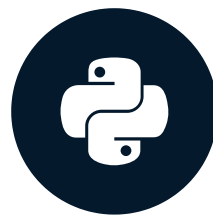


Introduction au traitement du langage naturel

NATURAL LANGUAGE PROCESSING (NLP) EN PYTHON



Fouad Trad
Machine Learning Engineer

Faites connaissance avec l'instructeur...



Fouad Trad

- Ingénieur en Machine Learning
- Chercheur scientifique
- NLP en cybersécurité et santé

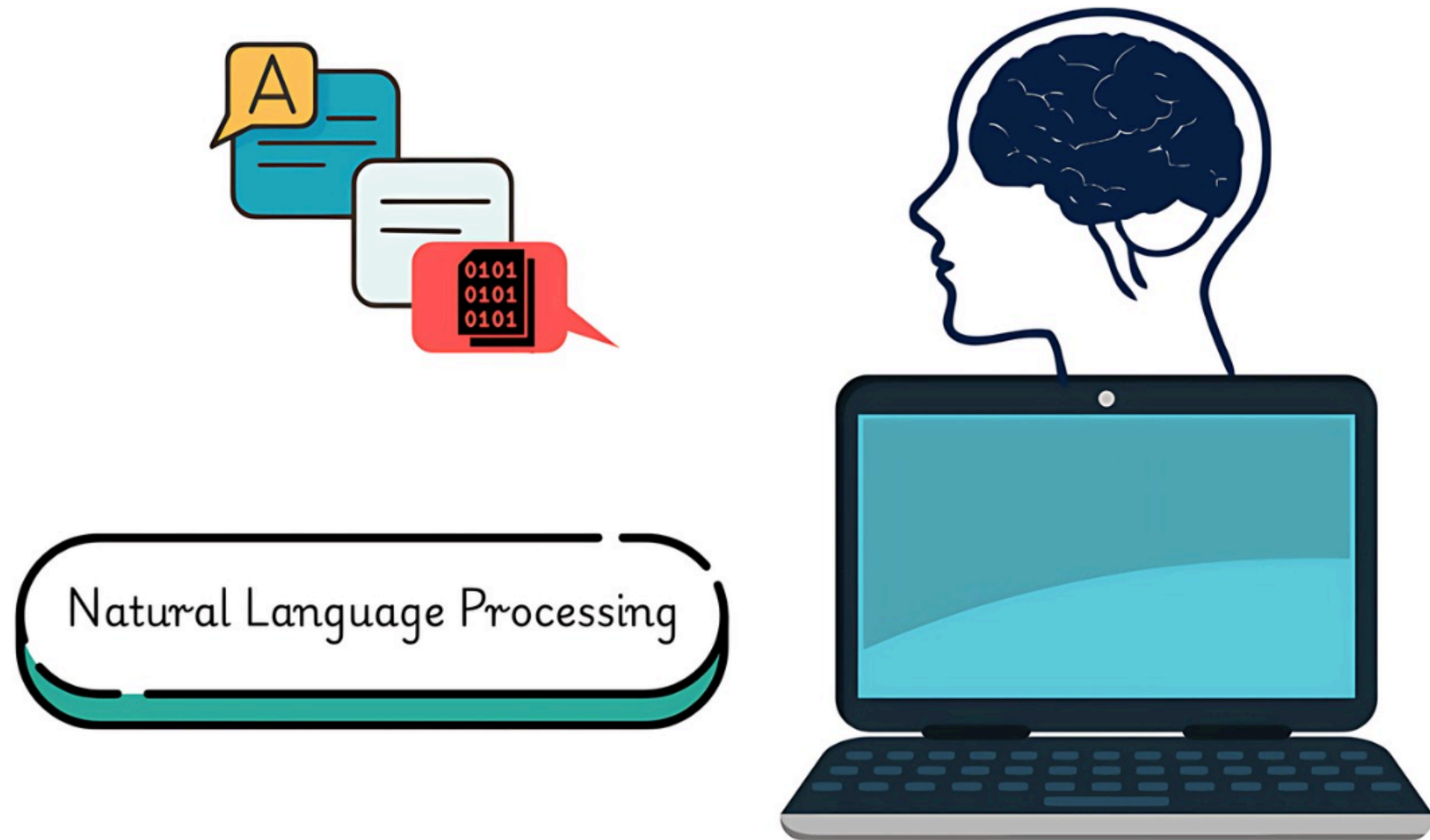
Qu'est-ce que le NLP ?

- Le langage est notre principal moyen de communiquer
- Les ordinateurs ne comprennent pas notre langue

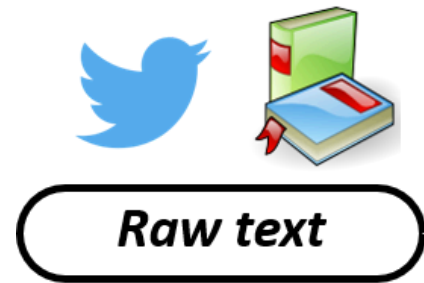


Qu'est-ce que le NLP ?

