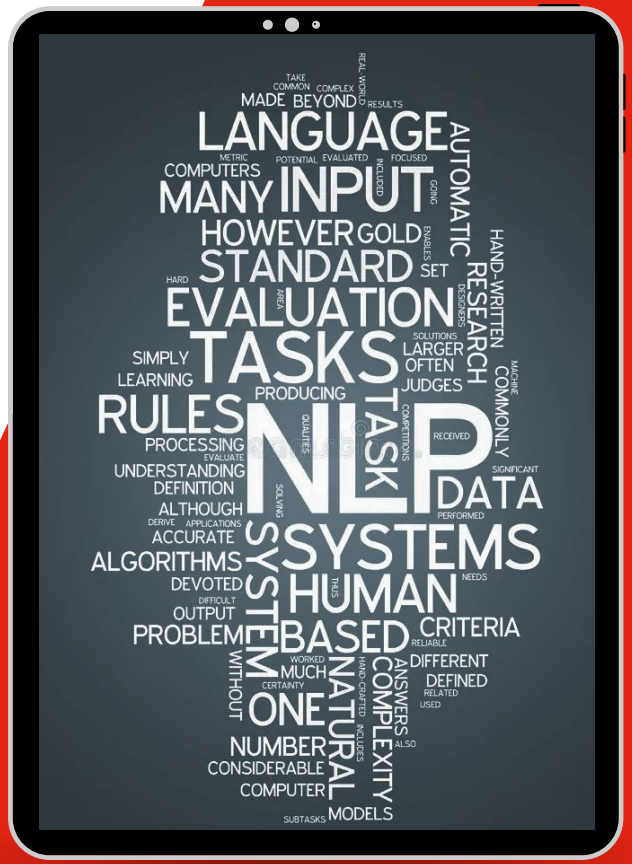
Text Preprocessing

Muhajir Akbar Hasibuan

Class Meeting 2

2023



Muhajir Akbar Hasibuan

Data Scientist

muhajirakbarhsb@gmail.com
+62 85297023234
Jakarta, Indonesia
linkedin.com/in/muhajirakbarhsb
github.com/muhajirakbarhsb
medium.com/@muhajirakbarhsb

Experience

Telkom Indonesia *Sept 2020 - present*

- Develop internal and external use cases
- Provide data understanding in making a model
- Provide Preparation and data engineering according to the use case implemented
- Provide data validation so that the analysis results are as expected
- Building modeling for the development or improvement of internal and external programs
- Provide descriptive and diagnostic insight into data processing
- Recommend and define new growth hacking strategy for digital marketing team

Project

- Pioneered a Robust Big Data Solution for MyIndihome, Revolutionizing Customer Experiences
- Orchestrated a High-Performing ML Team, Elevating myIndihome TV's Personalized Content Impact by 25%
- Envisioned and Executed a Cutting-Edge Big Data Solution for Elevated Customer Engagement on myIndihomeTV
- Fueled Business Insights via Dynamic Data Profiling, Performance Dashboards, and Insights for Langit Musik, RBT, and Upoint
- Engineered Innovative Big Data Solutions that Propelled Growth for PadiUmkm (E-Commerce)
- Powered Success for GameQoo through Strategic Big Data Solutions
- Architected and Established MLOps Framework, Elevating Telkom Indonesia's Digital Business Products
- Crafted Visionary Video Analytics Solutions for Telkom Indonesia's Revolutionary Digital Business IoT Product

MEMBER OF DATA SCIENTIST TASK FORCE | NOVEMBER 2021 - PRESENT
MEMBER OF AI TASK FORCE | NOVEMBER 2022- PRESENT

A Pool of data scientists and AI Engineer Expert in Telkom Indonesia. It was established to leverage data-driven culture for decision-making within the organization. (Applied Research, Standardization, Consultation)

Achievement

BEST Talent of the Year at Telkom Indonesia, Digital Business and Technology Division - Digital Technology and Platform -2022

Codex by Telkom Indonesia *Apr 2020- Sept 2020*

- Building Data Pipeline for Langit Musik Recommender System
- Business Analytic for Langit Musik
- Build Recommender System for Langit Musik

Universitas Syiah Kuala *2015-2020*

- Bachelor of Science, Statistics

Skills

Hard Skills

- Data Analytics
- Statistics
- Machine Learning
- Deep Learning
- MlOps
- Business Intelligence
- Data Engineering
- Cloud Computing
- Time Series & Demand Forecasting
- Natural Language Processing
- Fraud & Anomaly Detection
- Computer Vision

