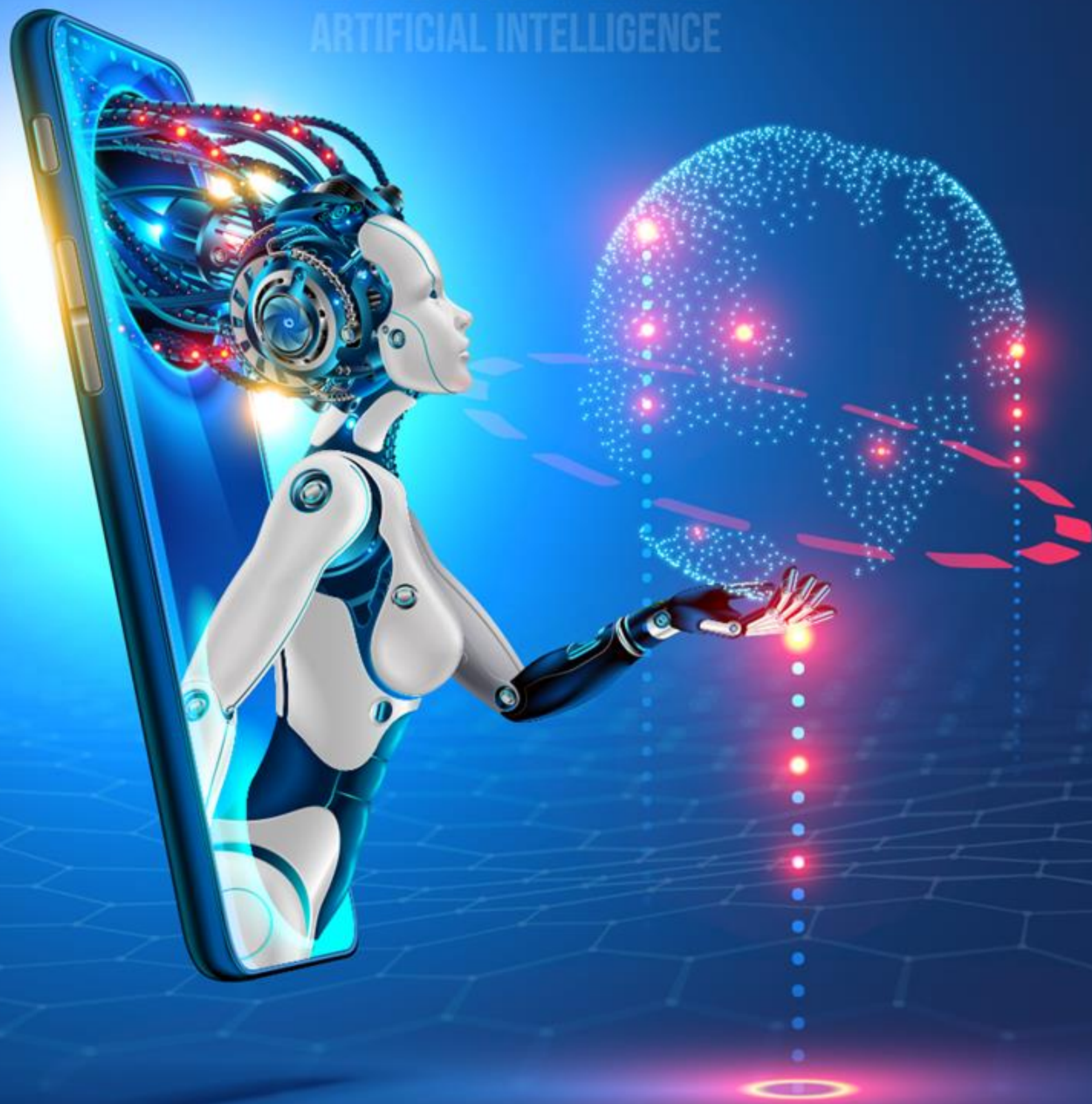
**Data Science with Python**

simplilearn

# Natural Language Processing (NLP) with SciKit Learn

simplilearn

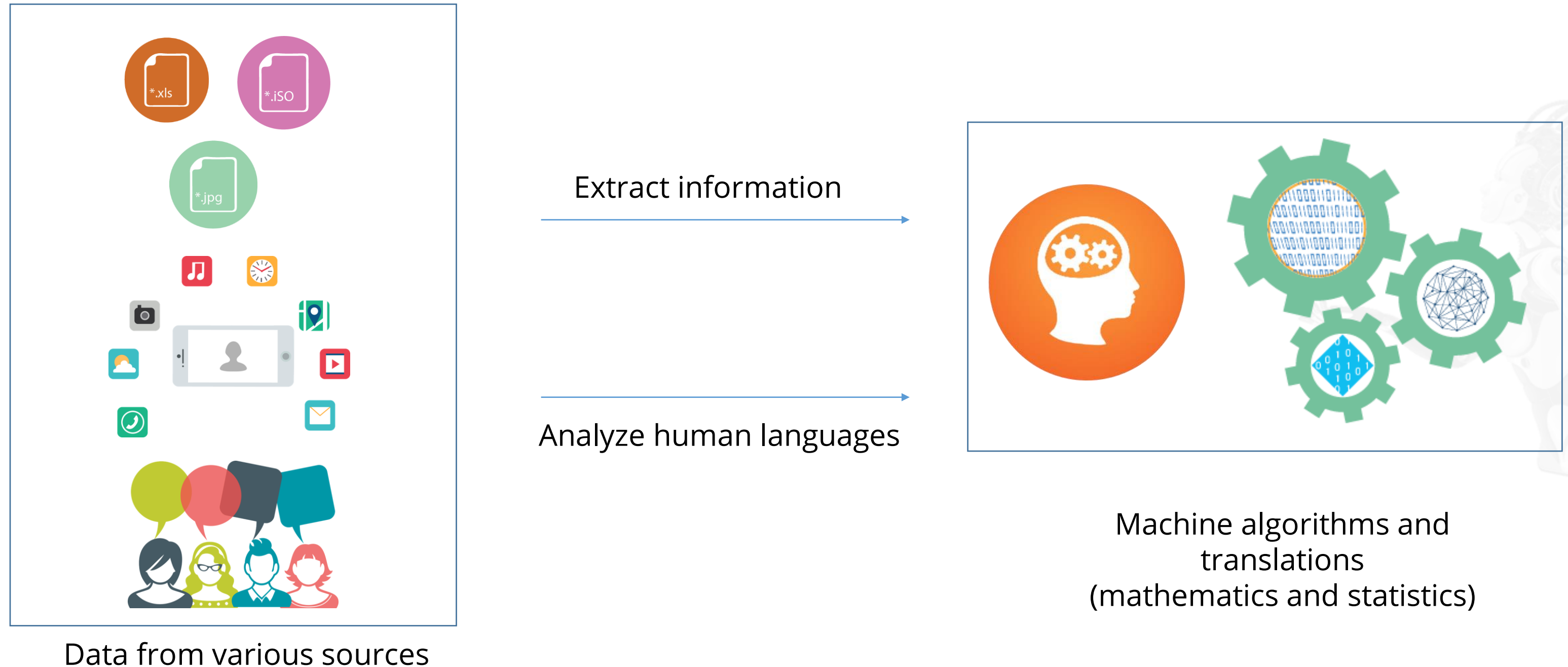# Learning Objectives

By the end of this lesson, you will be able to:

- Define natural language processing

- Explain the importance of natural language processing

- List the applications using natural language processing

- Outline the modules to load content and category

- Apply feature extraction techniques

- Implement the approaches of natural language processing

# Introduction to Natural Language Processing

# Natural Language Processing (NLP)

Natural language processing is an automated way to understand and analyze natural human languages and extract information from such data by applying machine algorithms.

Extract information

Analyze human languages

Data from various sources

Machine algorithms and translations
(mathematics and statistics)

# Natural Language Processing

It is also referred to as, the field of computer science or AI to extract the linguistics information from the underlying data.

Extract the linguistics information

# Why Natural Language Processing

The world is now connected globally due to the advancement of technology and devices.