Permet aux ordinateurs d'analyser le langage humain



Flux de travail en NLP



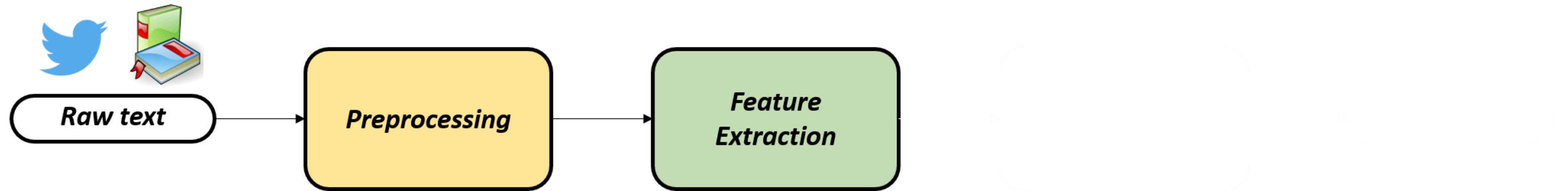
- Texte brut : d'un gazouillis à un paragraphe de livre

Flux de travail en NLP



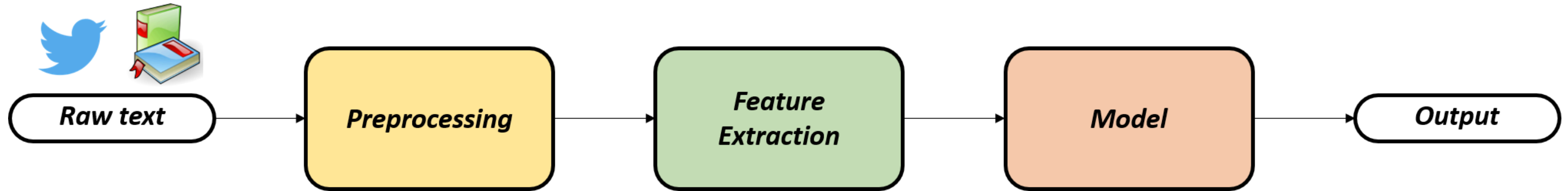
- **Texte brut** : d'un gazouillis à un paragraphe de livre
- **Prétraitement** : nettoyer le texte et retirer les éléments inutiles

Flux de travail en NLP



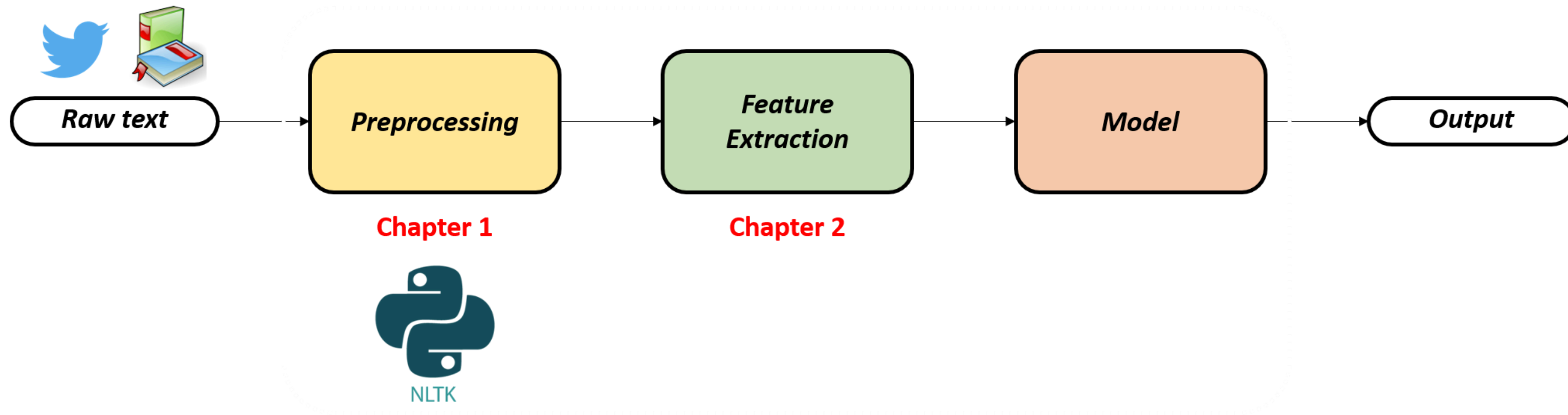
- **Texte brut** : d'un gazouillis à un paragraphe de livre
- **Prétraitement** : nettoyer le texte et retirer les éléments inutiles
- **Extraction de caractéristiques** : convertir le texte en nombres

