Natural Language Processing

NLP (Natural Language Processing) adalah teknologi yang membuat komputer bisa memahami, menganalisis, dan menghasilkan bahasa manusia. Fokusnya adalah pada teks dan ucapan, agar mesin bisa "mengerti" dan "berkomunikasi" seperti manusia. Untuk mengolah data dengan kolom text / object pastinya berbeda dengan mengolah data seperti biasanya, seperti:

- **Text Preprocessing**
  1. **Case Normalization**
     Digunakan untuk mengubah semua huruf ke bentuk huruf kecil untuk konsistensi teks dan menghindari duplikasi kata yang hanya berbeda huruf besar-kecil.
     Contoh: HARI SENIN -> hari senin
     Library (jika membutuhkan): re, nltk, spacy

  2. **Text Cleaning**
     Digunakan untuk menghapus karakter khusus, URL, atau tag HTML agar data hanya berupa teks yang relevan dan bersih.
     Contoh: kasus pem3unuh4n sumber: https://berita.com -> kasus pem3unuh4n sumber:
     Library (jika membutuhkan): re, nltk
     Contoh Implementasi: re.sub(r'http\S+|www\S+|https\S+', '', text)

  3. **Remove Punctuation and Numbers**
     Digunakan untuk menghapus tanda baca, angka, dan karakter khusus yang tidak relevan.
     Contoh: kasus pem3unuh4n sumber: -> kasus pemunuhn sumber
     Library (jika membutuhkan): re
     Contoh Implementasi: re.sub(r'[^\w\s]', '', text)

  4. **Tokenization**
     Digunakan untuk memecah teks menjadi kata atau kalimat untuk analisis lebih lanjut.
     Contoh: hari senin -> [‘hari’, ‘senin’]
     Library (jika membutuhkan): re, nltk, spacy

  5. **Stopword**
     Digunakan untuk menghapus kata-kata yang tidak terlalu penting / tidak memberikan informasi penting.
     Contoh: [‘di’, ‘hari’, ‘senin’] -> [‘hari’, ‘senin’]
     kata ‘di’ dihilangkan karena tidak memberikan informasi penting.
     Library (jika membutuhkan): re, nltk, spacy

Contoh Implementasi: nltk.corpus.stopwords.words('indonesian')

6. **Stemming dan Lemmatization**
   Stemming digunakan untuk memotong akhir kata menjadi bentuk dasar, sedangkan lemmatization digunakan untuk mengubah kata ke bentuk dasarnya (sesuai kamus bahasa).
   Contoh stemming: liburan -> libur
   Contoh lemmatization: berlarian -> lari
   Library (jika membutuhkan): nltk (untuk bahasa Indonesia masih belum tersedia), spacy (bisa untuk bahasa Indonesia), sastrawi (khusus bahasa Indonesia)

7. **Spelling Correction**
   Digunakan untuk memperbaiki ejaan yang salah.
   Contoh: pemunuhn -> pembunuhan
   Library (jika membutuhkan): textblob, nltk, symspellpy

- ## Pembobotan Kata

Setelah melakukan text preprocessing, langkah selanjutnya adalah pembobotan kata (feature extraction). Pembobotan ini bertujuan untuk mengubah teks menjadi representasi numerik yang bisa digunakan dalam model machine learning.

Jenis-Jenis Pembobotan Kata:

1. **Bag of Words (BoW)**
   BoW mengubah teks menjadi representasi numerik dengan menghitung frekuensi kemunculan kata di dokumen. Setiap kata diwakili oleh angka yang menunjukkan seberapa sering kata tersebut muncul, tanpa mempertimbangkan konteks atau urutan kata. Kelemahannya, BoW tidak menangkap hubungan yang mirip antara kata-kata. CountVectorizer adalah salah satu teknik yang menggunakan BoW

2. **Term Frequency-Inverse Document Frequency (TF-IDF)**
   TF-IDF mengkombinasikan Term Frequency (seberapa sering kata muncul di dokumen) dan Inverse Document Frequency (seberapa jarang kata muncul di semua dokumen). TF-IDF memberikan bobot lebih tinggi pada kata-kata yang penting dalam dokumen, tetapi jarang muncul di dokumen lainnya, sehingga mengurangi dampak kata-kata umum.