Analyzing tons of data

Identifying various languages

Applying quantitative analysis

Handling ambiguities
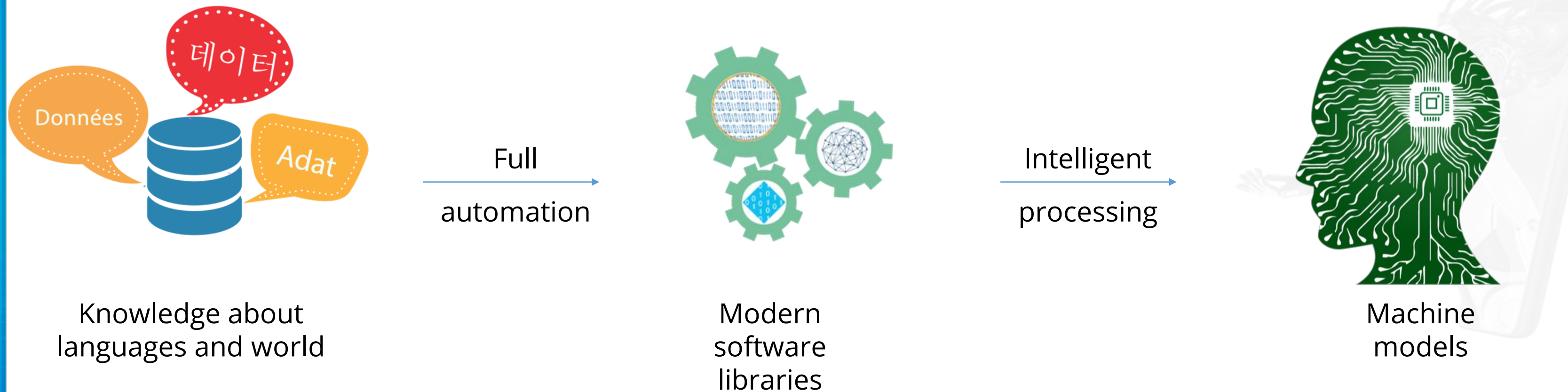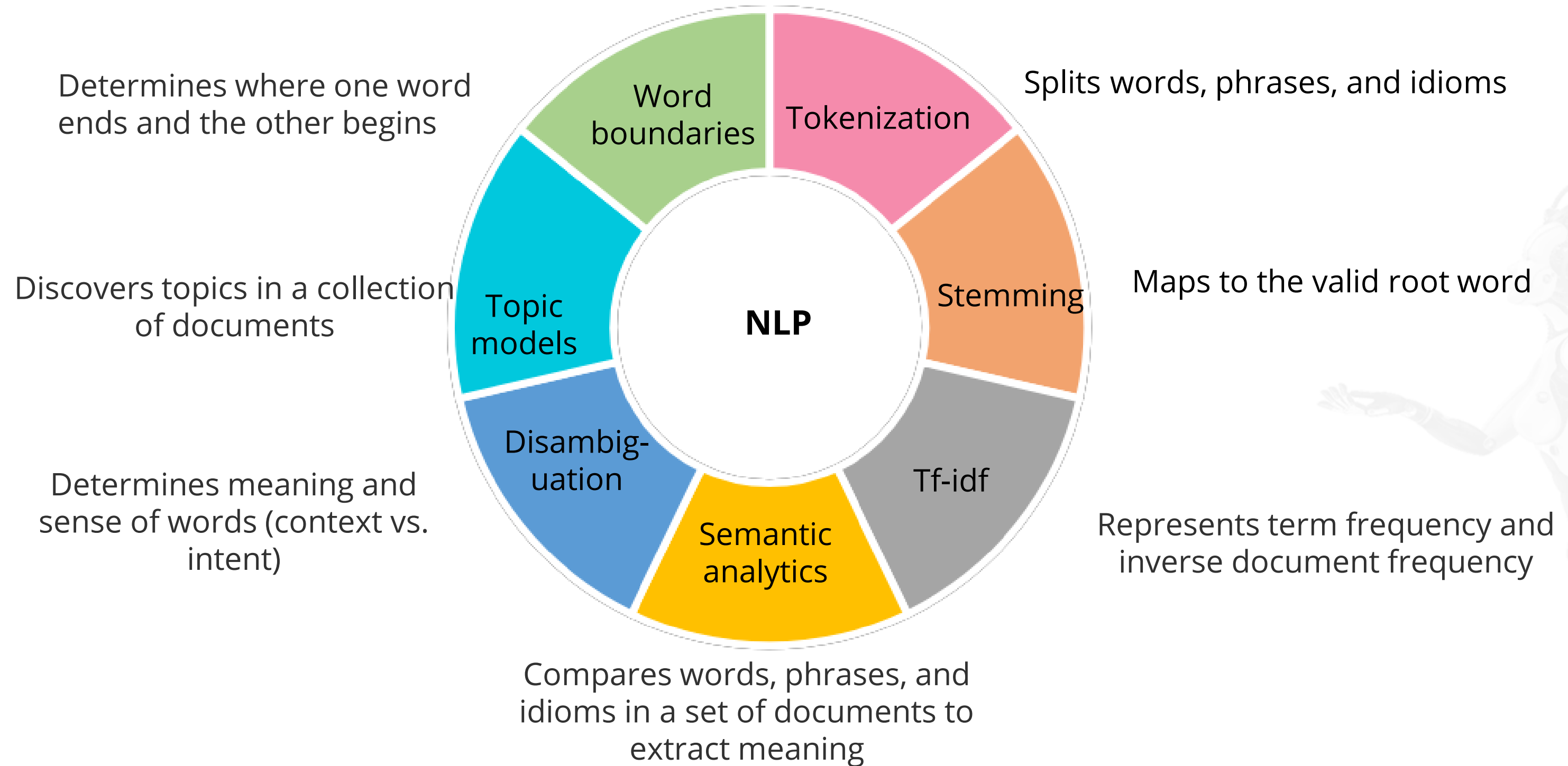
# Why Natural Language Processing

NLP can achieve full automation by using modern software libraries, modules, and packages.



Knowledge about
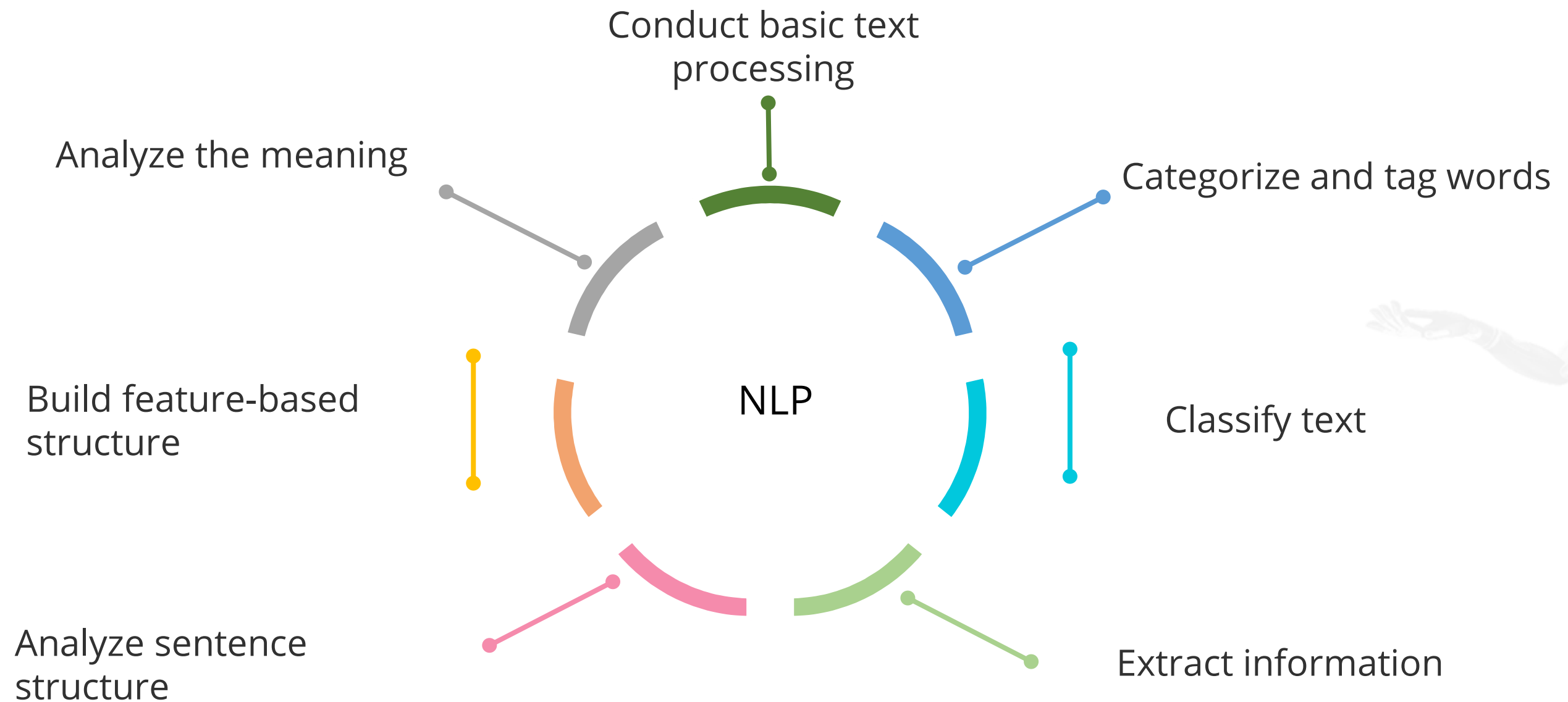languages and world

Full
automation

Modern
software
libraries

Intelligent
processing

Machine
models

# NLP Terminology



Determines where one word ends and the other begins

Word boundaries

Tokenization

Splits words, phrases, and idioms

Discovers topics in a collection of documents

Topic models

NLP

Stemming

Maps to the valid root word

Determines meaning and sense of words (context vs. intent)

Disambig-uation

Tf-idf

Represents term frequency and inverse document frequency

Semantic analytics

Compares words, phrases, and idioms in a set of documents to extract meaning

# NLP Approach for Text Data

Let us look at the Natural Language Processing approaches to analyze text data.