Flux de travail en NLP

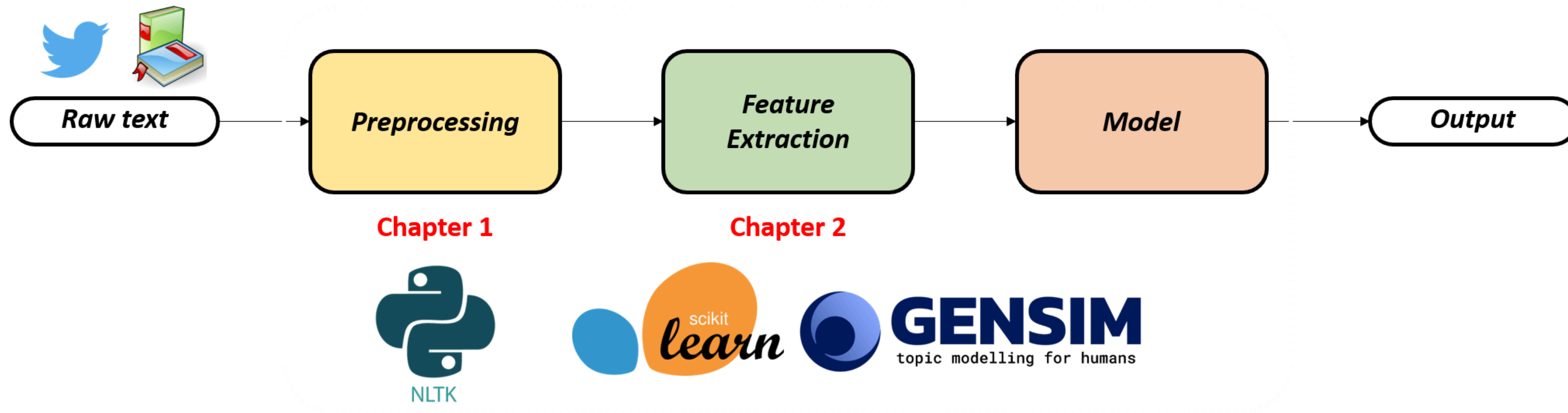


- **Texte brut** : d'un gazouillis à un paragraphe de livre
- **Prétraitement** : nettoyer le texte et retirer les éléments inutiles
- **Extraction de caractéristiques** : convertir le texte en nombres
- **Modèle** : analyser, prédire, classer, générer du nouveau contenu

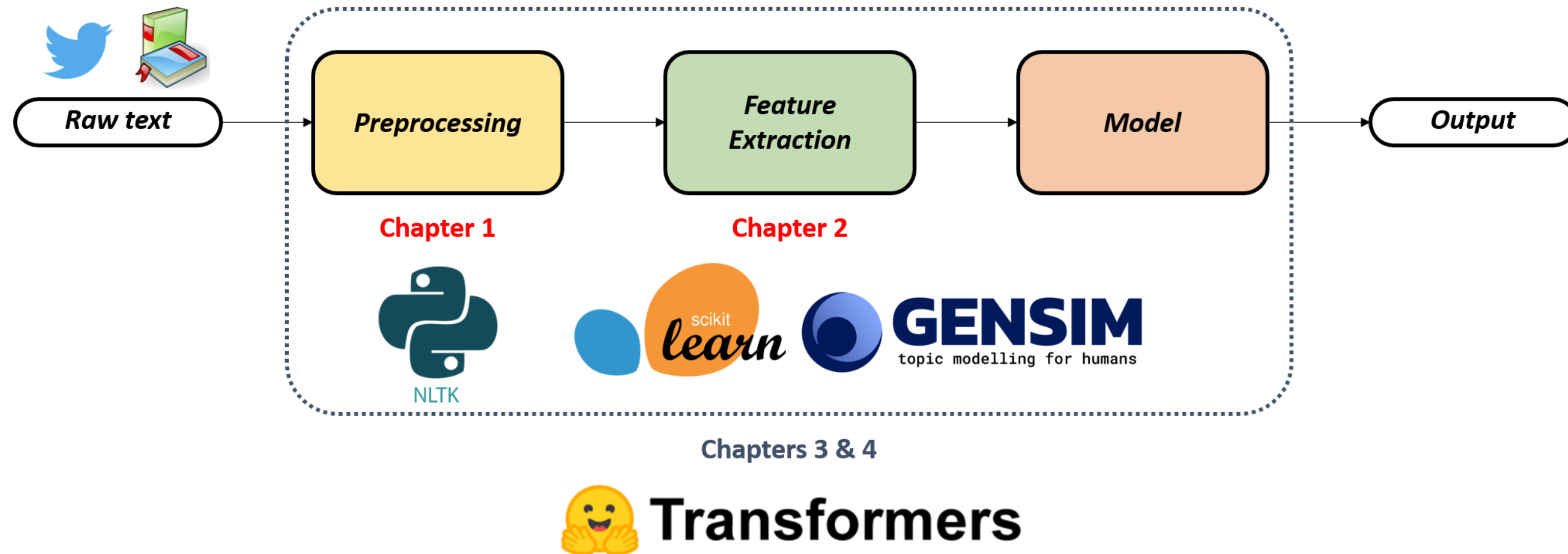
Plan du cours



Plan du cours



Plan du cours



Tokenisation

- Scinde le texte en jetons (petites unités gérables)



Tokenisation des phrases

- Texte → phrases
- Donne plus de clarté que d'analyser tout le texte

```
import nltk
nltk.download('punkt_tab')

text = "NLP is fun. Let's dive into it!"
sentences = nltk.sent_tokenize(text)
print(sentences)
```

```
["NLP is fun.", "Let's dive into it!"]
```



Tokenisation des mots

- Texte → mots et ponctuation
- Utile pour :
 - Repérer les termes clés
 - Compter la fréquence des mots

```
text = "Claim your free prize now!"  
words = nltk.word_tokenize(text)  
print(words)
```

```
['Claim', 'your', 'free', 'prize', 'now', '!']
```

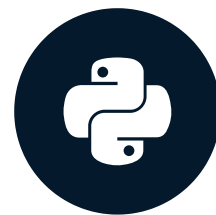


Passons à la pratique !