Tools Skills

- Python
- Pyspark
- Sql
- Apache Airflow
- GCP
- Docker
- MLflow
- Prometheus
- Evidently
- Grafana
- Redash, Metabase, Superset, Looker Studio
- Pytorch

Class Meeting 1: Introduction to NLP (1 session)
Introduction to NLP
Basics of text data and its characteristics
The importance of NLP in today's world
Overview of the course structure and objectives

Class Meeting 2: Text Preprocessing (1 session)
Understanding the text preprocessing pipeline
Tokenization, stemming, and lemmatization
Stop words removal
Hands-on exercises with Python for text preprocessing

Class Meeting 3: Text Representation (1 session)
Introduction to text representation techniques
Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF)
n-gram
Word embeddings (Word2Vec, GloVe)**just an introduction
Practical exercises on text representation

Class Meeting 4: Introduction to ML, DL Supervised and Unsupervised (1 session)
Introduction to deep learning (ML)
Introduction to deep learning (DL)
DL vs. traditional machine learning (ML)
Machine Learning and their applications
Neural networks and their applications
Basic ML DL concepts and terminology

Class Meeting 5: Sentiment Analysis (1 session)
What is sentiment analysis?
Data collection and labeling for sentiment analysis
Building a sentiment analysis model
Practical sentiment analysis examples

Class Meeting 6: Text Classification (1 session)
Introduction to text classification
Binary and multi-class classification
Building a text classification model
Real-world text classification examples

Class Meeting 7: Introduction to Topic Modeling(1 session)
Introduction to topic Modelingg
Topic Modeling Implementation
LDA(Latent dirichlet allocation)
Hands-on exercises with Topic Modeling

Class Meeting 8: Mid-term Exam

Class Meeting 9: Introduction to Word Embeddings (1 session)
Fundamentals of word embeddings
Word2Vec, GloVe, and FastText
Word embedding applications
Hands-on exercises with word embedding

Class Meeting 10: Attention Mechanisms (1 session)
Introduction to attention mechanisms
Self-attention and multi-head attention
Transformers architecture
Practical examples of attention in NLP

Class Meeting 11: Transformer Models (1 session)
In-depth study of the Transformer model
Pre-trained transformer models (BERT, GPT-2)
Fine-tuning transformers for NLP tasks
Transformer-based applications

Class Meeting 12: Advanced NLP Topics (1 session)
Advanced NLP topics such as BERT, XLNet, and RoBERTa
Transfer learning in NLP
Ethics in NLP
Recent developments and trends in NLP

Class Meeting 13: Advanced NLP Topics and Deployment Process in Industry (1 session)
Introduction to LLM
Introduction how industry utilize NLP to generate revenue
Introduction MLOps for NLP(bonuses from practitioners)

Class Meeting 14-15: Project Work (2 sessions)
Dedicated sessions for students to work on NLP projects with guidance and assistance.

Class Meeting 16: Project Presentations and Conclusion (1 session)
Students present their NLP projects
Recap of key takeaways from the course
Discuss further resources for NLP enthusiasts
Course conclusion and feedback

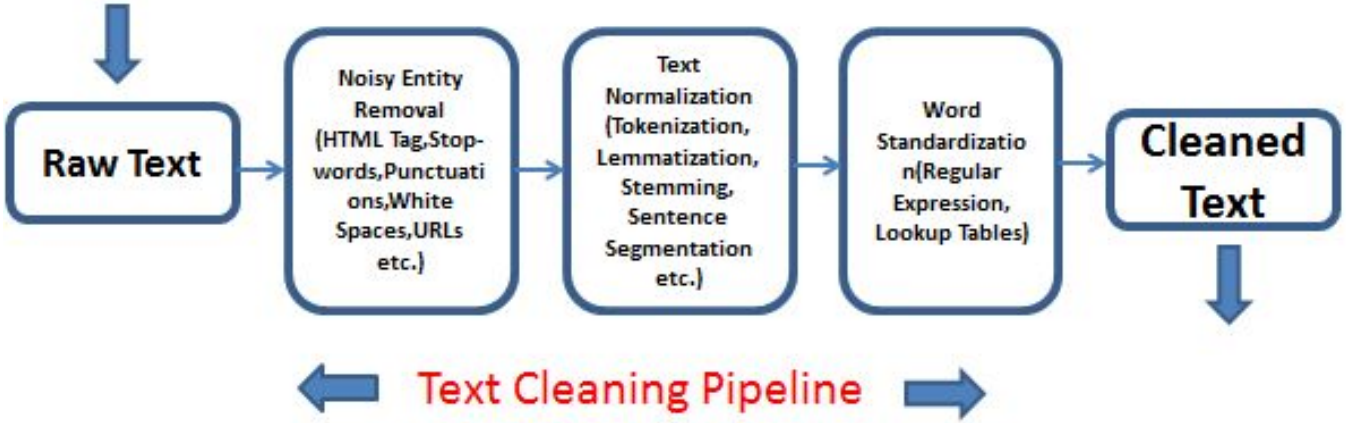
Agenda

- Introduction to Text Preprocessing
- Tokenization, Stemming, and Normalization
- Stop Words Removal
- Hands-on Python Exercises