Conduct basic text processing

Analyze the meaning

Categorize and tag words

NLP

Build feature-based structure

Classify text

Analyze sentence structure

Extract information

**Problem Statement:** Demonstrate the installation of NLP environment

**Access:** Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

# Sentence Analysis

**Problem Statement:** Demonstrate how to perform the sentence analysis

**Access:** Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

# Applications of NLP

# Applications of NLP
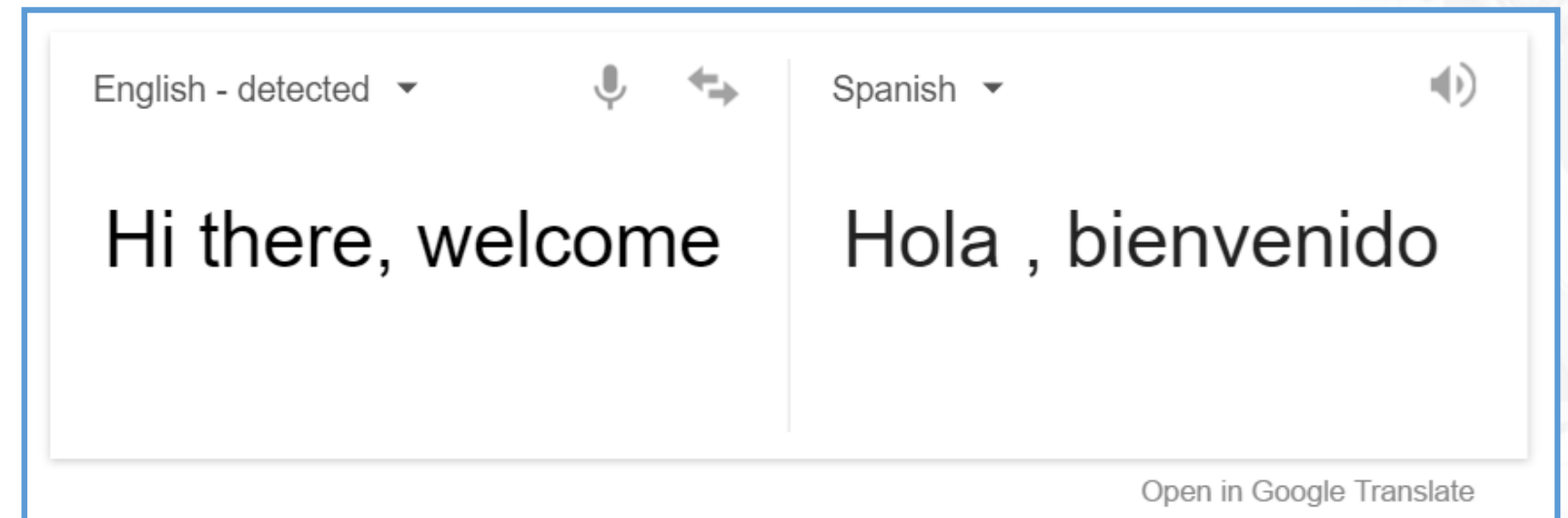
**Machine Translation**

**Speech Recognition**

**Sentiment Analysis**

Machine translation is used to translate one language into another. Google Translate is an example. It uses NLP to translate the input data from one language to another.



English - detected | Spanish

Hi there, welcome | Hola , bienvenido

Open in Google Translate

# Applications of NLP

**Machine Translation**

**Speech Recognition**

**Sentiment Analysis**

The speech recognition application understands human speech and uses it as input information. It is useful for applications like Siri, Google Now, and Microsoft Cortana.
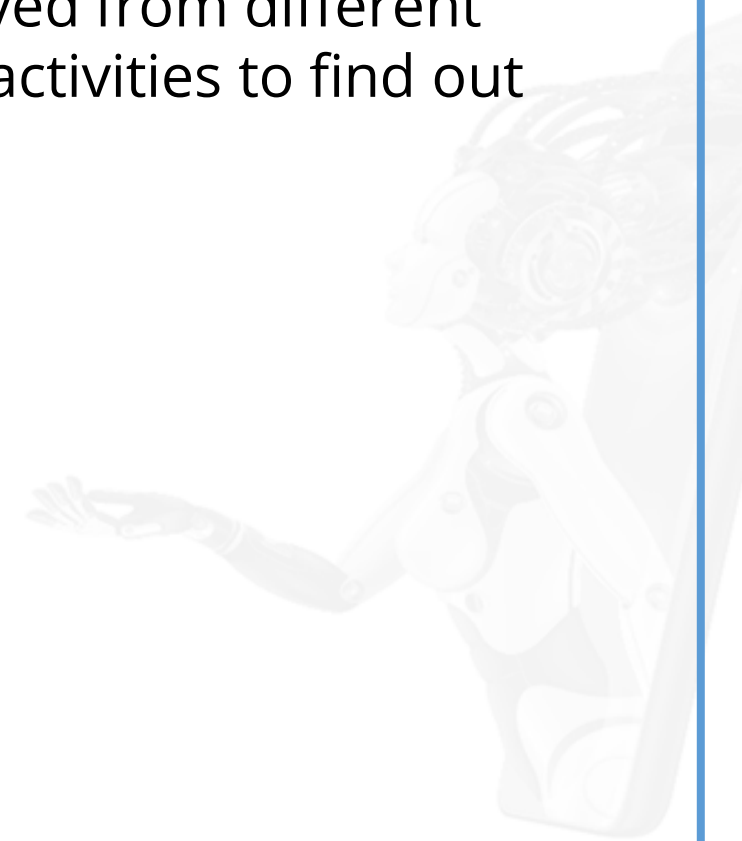


Hi. I'm Cortana.
Ask me a question!
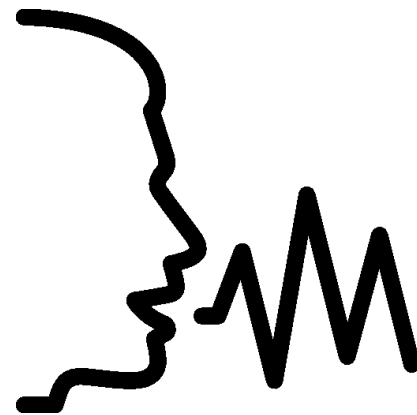
# Applications of NLP

**Machine Translation**
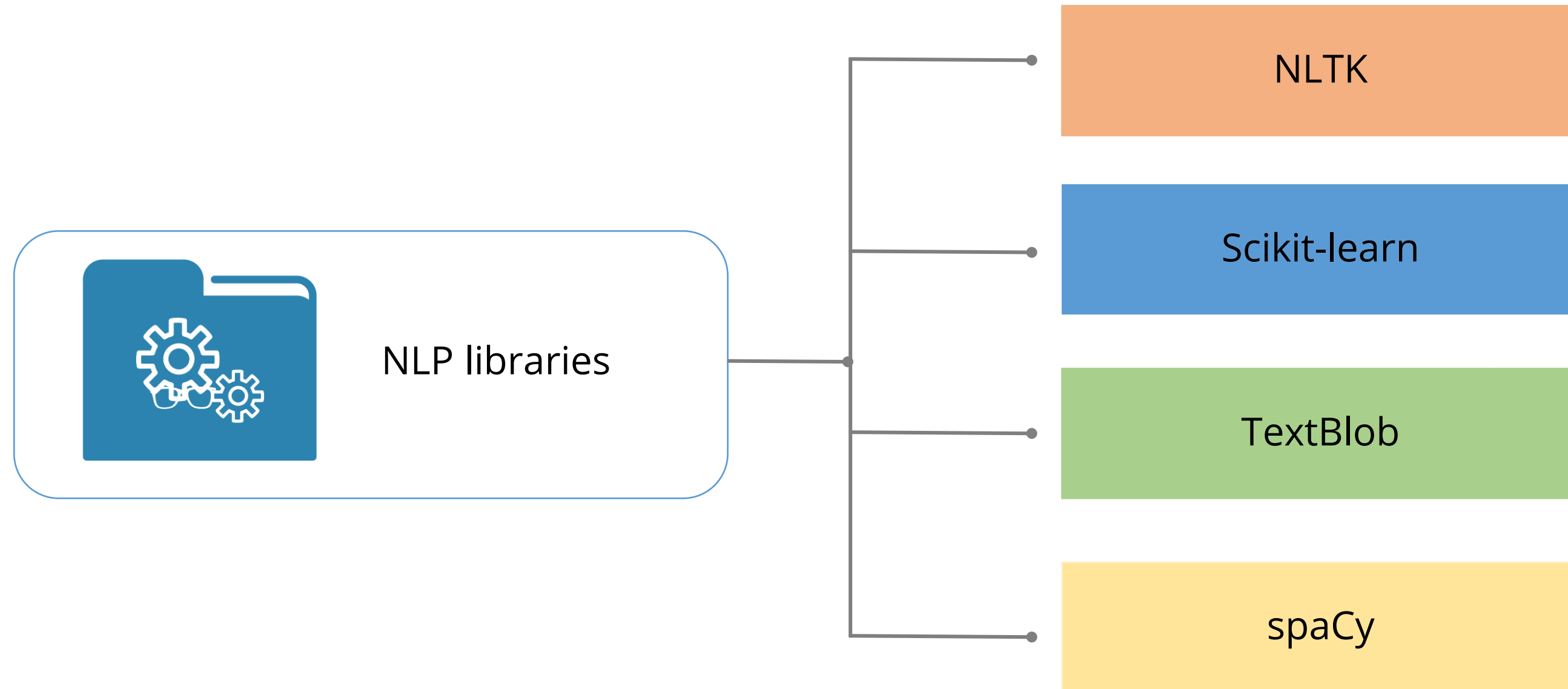
**Speech Recognition**

**Sentiment Analysis**

Sentiment analysis is achieved by processing tons of data received from different interfaces and sources. For example, NLP uses all social media activities to find out the popular topic of discussion or importance.