NATURAL LANGUAGE PROCESSING (NLP) EN PYTHON

Traitement des stop words et de la ponctuation

NATURAL LANGUAGE PROCESSING (NLP) EN PYTHON



Fouad Trad
Machine Learning Engineer

Stop words

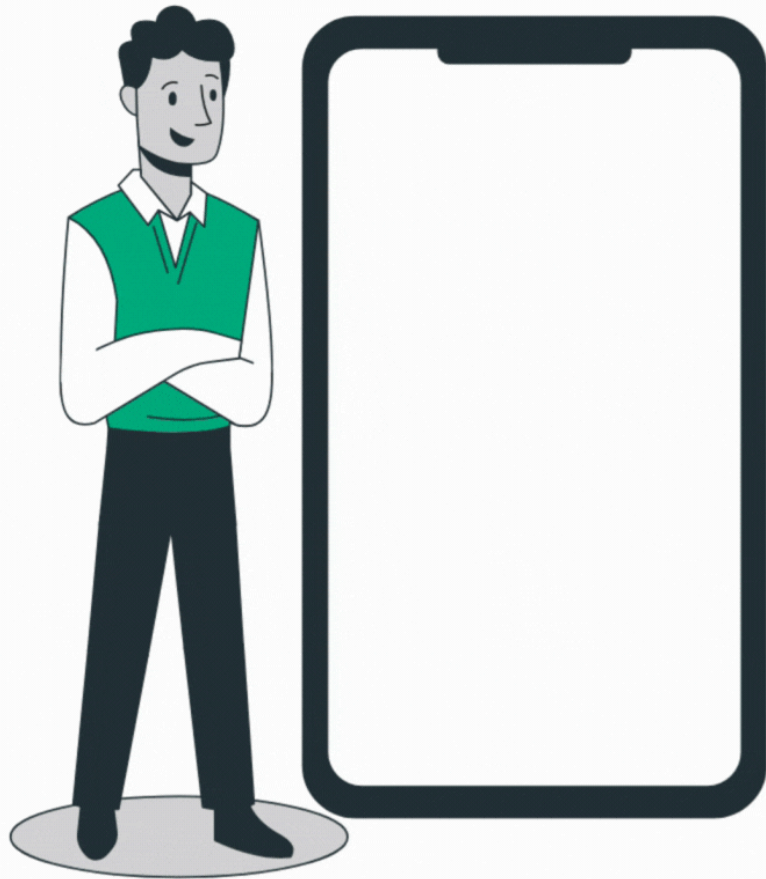
- Fréquents, mais apportent peu au contexte pour la machine
- Peu utiles dans bien des tâches de TAL
- Les retirer aide les modèles à se concentrer sur les mots importants