Introduction to Text Processing

Text Processing

Text preprocessing is the essential initial step in natural language processing that involves **cleaning and structuring raw text data** to make it suitable for analysis and modeling.



NLTK

- NLTK is a Python library for working with human language data.
- It offers various text processing capabilities.
- Functions include tokenization, stemming, lemmatization, part-of-speech tagging, and more.
- NLTK is widely used in research and applications in the field of Natural Language Processing (NLP).

```
!pip install nltk
```

Sastrawi

- Sastrawi is a Python library designed specifically for Indonesian text preprocessing.
- It specializes in stemming for Bahasa Indonesia.
- Its main purpose is to remove suffixes from Indonesian words to obtain their root forms.
- Sastrawi is valuable for text mining and analysis in Bahasa Indonesia.

```
!pip install sastrawi
```

Case Folding

Case Folding

Case folding is a text normalization technique used in natural language processing (NLP) and text processing to standardize the text by converting all characters to a common case, **typically lowercase**. The goal of case folding is to ensure that text data is consistent and can be processed more effectively by eliminating variations in character case.

```
[9] kalimat = "Berikut ini adalah 5 negara dengan pendidikan terbaik di dunia adalah Korea Selatan, Jepang, Singapura, Hong Kong, dan Finlar
    lower_case = kalimat.lower()
    print(lower_case)
```

```
berikut ini adalah 5 negara dengan pendidikan terbaik di dunia adalah korea selatan, jepang, singapura, hong kong, dan finlandia.
```

Removing Numbers

Remove numbers if they are not relevant to what you are analyzing, such as house numbers, phone numbers, etc. Regular expressions (regex) can be used to remove numeric characters. Python has a module for performing tasks related to regex.

```
[14] import re # regular exoression modul
     kalimat = "Berikut ini adalah 5 negara dengan pendidikan terbaik di dunia adalah Korea Selatan, Jepang, Singapura, Hong Kong, dan Finlar
     hasil = re.sub(r"\d+",'', kalimat)
     print(hasil)

     Berikut ini adalah  negara dengan pendidikan terbaik di dunia adalah Korea Selatan, Jepang, Singapura, Hong Kong, dan Finlandia.
```

Removing Punctuation

Similar to numbers, punctuation marks in a sentence don't significantly impact text preprocessing. You can remove punctuation marks like **[!”#$%&’()*+,-./:;<=>?@[]^_`{|}~]** in Python as shown below:

```
[16] import string
     kalimat = "Ini &adalah [contoh] kalimat? {dengan} tanda. baca?!!"
     hasil = kalimat.translate(str.maketrans("","",string.punctuation))
     print(hasil)

     Ini adalah contoh kalimat dengan tanda baca
```

Removing whitespace(empty Characters)

To remove spaces at the beginning and end, you can use the strip() function in Python. See the code below:

```
[19] kalimat = " \t ini kalimat contoh\t "
     hasil = kalimat.strip()
     print(kalimat)
     print(hasil)
```

```
         ini kalimat contoh
ini kalimat contoh
```

Tokenization

Tokenization

Tokenization is the fundamental process of breaking down a block of text into its individual components or "tokens," which can be words, phrases, or even sentences. For example, consider the sentence:

Input: "The quick brown fox jumps over the lazy dog."

Tokenized Output: ["The", "quick", "brown", "fox", "jumps", "over", "the", "lazy", "dog", "."]

In this example, each word and the punctuation mark (period) is a token, and tokenization helps make the text data suitable for various natural language processing tasks.

Tokenization

```
[26] import nltk
     from nltk.tokenize import word_tokenize
     #nltk.download('punkt')


     kalimat = "rumah idaman adalah rumah yang bersih."
     kalimat = kalimat.translate(str.maketrans('','',string.punctuation)).lower()
     tokens = nltk.tokenize.word_tokenize(kalimat)
     print(tokens)

     ['rumah', 'idaman', 'adalah', 'rumah', 'yang', 'bersih']
```

We can also obtain information about the frequency of each token using the FreqDist() class available in the NLTK module.