# Major NLP Libraries



NLP libraries

- NLTK
- Scikit-learn
- TextBlob
- spaCy

# The Scikit-Learn Approach

It is a very powerful library with a set of modules to process and analyze natural language data, such as text and images, and extract information using machine learning algorithms.
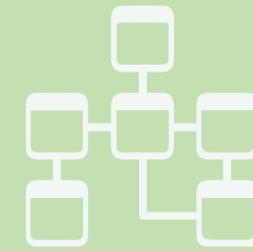
### Built-in module

Contains built-in modules to load the dataset's content and categories.

### Feature extraction

A way to extract information from data which can be text or images.
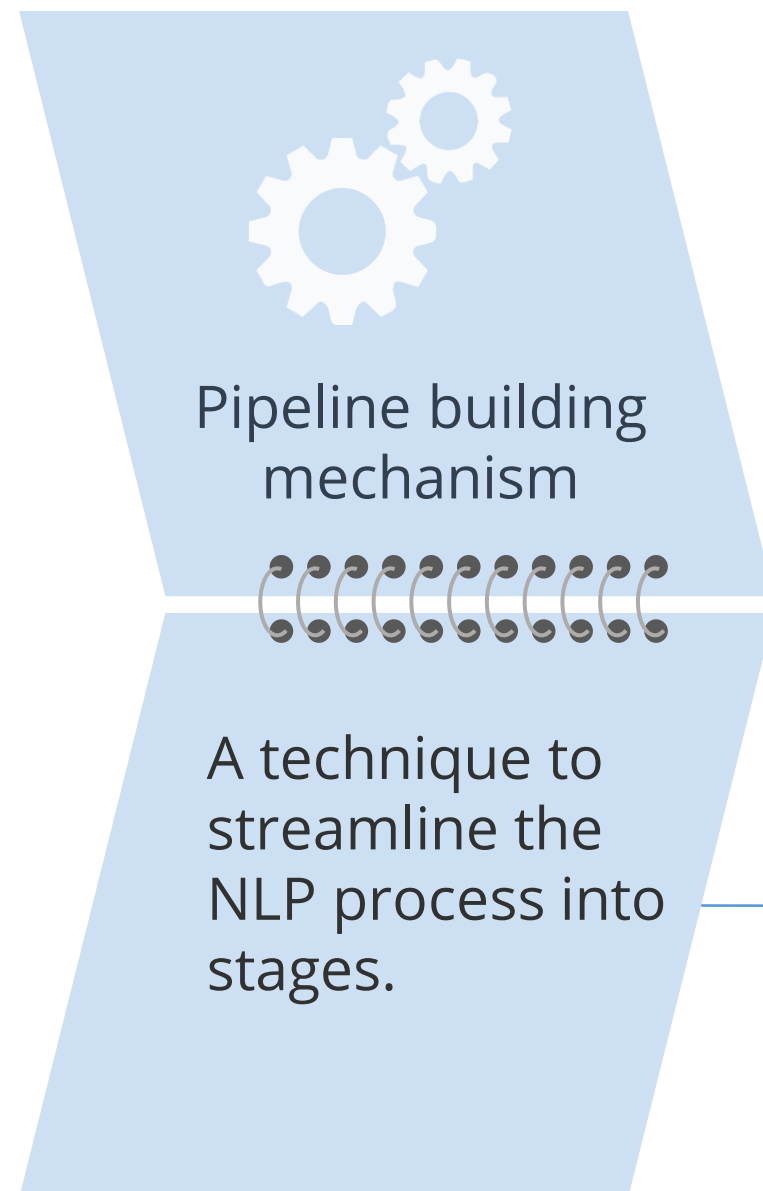
### Model training

Analyzes the content based on particular categories and then trains them according to a specific model.

# The Scikit-Learn Approach

It is a very powerful library with a set of modules to process and analyze natural language data, such as texts and images, and extract information using machine learning algorithms.

## Pipeline building mechanism

## A technique to streamline the NLP process into stages.

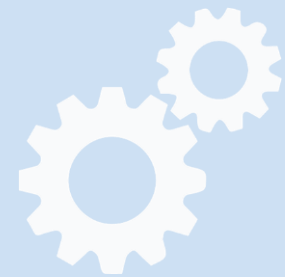Various stages of pipeline learning

1. Vectorization
2. Transformation
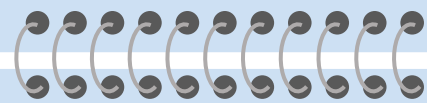3. Model training and application

# The Scikit-Learn Approach

It is a very powerful library with a set of modules to process and analyze natural language data, such as texts and images, and extract information using machine learning algorithms.
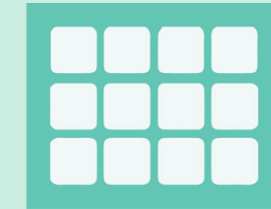
**Pipeline building mechanism**

A technique in Scikit-learn approach to streamline the NLP process into stages.

**Performance optimization**

In this stage we train the models to optimize the overall process.

**Grid search for finding good parameters**
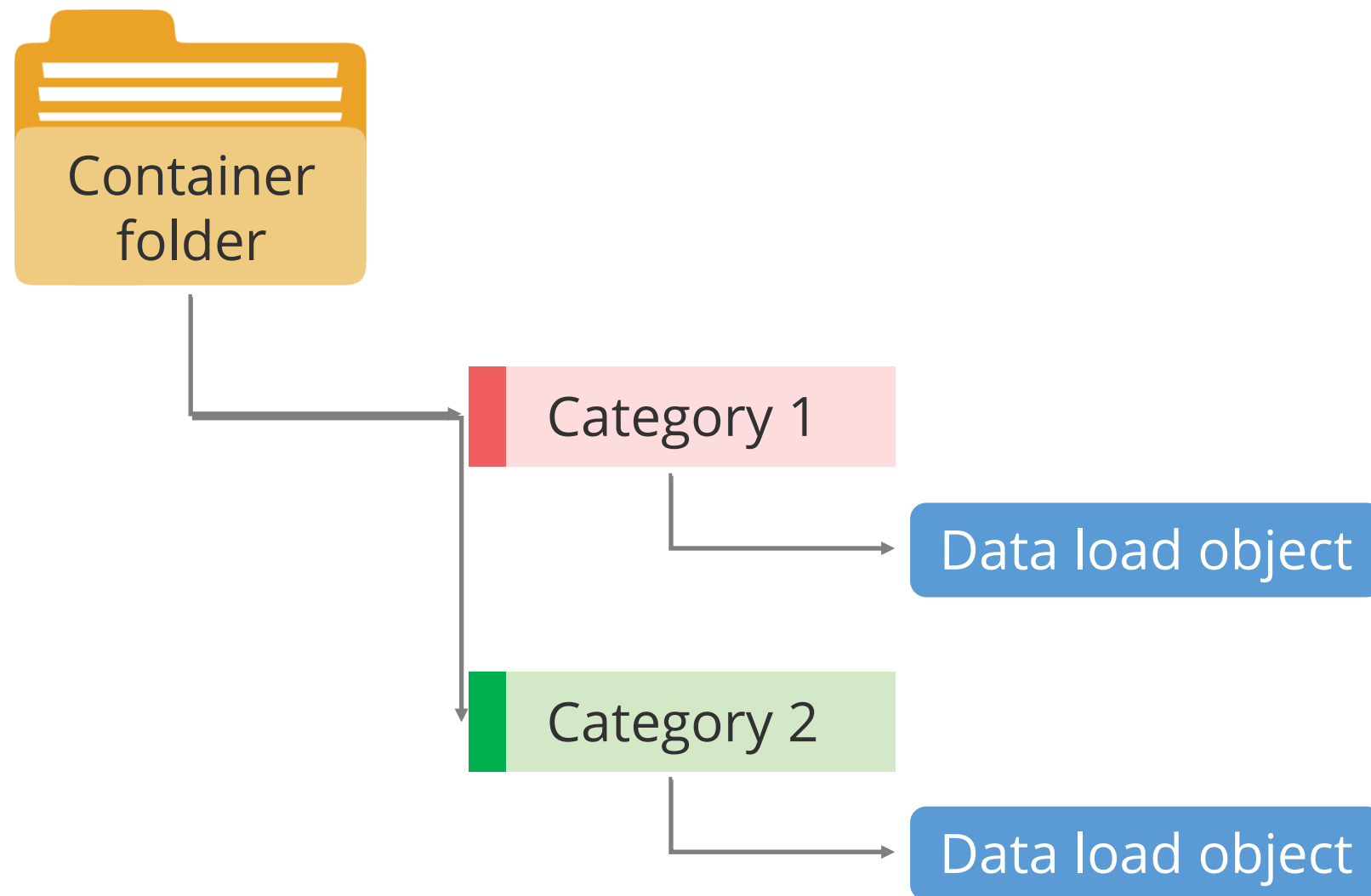
It's a powerful way to search parameters affecting the outcome for model training purposes.

simplilearn

Modules to Load Content and Category

# Modules to Load Content and Category

Scikit-learn has many built-in datasets. There are several methods to load these datasets with the help of a data load object.



```
In [ ]: #Load dataset
        load_data = sklearn.datasets.load_files()
```

# Modules to Load Content and Category

The text files are loaded with categories as subfolder names.