the as will
and at
for with be
by is in
was to an of a ...
that on

Retrait des stop words

Utile pour

Comprendre le sujet d'un texte

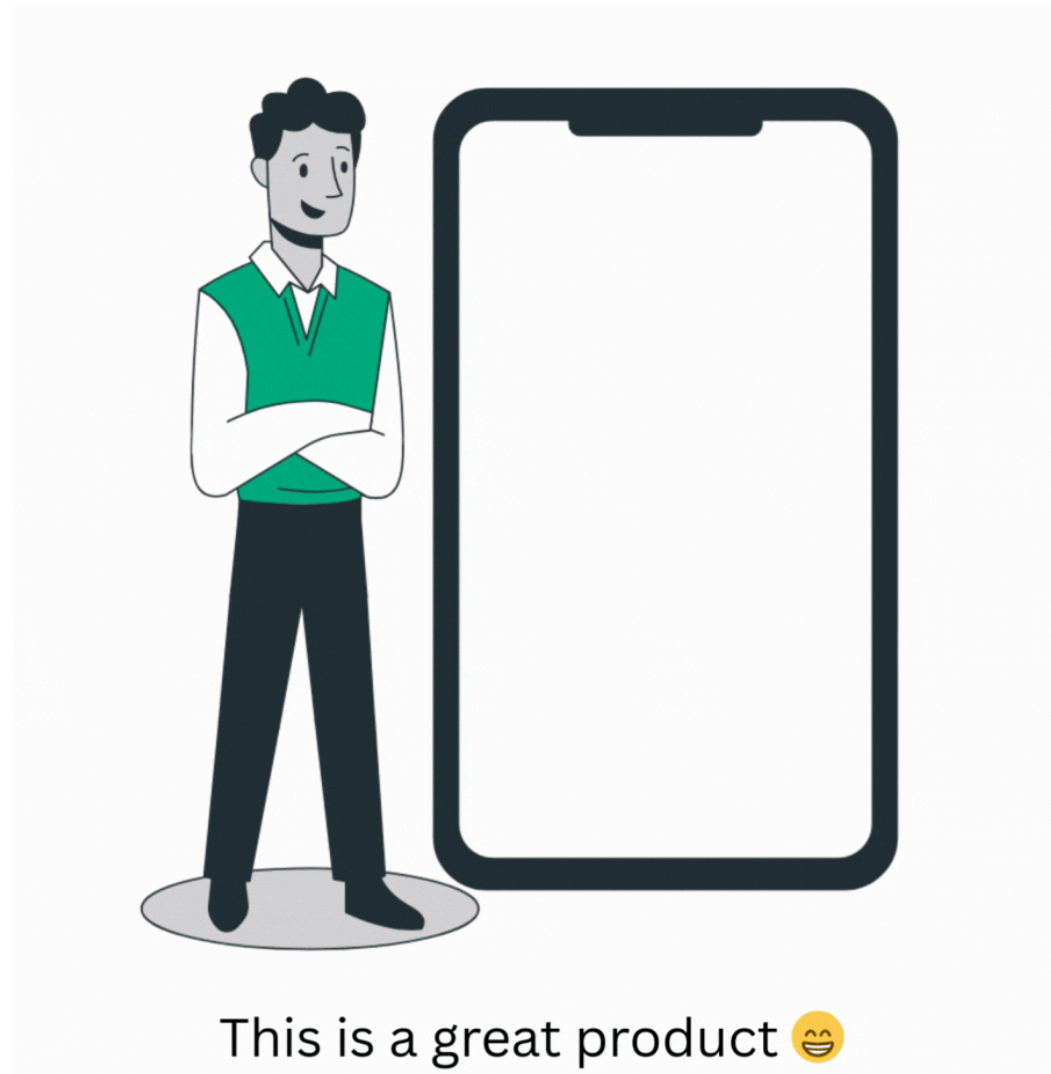


This is a great product 😊

Retrait des stop words

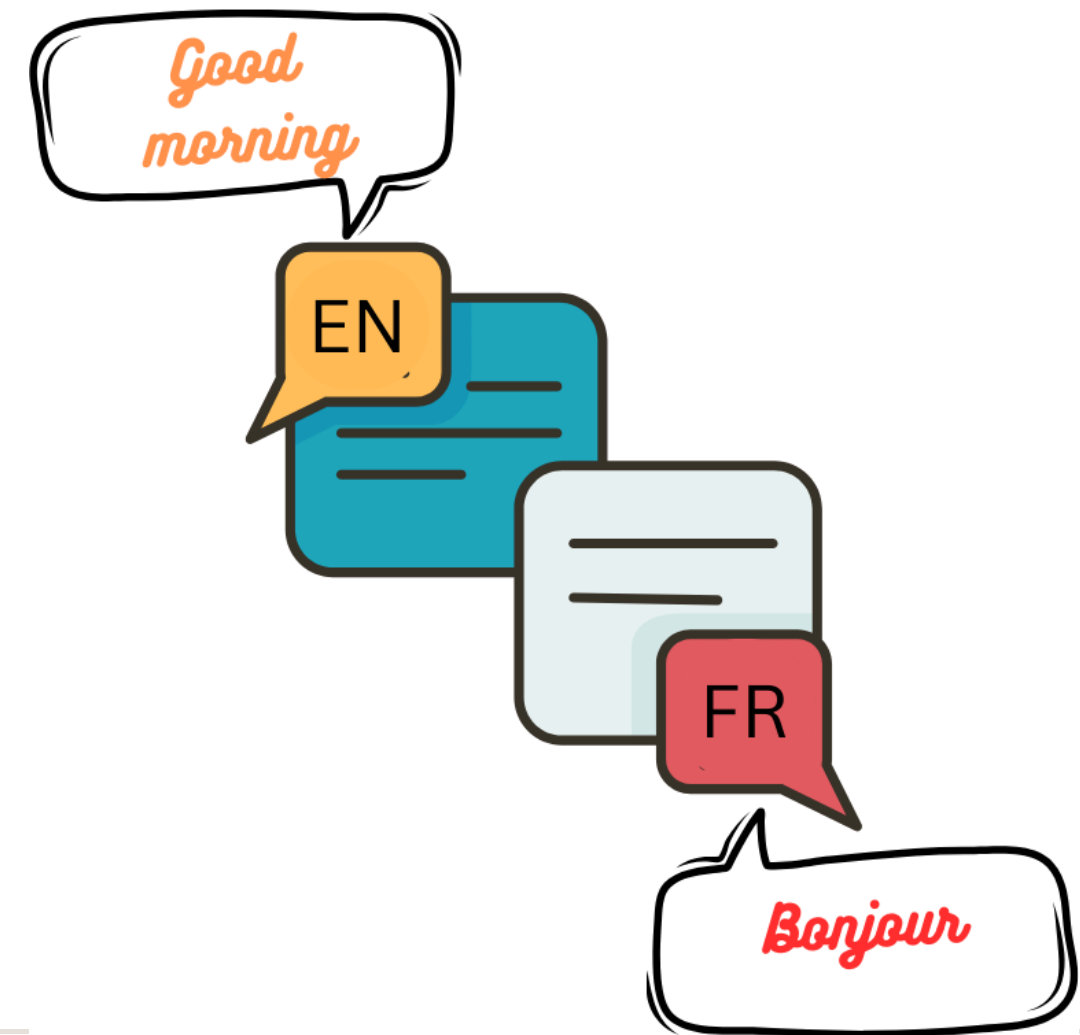
Utile pour

Comprendre le sujet d'un texte



Pas utile pour

Les tâches où chaque mot compte dans le texte



Accéder aux stop words

NLTK fournit une liste de stop words pour plusieurs langues

```
from nltk.corpus import stopwords
nltk.download('stopwords')

stop_words = stopwords.words('english')
print(stop_words[:10])
```

```
['a', 'about', 'above', 'after', 'again', 'against', 'ain', 'all', 'am', 'an']
```