3. **Word Embeddings**
   Word embeddings adalah representasi kata dalam bentuk vektor yang mencerminkan makna semantik kata berdasarkan konteks. Vektor kata diciptakan oleh model neural network, seperti Word2Vec atau GloVe, yang

menangkap hubungan semantik antar kata, sehingga mampu menangkap arti kata berdasarkan konteks.

Langkah - Langkah Membuat TF-IDF:

1. **Menghitung Term Frequency (TF)**
   TF digunakan untuk menghitung seberapa sering suatu kata muncul dalam dokumen (atau baris dalam dataframe) dibandingkan kata lainnya. Ini membantu menentukan kata-kata penting yang mewakili isi dokumen.
   Contoh: Jika dalam sebuah baris dataframe tentang "mobil", kata "mobil" sering muncul, maka kata itu dianggap penting untuk baris tersebut.

$$tf_{ij} = \frac{f_d(i)}{\max\limits_{j \in d} f_d(j)}$$

   Keterangan:
   - **TF_{i}:** Seberapa sering kata i muncul di dokumen j dibandingkan kata lain.
   - **$f_a(i)$:** Jumlah kemunculan kata i di dokumen a.
   - **max $f_a(j)$:** Jumlah kemunculan kata yang paling sering muncul di dokumen a.

2. **Menghitung Inverse Document Frequency (IDF)**
   IDF digunakan untuk mengukur seberapa unik atau jarang sebuah kata muncul di kumpulan dokumen. Kata yang sering muncul di banyak dokumen (misalnya "dan" atau "adalah") dianggap kurang penting, sedangkan kata yang jarang muncul lebih dihargai karena lebih spesifik dan relevan terhadap dokumen tertentu.

$$idf(t, D) = \log\left(\frac{N}{df(t)+1}\right)$$

   Keterangan:
   - **N:** Jumlah total dokumen dalam kumpulan data.
   - **df(t):** Jumlah dokumen yang mengandung kata t.
   - **+1:** Untuk mencegah pembagian dengan nol jika suatu kata tidak muncul di dokumen mana pun.

- **log:** Digunakan untuk mereduksi skala perbedaan nilai IDF, sehingga nilai ekstrem tidak terlalu besar.

3. **TF-IDF**
   Setelah menghitung TF dan IDF, langkah berikutnya adalah menggabungkannya menjadi TF-IDF.

$$\text{TF-IDF}(i,j) = \text{TF}(i,j) \times \text{IDF}(i)$$

Keterangan:

- TF(i, j): Seberapa sering kata i muncul dalam dokumen j, dibandingkan kata lain di dokumen tersebut.
- IDF(i): Mengukur seberapa unik atau jarang kata i muncul di seluruh kumpulan dokumen. Kata yang sering muncul di semua dokumen akan memiliki nilai IDF rendah, sedangkan kata yang jarang muncul akan memiliki nilai IDF tinggi.

reference:

- **text preprocessing:**
  1. https://youtu.be/GHThIlwFeVs?si=qf8SwWaMfPlCSe_6
  2. https://youtu.be/FL_FLfzVrk8?si=GLWlZdulH_tu2neA
  3. https://youtu.be/x2cLmkxvAe4?si=MKJSSGV8Cmo01uza

- **tf-idf:**
  1. https://youtu.be/zLMEnNbdh4Q?si=NDYcy2u2ns2yhyly
  2. https://yunusmuhammad007.medium.com/tf-idf-term-frequency-inverse-document-frequency-representasi-vector-data-text-2a4eff56cda
  3. https://youtu.be/B3UZ8DxHocQ?si=GxHrWmAq9dxDSuXK
  4. https://youtu.be/UmRv-BuVAKk?si=LPHFAOJrVW295AsV