Container folder

Category 1

TXT

Category 2

TXT

Extract features

NumPy array
SciPy matrix
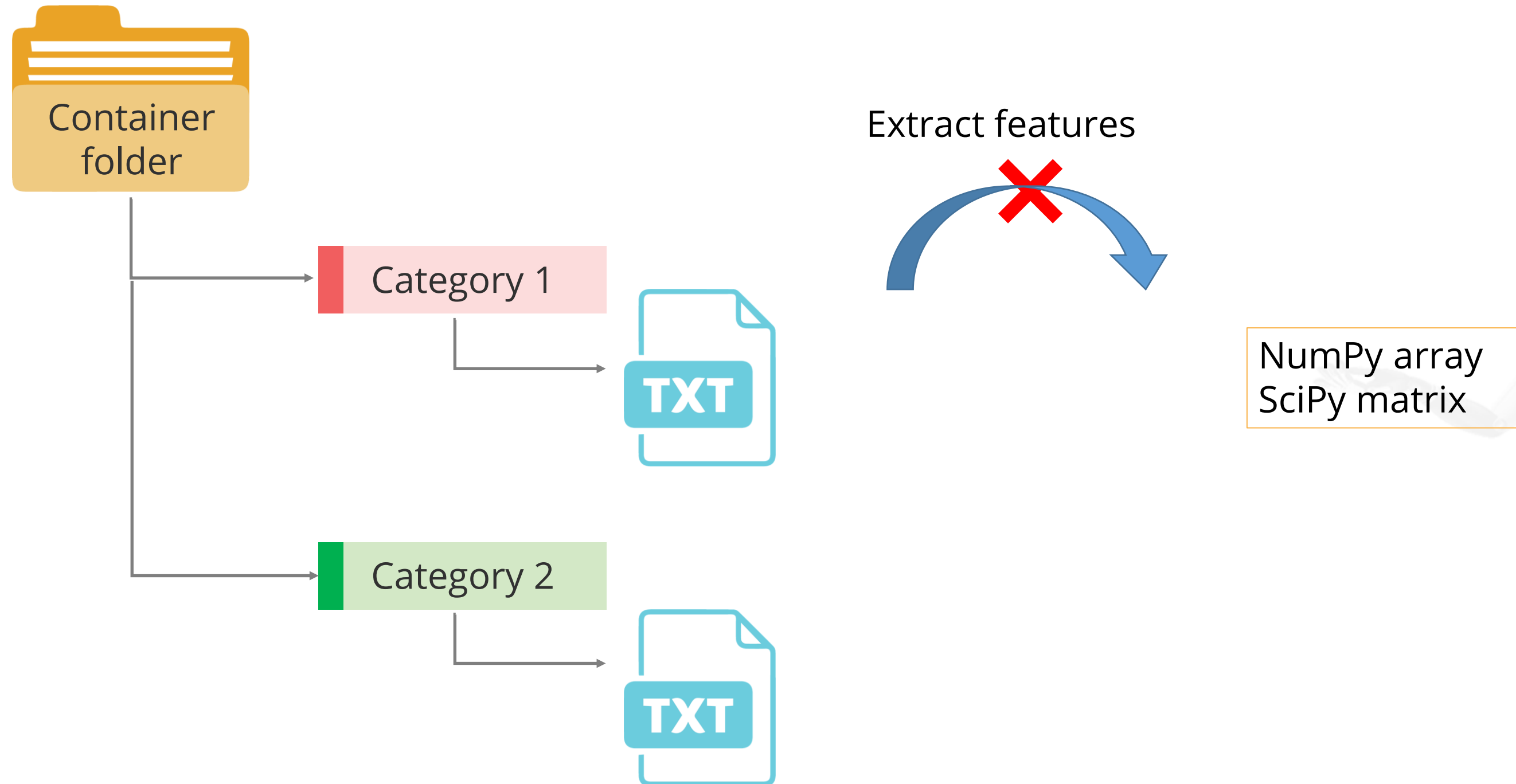
# Modules to Load Content and Category

In [ ] :

```
#Build a feature extraction transformer
From sklearn.feature_extraction.text import <appropriate transformer>
```

# Modules to Load Content and Category

The attributes of a data load object are:

| Data load object | Attributes | | |
|---|---|---|---|
| | | Bunch | Contains fields and can be accessed as dict keys or an object |
| | | Target names | Has the list of requested categories |
| | | Data | Refers to an attribute in the memory |
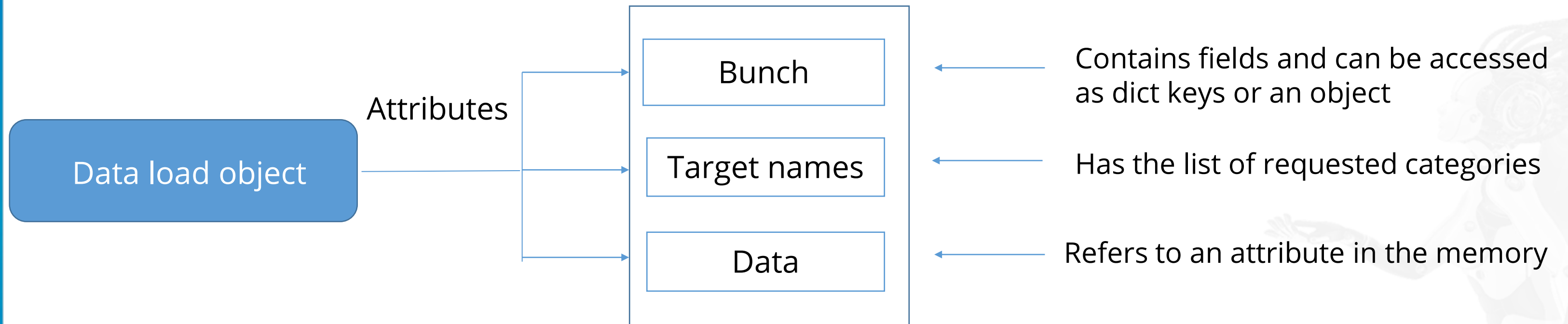
# Modules to Load Content and Category

The example shows how a dataset can be loaded using Scikit-learn:

In [1]:
```
#load dataset
from sklearn.datasets import load_digits
```
← Import the dataset

In [2]:
```
#create object of the loaded dataset
digit_dataset = load_digits()
```
← Load dataset

In [3]:
```
# use built in descr function to describe dataset
digit_dataset.DESCR
```
← Describe the dataset

Out[3]: "Optical Recognition of Handwritten Digits Data Set\n================================================
==\n\nNotes\n-----\nData Set Characteristics:\n    :Number of Instances: 5620\n    :Number of Attribut
es: 64\n    :Attribute Information: 8x8 image of integer pixels in the range 0..16.\n    :Missing Attr
ibute Values: None\n    :Creator: E. Alpaydin (alpaydin '@' boun.edu.tr)\n    :Date: July; 1998\n\nThi
s is a copy of the test set of the UCI ML hand-written digits datasets\nhttp://archive.ics.uci.edu/ml/
datasets/Optical+Recognition+of+Handwritten+Digits\n\nThe data set contains images of hand-written dig
its: 10 classes where\neach class refers to a digit.\n\nPreprocessing programs made available by NIST
were used to extract\nnormalized bitmaps of handwritten digits from a preprinted form. From a\ntotal o
f 43 people, 30 contributed to the training set and different 13\nto the test set. 32x32 bitmaps are d
ivided into nonoverlapping blocks of\n4x4 and the number of on pixels are counted in each block. This

# Modules to Load Content and Category

Let us see how functions like type, .data, and .target help in analyzing a dataset.

```
In [4]:  #view type of dataset
         type(digit_dataset)

Out[4]:  sklearn.datasets.base.Bunch
```
View type of dataset

```
In [5]:  #view data
         digit_dataset.data

Out[5]:  array([[  0.,    0.,    5., ...,    0.,    0.,    0.],
                [  0.,    0.,    0., ...,   10.,    0.,    0.],
                [  0.,    0.,    0., ...,   16.,    9.,    0.],
                ...,
                [  0.,    0.,    1., ...,    6.,    0.,    0.],
                [  0.,    0.,    2., ...,   12.,    0.,    0.],
                [  0.,    0.,   10., ...,   12.,    1.,    0.]])
```
View data

```
In [6]:  #view target
         digit_dataset.target

Out[6]:  array([0, 1, 2, ..., 8, 9, 8])
```
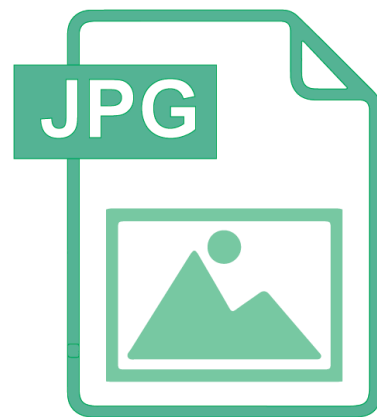View target

# Feature Extraction

Feature extraction is a technique to convert the content into the numerical vectors to perform machine learning.