Retirer les stop words

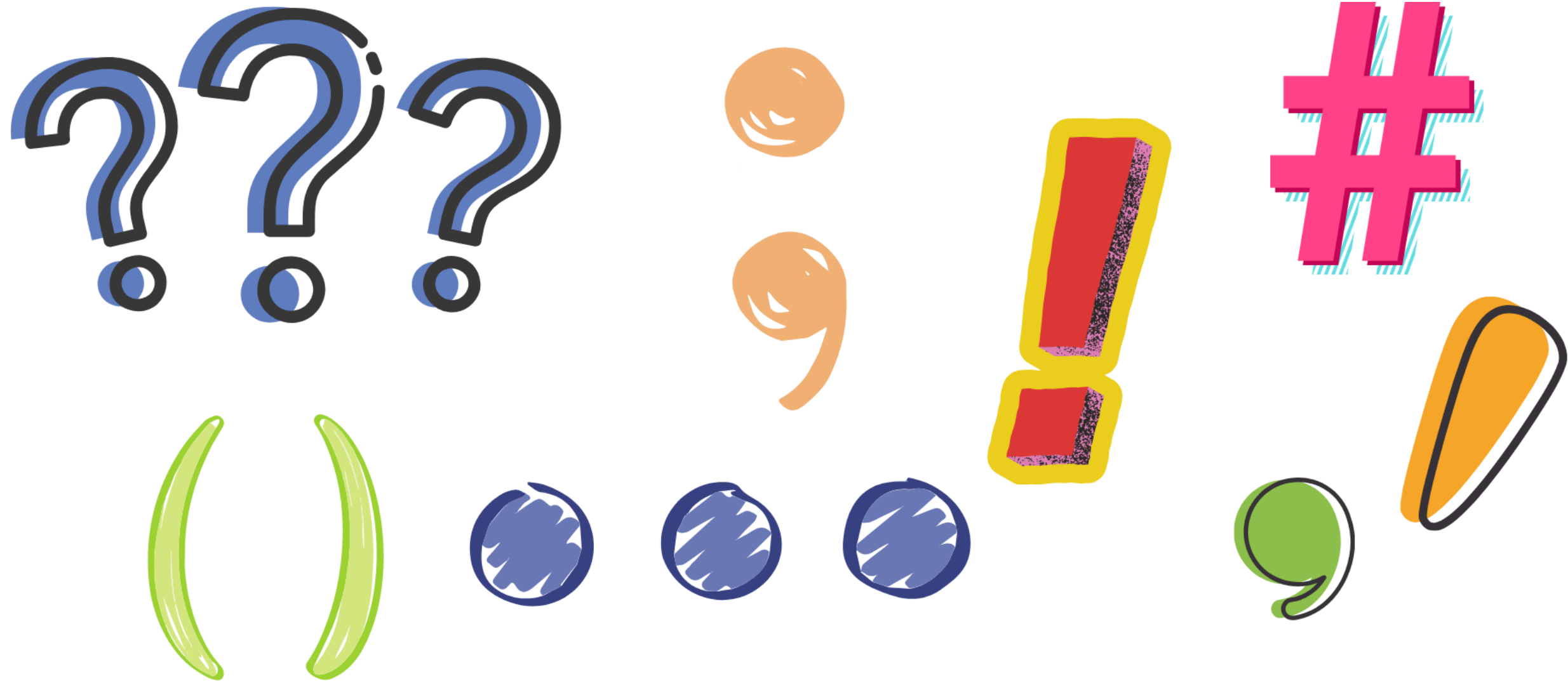
```
from nltk.tokenize import word_tokenize

text = "This is an example to demonstrate removing stop words."
tokens = word_tokenize(text)
# The .lower() method helps with case sensitivity
filtered_tokens = [word for word in tokens if word.lower() not in stop_words]
print(filtered_tokens)
```

```
['example', 'demonstrate', 'removing', 'stop', 'words', '.']
```