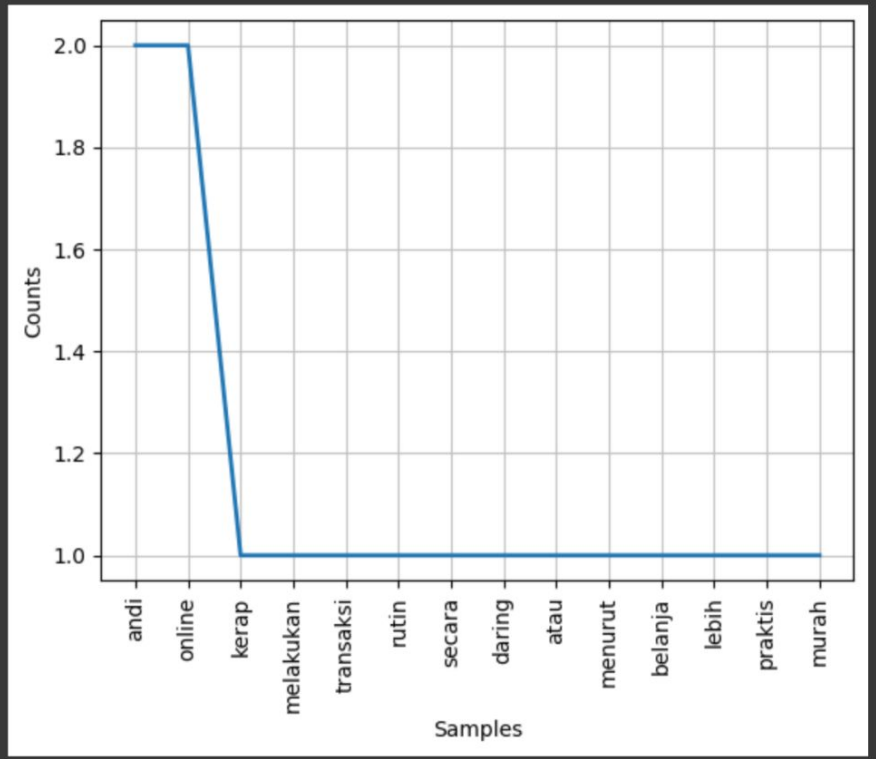
```
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
kalimat = "Andi kerap melakukan transaksi rutin secara daring atau online. Menurut Andi belanja online lebih praktis & murah."
kalimat = kalimat.translate(str.maketrans('','',string.punctuation)).lower()

tokens = nltk.tokenize.word_tokenize(kalimat)
kemunculan = nltk.FreqDist(tokens)
print(kemunculan.most_common())

[('andi', 2), ('online', 2), ('kerap', 1), ('melakukan', 1), ('transaksi', 1), ('rutin', 1), ('secara', 1), ('daring', 1), ('atau', 1), (
```

Tokenization

```
import matplotlib.pyplot as plt
kemunculan.plot(30,cumulative=False)
plt.show()
```



Tokenizing Sentence

The same principle can be applied to separate sentences within a paragraph. You can use the **sent_tokenize()** class in the NLTK module. I have added sentences to the example below:

```
[30] from nltk.tokenize import sent_tokenize
     kalimat = "Andi kerap melakukan transaksi rutin secara daring atau online. Menurut Andi belanja online lebih praktis & murah."

     tokens = nltk.tokenize.sent_tokenize(kalimat)
     print(tokens)

     ['Andi kerap melakukan transaksi rutin secara daring atau online.', 'Menurut Andi belanja online lebih praktis & murah.']
```

StopWords

StopWords

Filtering is the process of extracting important words from tokenized results using either a stoplist algorithm (discarding less important words) or a wordlist (retaining important words).

A stopword is a common word that typically appears in large quantities and is considered to have little meaning. Examples of stopwords in the Indonesian language include **"yang," "dan," "di," "dari,"** etc. The underlying purpose of using stopwords is to **remove low-information words from a text so that we can focus on the important ones instead**.