**TXT**

For example: Large datasets or documents

Text feature extraction

**JPG**
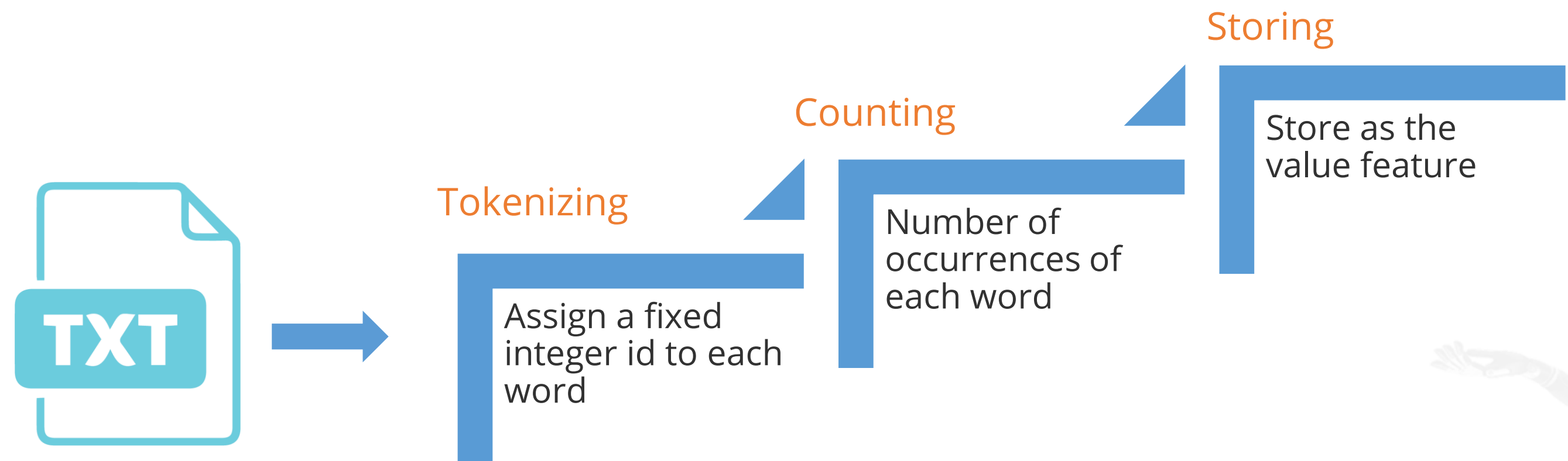
For example: Patch extraction, hierarchical clustering

Image feature extraction

# Bag of Words

# Bag of Words

Bag of words is used to convert text data into numerical feature vectors with a fixed size.

**Storing**

Store as the value feature

**Counting**

Number of occurrences of each word

**Tokenizing**

Assign a fixed integer id to each word

**TXT**

Corpus of document

|  | Token 1 | Token 2 | Token 3 | Token 4 |
|---|---|---|---|---|
| Document 1 | 42 | 32 | 119 | 3 |
| Document 2 | 1118 | 0 | 0 | 89 |
| Document 3 | 0 | 0 | 0 | 55 |

simplilearn

# CountVectorizer Class Signature

Class

**class**

**sklearn.feature_extraction.text.CountVectorizer**

*(input='content', encoding='utf-8',*

*decode_error='strict', strip_accents=None,*

*lowercase=True, preprocessor=None,*

*tokenizer=None, stop_words=None,*

*token_pattern='(?u)\b\w\w+\b', ngram_range=(1, 1),*

*analyzer='word', max_df=1.0, min_df=1,*

*max_features=None, vocabulary=None,*

*binary=False, dtype=<class 'numpy.int64'>)*

Specifies number of components to keep

Encoding used to decode the input

Removes accents

Built-in stop words list

Min Threshold

Max Threshold

File name or sequence of strings

Overrides string tokenizer

# Bags of Words

**Problem Statement:** Demonstrate the Bag of Words technique

**Access:** Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

ASSISTED PRACTICE

# Text Feature Extraction Considerations

# Text Feature Extraction Considerations

| | |
|---|---|
| **Sparse** | This utility deals with sparse matrix while storing them in memory. Sparse data is commonly noticed when it comes to extracting feature values, especially for large document datasets. |
| **Vectorizer** | It implements tokenization and occurrence. Words with minimum two letters get tokenized. We can use the analyzer function to vectorize the text data. |
| **Tf-idf** | It is a term weighing utility for term frequency and inverse document frequency. Term frequency indicates the frequency of a particular term in the document. Inverse document frequency is a factor which diminishes the weight of terms that occur frequently. |
| **Decoding** | This utility can decode text files if their encoding is specified. |

# Model Training

An important task in model training is to identify the right model for the given dataset. The choice of model completely depends on the type of dataset.

**Supervised**

Models predict the outcome of new observations and datasets, and classify documents based on the features and response of a given dataset.

Example: Naïve Bayes, SVM, linear regression, K-NN neighbors

**Unsupervised**

Models identify patterns in the data and extract its structure. They are also used to group documents using clustering algorithms.

Example: K-means

# Naïve Bayes Classifier

It is the most basic technique for classification of text.

**Advantages:**
- It is efficient as it uses limited CPU and memory.
- It is fast as the model training takes less time.

**Uses:**
- Naïve Bayes is used for sentiment analysis, email spam detection, categorization of documents, and language detection.
- Multinomial Naïve Bayes is used when multiple occurrences of the words matter.

# Naïve Bayes Classifier

Let us take a look at the signature of the multinomial Naïve Bayes classifier:

Class

Learn Class prior probabilities

```
class sklearn.naive_bayes.MultinomialNB(alpha=1.0, fit_prior=True, class_prior=None)
```

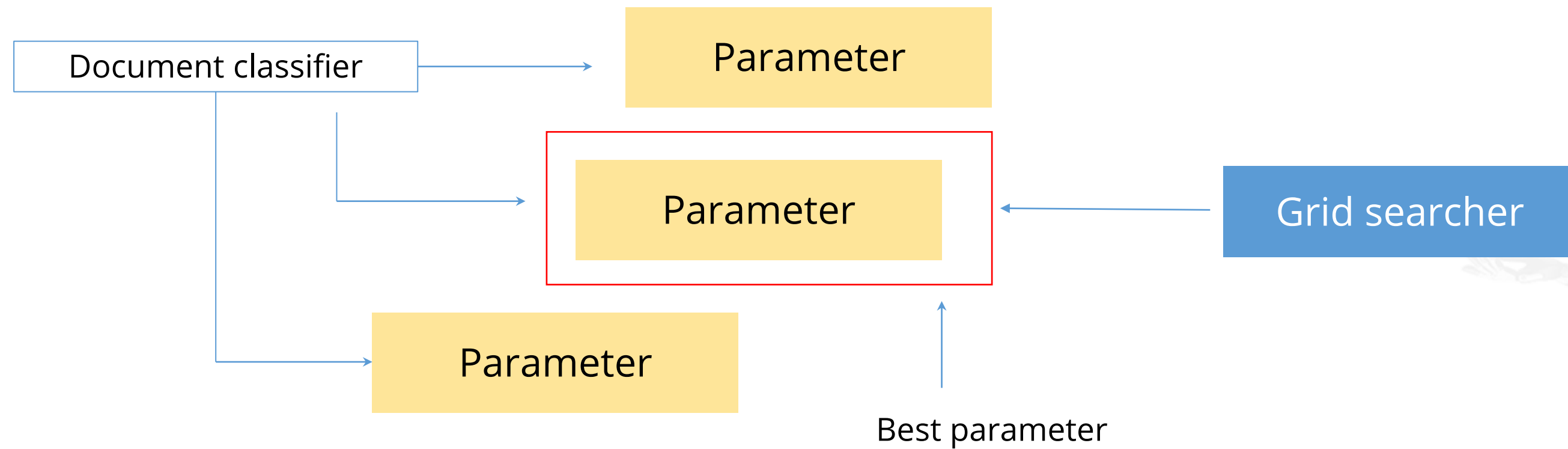Smoothing parameter
(0 for no smoothing)

Prior probabilities of the classes

# Grid Search and Multiple Parameters

Document classifiers can have many parameters. A Grid approach helps to search the best parameters for model training and predicting the outcome accurately.