Punctuation

- Structure la langue pour les humains
- Souvent sans information utile en TAL



Retrait de la ponctuation

Utile pour

Trouver des mots fréquents ou importants
dans des documents



Retrait de la ponctuation

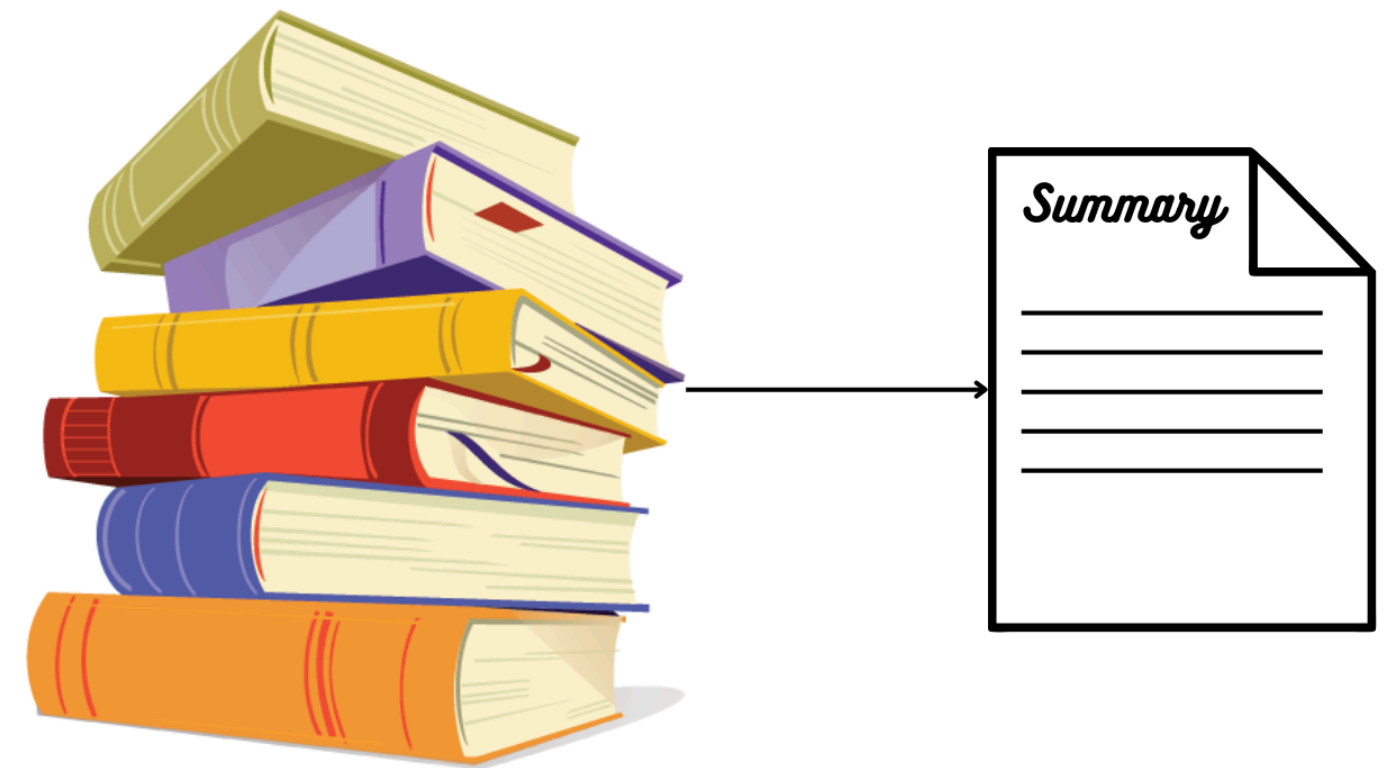
Utile pour

Trouver des mots fréquents ou importants dans des documents



Pas utile pour

Les tâches où il faut garder la structure des phrases pour la clarté



Accéder à la ponctuation et la retirer

```
import string
print(string.punctuation)
```

```
!"#$%&'()*+,-./:;<=>?@[\\]^_`{|}~
```

```
text = "This is an example to demonstrate removing stop words."
tokens = word_tokenize(text)
filtered_tokens = [word for word in tokens if word.lower() not in stop_words]

clean_tokens = [word for word in filtered_tokens if word not in string.punctuation]
print(clean_tokens)
```

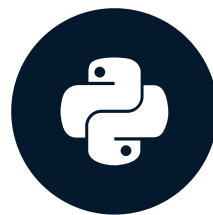
```
['example', 'demonstrate', 'removing', 'stop', 'words']
```

Passons à la pratique !

NATURAL LANGUAGE PROCESSING (NLP) EN PYTHON

Techniques de normalisation du texte

NATURAL LANGUAGE PROCESSING (NLP) EN PYTHON



Fouad Trad
Machine Learning Engineer

Normalisation du texte

- La normalisation met les mots dans un format standard
- Utile pour la classification, la recherche, l'analyse de sentiment
- Techniques à voir :
 - Mise en minuscules
 - Stemming
 - Lemmatisation



I enjoy watching movies
with great direction!



I enjoyed watching movies
that are well-directed.



Movies with excellent direction
are something I enjoy.