Stopwords using NLTK

```python
from nltk.tokenize import sent_tokenize, word_tokenize
nltk.download('stopwords')
from nltk.corpus import stopwords

kalimat = "Andi kerap melakukan transaksi rutin secara daring atau online. Menurut Andi belanja online lebih praktis & murah."
kalimat = kalimat.translate(str.maketrans('','',string.punctuation)).lower()

tokens = word_tokenize(kalimat)
listStopword =  set(stopwords.words('indonesian'))

removed = []
for t in tokens:
    if t not in listStopword:
        removed.append(t)

print(removed)
```

```
['andi', 'kerap', 'transaksi', 'rutin', 'daring', 'online', 'andi', 'belanja', 'online', 'praktis', 'murah']
```

Stopwords using Sastrawi

```
[33] from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
     from nltk.tokenize import word_tokenize
     factory = StopWordRemoverFactory()
     stopword = factory.create_stop_word_remover()
     kalimat = "Andi kerap melakukan transaksi rutin secara daring atau online. Menurut Andi belanja online lebih praktis & murah."
     kalimat = kalimat.translate(str.maketrans('','',string.punctuation)).lower()
     stop = stopword.remove(kalimat)
     tokens = nltk.tokenize.word_tokenize(stop)
     print(tokens)

     ['andi', 'kerap', 'melakukan', 'transaksi', 'rutin', 'daring', 'online', 'andi', 'belanja', 'online', 'lebih', 'praktis', 'murah']
```

Add your own stopword list

```python
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
from nltk.tokenize import word_tokenize
import string
import nltk

# Create a factory for the stopword remover
factory = StopWordRemoverFactory()
stopword = factory.create_stop_word_remover()

# Define your custom stopwords
custom_stopwords = ["andi"]

kalimat = "Andi kerap melakukan transaksi rutin secara daring atau online. Menurut Andi belanja online lebih praktis & murah."

# Remove punctuation and convert to lowercase
kalimat = kalimat.translate(str.maketrans('', '', string.punctuation)).lower()

# Apply Sastrawi stopword removal
stop = stopword.remove(kalimat)

# Tokenize the text
tokens = nltk.tokenize.word_tokenize(stop)

# Remove custom stopwords
tokens = [x for x in tokens if x not in custom_stopwords]

print(tokens)
```

```
['kerap', 'melakukan', 'transaksi', 'rutin', 'daring', 'online', 'belanja', 'online', 'lebih', 'praktis', 'murah']
```

Add your own stopword list

Adjusting the list of stopwords should be done **at the beginning** of each analysis project. It may not be a cumbersome task, but if not done, i**t can lead to misinterpretation** of the data.

Stemming

Stemming

Stemming is the process of removing word inflections to **reduce them to their base form**, although this base form may not necessarily be the same as the root word. For example, the words **"mendengarkan," "dengarkan," and "didengarkan"** will be transformed into the word **"dengar"**

The idea is that when you search for a document like **"cara membuka lemari"** you also want to see documents mentioning **"cara dibuka lemari"** or **"cara terbuka lemari,"** even though they may sound less precise. Of course, you want to **match all word variations** to bring up the most relevant documents.

Stemming english(using NLTK)

```
[38] from nltk.stem import PorterStemmer

     ps = PorterStemmer()

     kata = ["program", "programs", "programer", "programing", "programers"]

     for k in kata:
         print(k, " : ", ps.stem(k))
```

```
program  :  program
programs  :  program
programer  :  program
programing  :  program
programers  :  program
```

Stemming indonesia(using Sastrawi)

```python
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
factory = StemmerFactory()
stemmer = factory.create_stemmer()

kalimat = "Andi kerap melakukan transaksi rutin secara daring atau online. Menurut Andi belanja online lebih praktis & murah."
hasil = stemmer.stem(kalimat)
print(hasil)
```

```
andi kerap laku transaksi rutin cara daring atau online turut andi belanja online lebih praktis murah
```

Normalization

Alay or Slang Words

Normalization of slang words in Natural Language Processing (NLP) is an essential preprocessing step to ensure that text data is consistent and understandable. **Slang words often deviate from standard language rules, and normalizing them helps improve the quality of text analysis**.

alay,normal
matab,mantap
tdk,tidak
muantap,mantap
gk,tidak
bs,bisa
slalu,selalu
nyesel,nyesal
chenel,channel
isok,bisa
cuman,cuma
klo,kalau
gk,tidak
y,ya
bru,baru
mai,mau
nggak,tidak
lgi,lagi
da,ada
bangat,banget
mntap,mantap
kenap,kenapa
bgus,bagus
sdh,sudah
knpa,kenapa
mantab,mantap
gabisa,gakbisa
sma,sama
skr,sekarang
sj,saja
gw,saya
imail,email
channel,channel
apliaksi,aplikasi

Python Time

https://github.com/muhajirakbarhsb/NLP_class_2023/blob/main/Week_Class_2.ipynb