Category 1

Extract features of a document

Document classifier

Category 2

# Grid Search and Multiple Parameters

Document classifier

Parameter
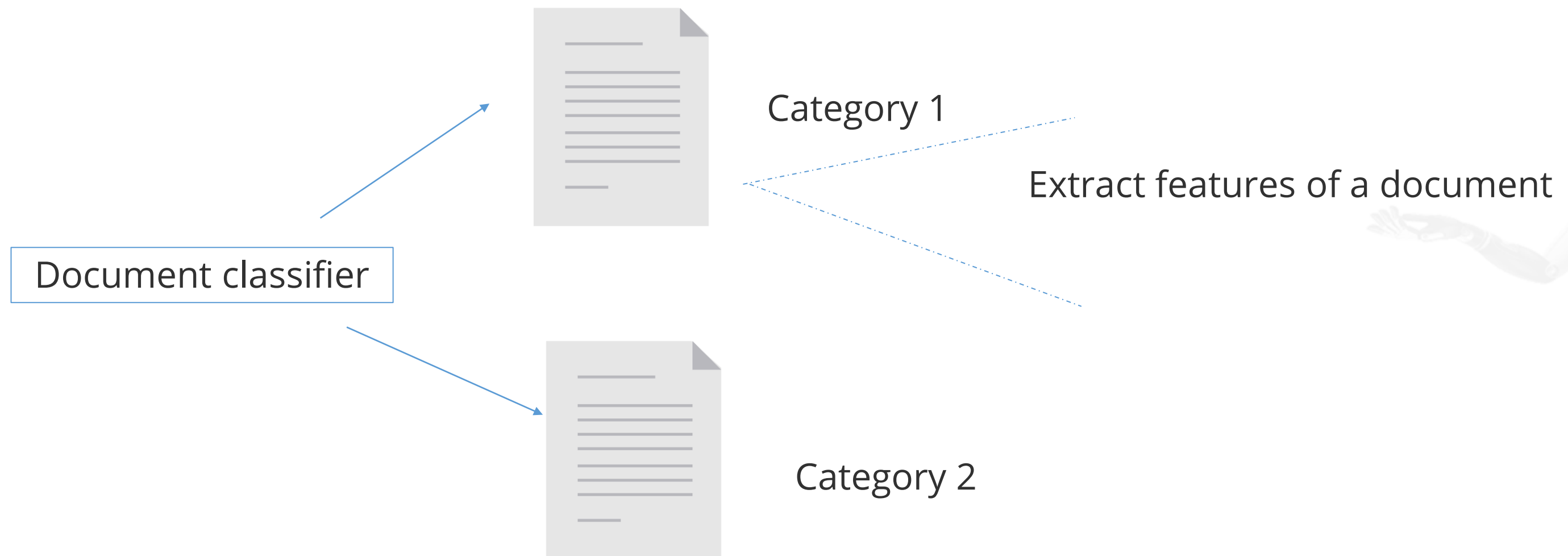
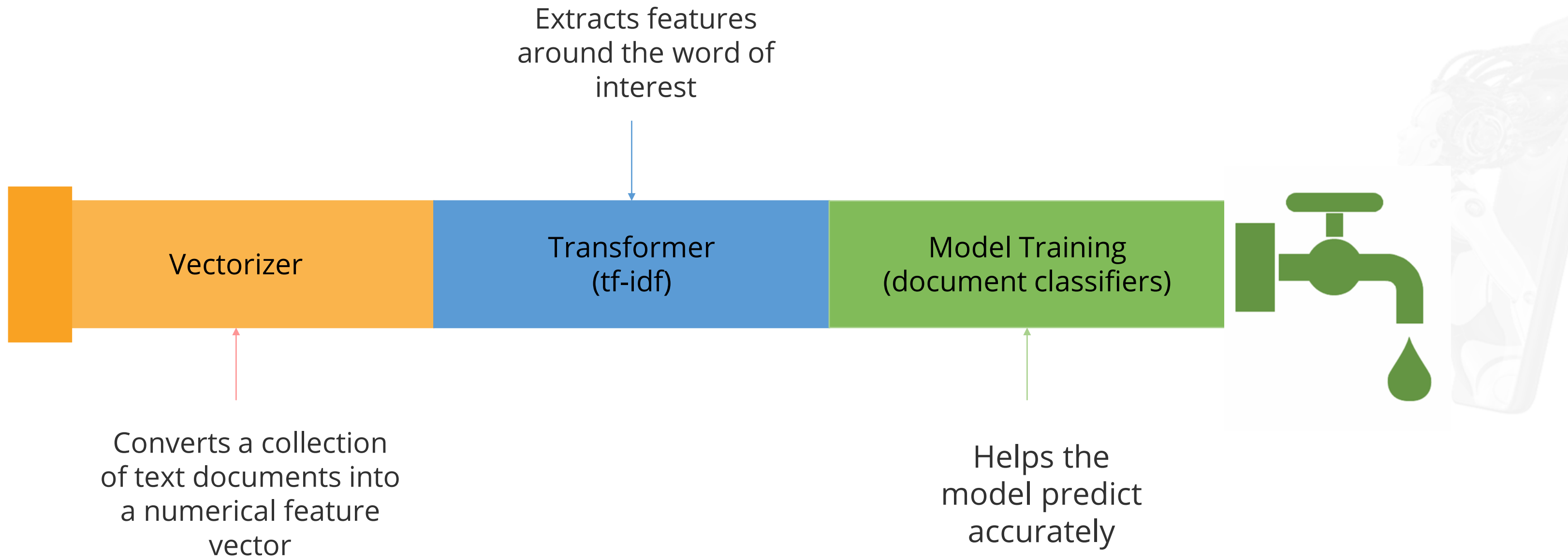Parameter

Parameter

Best parameter

Grid searcher

# Grid Search and Multiple Parameters

In grid search mechanism, the whole dataset can be divided into multiple grids and a search can be run on the entire grid or a combination of grids.

Grid searcher

Best parameter

Parameter 1

Parameter 2

Parameter 3
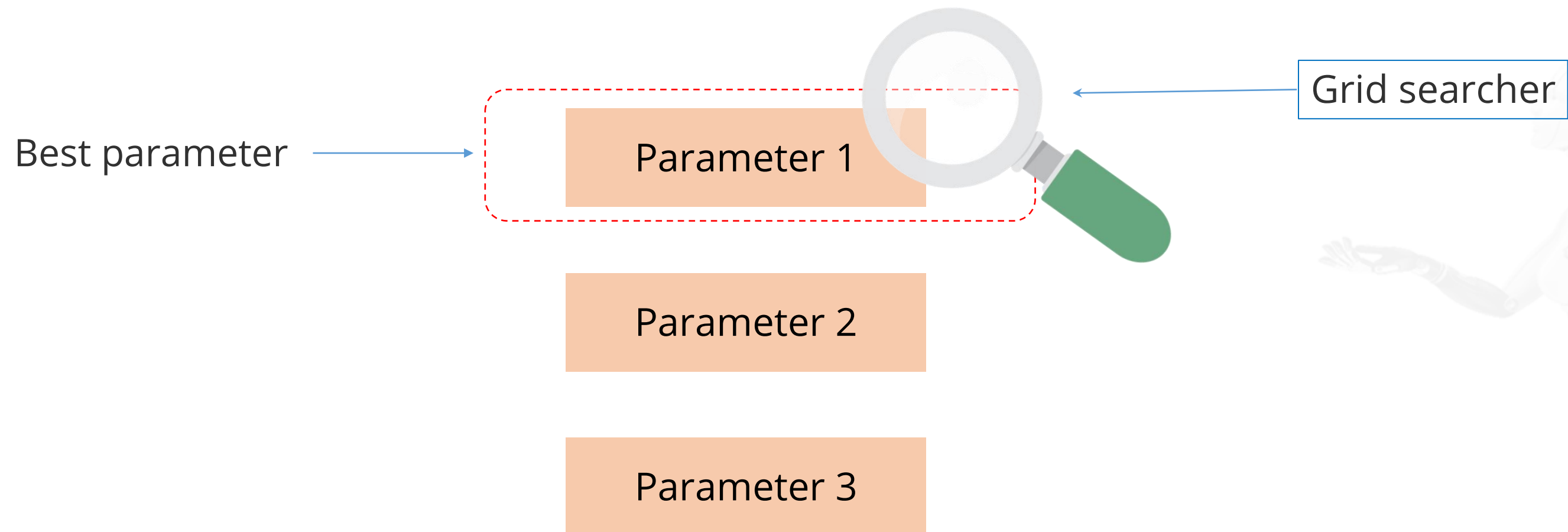
# Pipeline

A pipeline is a combination of vectorizers, transformers, and model training.

Extracts features
around the word of
interest

| Vectorizer | Transformer (tf-idf) | Model Training (document classifiers) |

Converts a collection
of text documents into
a numerical feature
vector

Helps the
model predict
accurately

# Pipeline and Grid Search

**Problem Statement:** Demonstrate the Pipeline and Grid Search technique.

**Access:** Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

# Analyzing the Spam Collection Dataset

**Problem Statement:**

Analyze the given Spam Collection dataset to:
1. View information on the spam data
2. View the length of messages,
3. Define a function to eliminate stop words
4. Apply Bag of Words
5. Apply tf-idf transformer
6. Detect Spam with Naïve Bayes model

UNASSISTED PRACTICE

# Analyzing the Spam Collection Dataset

**Instructions on performing the assignment:**
•Download the Spam Collection dataset from the "Resource" tab. Upload it using the right syntax to use and analyze it.

**Common instructions:**
•If you are new to Python, download the "Anaconda Installation Instructions" document from the "Resources" tab to view the steps for installing Anaconda and the Jupyter notebook.
•Download the "Assignment 01" notebook and upload it on the Jupyter notebook to access it.
•Follow the provided cues to complete the assignment.