Mise en minuscules

- Sans normalisation, ces mots seront traités comme des jetons distincts
- Cela induit les modèles en erreur et gonfle le vocabulaire

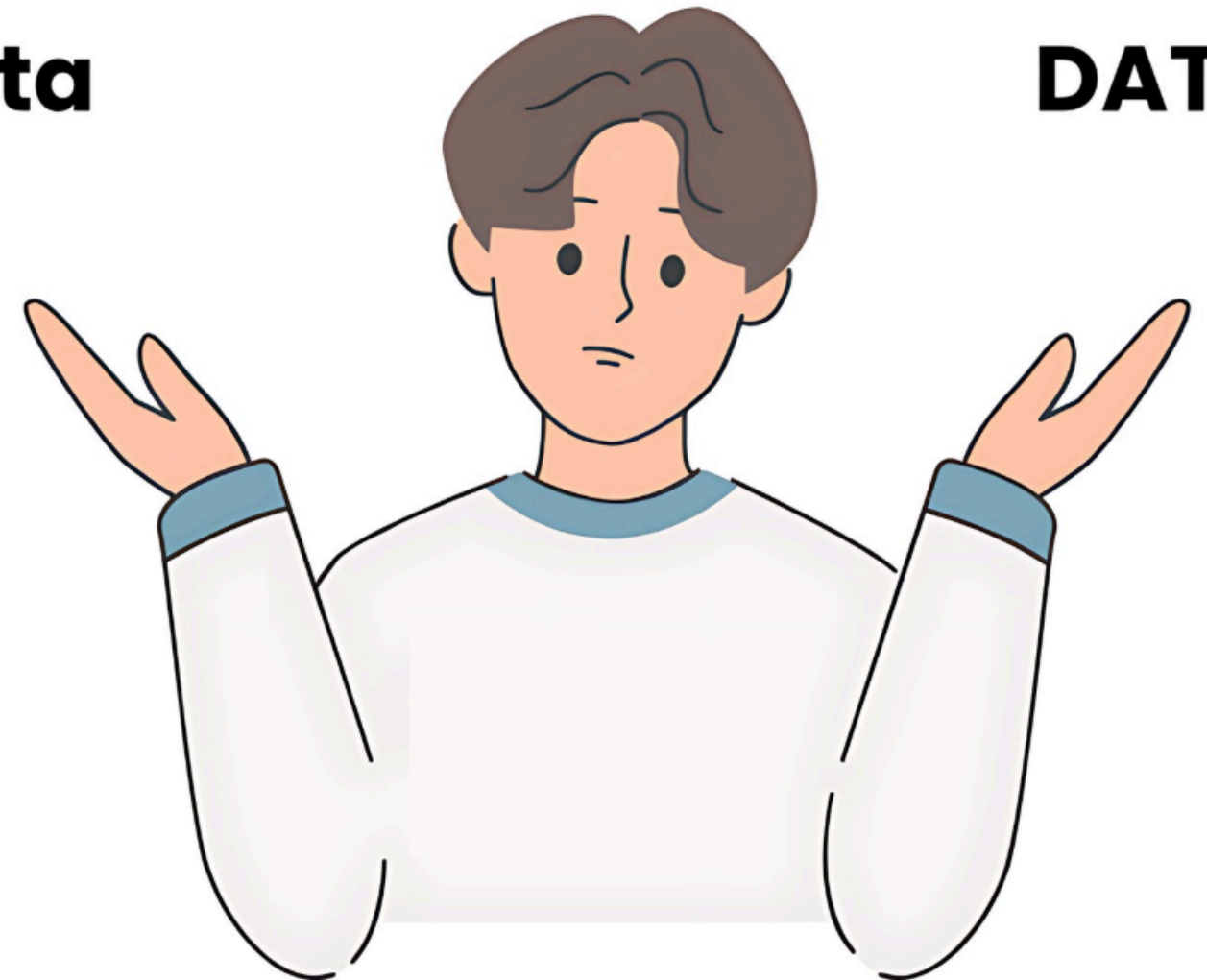
```
text = """Data scientists and data  
engineers need DATA"""  
lower_text = text.lower()  
print(lower_text)
```

```
data scientists and data  
engineers need data
```

data

Data

DATA



Applicabilité de la mise en minuscules

Utile pour

Tâches fondées sur l'analyse de mots-clés



Applicabilité de la mise en minuscules

Utile pour

Tâches fondées sur l'analyse de mots-clés



Peu utile pour

Tâches nécessitant de distinguer des mots dont le sens dépend de la casse



us



US

Stemming

- Réduit les mots à leur racine en coupant les suffixes
 - running → run
 - reading → read
- Utile pour regrouper des variantes proches d'un mot
- Peut produire des mots non valides
 - organization → organizat



Stemming en code

```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()

tokens = ['running', 'bats', 'organizations', 'reading']

stemmed = [stemmer.stem(word) for word in tokens]

print(stemmed)
```

```
['run', 'bat', 'organizat', 'read']
```

Lemmatisation

- Réduit les mots à leur forme de base
- Garantit un mot valide en sortie
 - organizations → organization
- Utile quand l'exactitude grammaticale est essentielle
- Plus lent que le stemming, mais plus précis

Lemmatization



Lemmatisation en code

```
from nltk.stem import WordNetLemmatizer
nltk.download('wordnet')

lemmatizer = WordNetLemmatizer()

tokens = ['running', 'bats', 'organizations', 'reading']
lemmatized = [lemmatizer.lemmatize(word) for word in tokens]

print(lemmatized)
```

```
['running', 'bat', 'organization', 'reading']
```

Stemming

- Plus rapide
- Produit des mots hors dictionnaire
- À privilégier si la vitesse est prioritaire



Lemmatisation

- Plus lente
- Plus exacte
- À privilégier si l'exactitude grammaticale est requise



Passons à la pratique !

NATURAL LANGUAGE PROCESSING (NLP) EN PYTHON