# Analyzing the Sentiment Dataset using NLP

**Problem Statement:**

Analyze the Sentiment dataset using NLP to:
1. View the observations
2. Verify the length of the messages and add it as a new column
3. Apply a transformer and fit the data in the bag of words
4. Print the shape for the transformer
5. Check the model for predicted and expected values

# Analyzing the Sentiment Dataset using NLP

**Instructions on performing the assignment:**
- Download the Sentiment dataset from the "Resource" tab. Upload it to your Jupyter notebook to work on it.

**Common instructions:**
- If you are new to Python, download the "Anaconda Installation Instructions" document from the "Resources" tab to view the steps for installing Anaconda and the Jupyter notebook.
- Download the "Assignment 02" notebook and upload it on the Jupyter notebook to access it.
- Follow the provided cues to complete the assignment.

UNASSISTED PRACTICE

# Key Takeaways

You are now able to:

- Define natural language processing

- Explain the importance of natural language processing

- List the applications using natural language processing

- Outline the modules to load content and category

- Apply feature extraction techniques

- Implement the approaches of natural language processing

**Knowledge Check**

**In NLP, tokenization is a way to _____.**

a.    Find the grammar of the text

b.    Analyze the sentence structure

c.    Find ambiguities

d.    Split text data into words, phrases, and idioms

**In NLP, tokenization is a way to _____.**

a. Find the grammar of the text

b. Analyze the sentence structure

c. Find ambiguities

d. Split text data into words, phrases, and idioms

The correct answer is  **d**

**Splitting text data into words, phrases, and idioms is known as tokenization and each individual word is known as token.**

**Knowledge Check**

**2**

**What is the tf-idf value in a document?**

a. Directly proportional to the number of times a word appears

b. Inversely proportional to the number of times a word appears

c. Offset by frequency of the words in corpus

d. Increase with frequency of the words in corpus

**What is the tf-idf value in a document?**

a. Directly proportional to the number of times a word appears

b. Inversely proportional to the number of times a word appears

c. Offset by frequency of the words in corpus

d. Increase with frequency of the words in corpus

The correct answer is **a,c**

**td-idf value reflects how important a word is to a document. It is directly proportional to the number of times a word appears and is offset by frequency of the words in corpus.**

simplilearn

**Knowledge Check**

**3**

**In grid search, if n_jobs = -1, then which of the following is correct?**

a. Uses only 1 CPU core

b. Detects all installed cores and uses them all

c. Searches for only one parameter

d. All parameters will be searched on a given grid

**In grid search, if n_jobs = -1, then which of the following is correct?**

a.   Uses only 1 CPU core

b.   Detects all installed cores and uses them all

c.   Searches for only one parameter

d.   All parameters will be searched on a given grid

The correct answer is   **b**

**Detects all installed cores on the machine and uses all of them.**

**Identify the correct example of Topic Modeling from the following options:**

a.  Machine translation

b.  Speech recognition

c.  News aggregators

d.  Sentiment analysis

**Knowledge Check**

**4**

**Identify the correct example of Topic Modeling from the following options:**

a. Machine translation

b. Speech recognition

c. News aggregators

d. Sentiment analysis

The correct answer is **c**

**'Topic model' is statistical modeling and used to find latent groupings in the documents based upon the words. For example, news aggregators.**

**How do we save memory while operating on Bag of Words which typically contain high-dimensional sparse datasets?**

a. Distribute datasets in several blocks or chunks

b. Store only non-zero parts of the feature vectors

c. Flatten the dataset

d. Decode them

**Knowledge Check**

**5**

**How do we save memory while operating on Bag of Words which typically contain high-dimensional sparse datasets?**

a.  Distribute datasets in several blocks or chunks

b.  Store only non-zero parts of the feature vectors

c.  Flatten the dataset

d.  Decode them
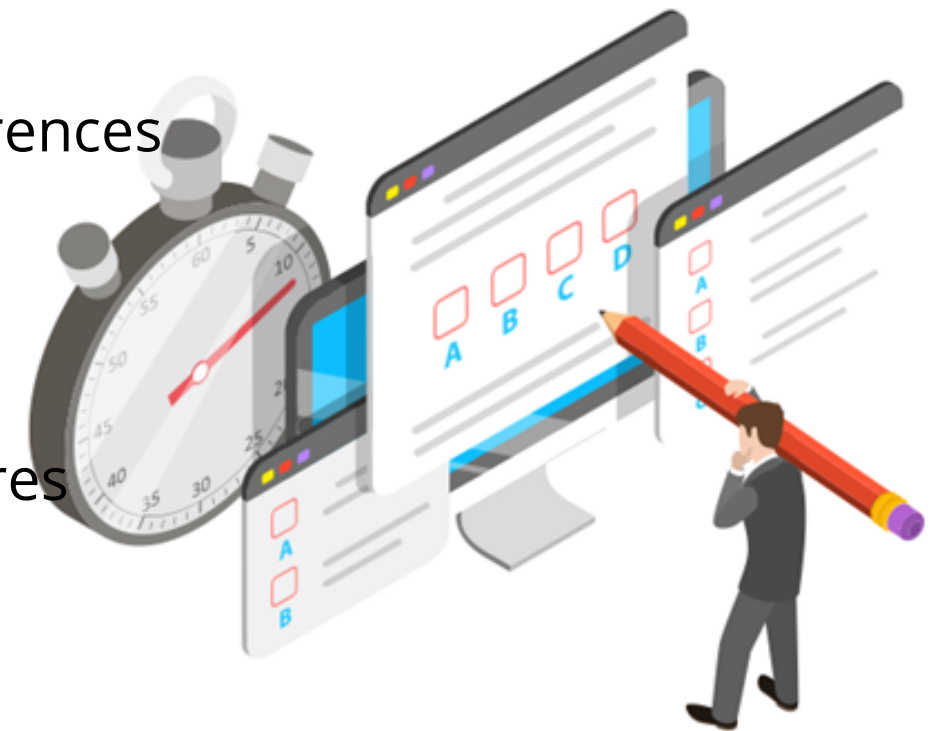
The correct answer is  **b**

**In features vector, there will be several values with zeros. The best way to save memory is to store only non-zero parts of the feature vectors.**

**What is the function of the sub-module feature_extraction.text.CountVectorizer?**

a. Convert a collection of text documents to a matrix of token counts

b. Convert a collection of text documents to a matrix of token occurrences

c. Transform a count matrix to a normalized form

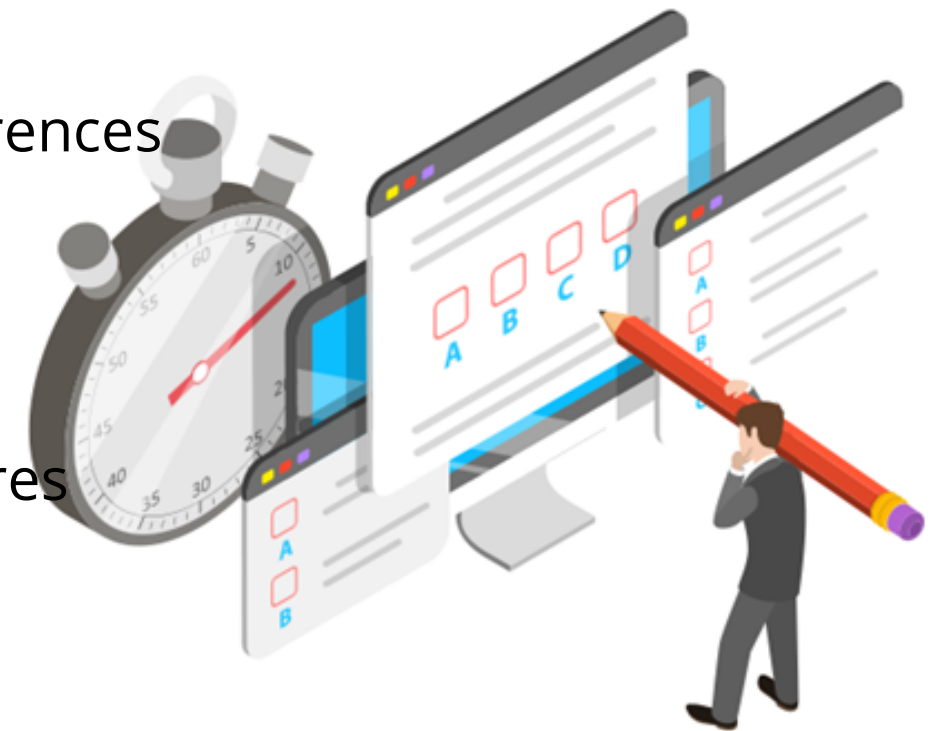d. Convert a collection of raw documents to a matrix of TF-IDF features

**What is the function of the sub-module feature_extraction.text.CountVectorizer?**

a. Convert a collection of text documents to a matrix of token counts

b. Convert a collection of text documents to a matrix of token occurrences

c. Transform a count matrix to a normalized form

d. Convert a collection of raw documents to a matrix of TF-IDF features

The correct answer is **a**

**The function of the sub-module feature_extraction.text.CountVectorizer is to convert a collection of text documents to a matrix of token counts.**