Perhitungan Manual untuk TF-IDF

Contoh dokumen :

Dokumen 1: "saya suka makan nasi"

* saya: TF = 1/4 = 0.25, IDF = log(3/2) = 0.176, TF-IDF = 0.25 \* 0.176 = 0.044
* suka: TF = 1/4 = 0.25, IDF = log(3/2) = 0.176, TF-IDF = 0.25 \* 0.176 = 0.044
* makan: TF = 1/4 = 0.25, IDF = log(3/1) = 0.477, TF-IDF = 0.25 \* 0.477 = 0.119
* nasi: TF = 1/4 = 0.25, IDF = log(3/3) = 0, TF-IDF = 0.25 \* 0 = 0

Dokumen 2: "saya tidak suka nasi"

* saya: TF = 1/4 = 0.25, IDF = log(3/2) = 0.176, TF-IDF = 0.25 \* 0.176 = 0.044
* tidak: TF = 1/4 = 0.25, IDF = log(3/1) = 0.477, TF-IDF = 0.25 \* 0.477 = 0.119
* suka: TF = 1/4 = 0.25, IDF = log(3/2) = 0.176, TF-IDF = 0.25 \* 0.176 = 0.044
* nasi: TF = 1/4 = 0.25, IDF = log(3/3) = 0, TF-IDF = 0.25 \* 0 = 0

Dokumen 3: "nasi adalah makanan pokok"

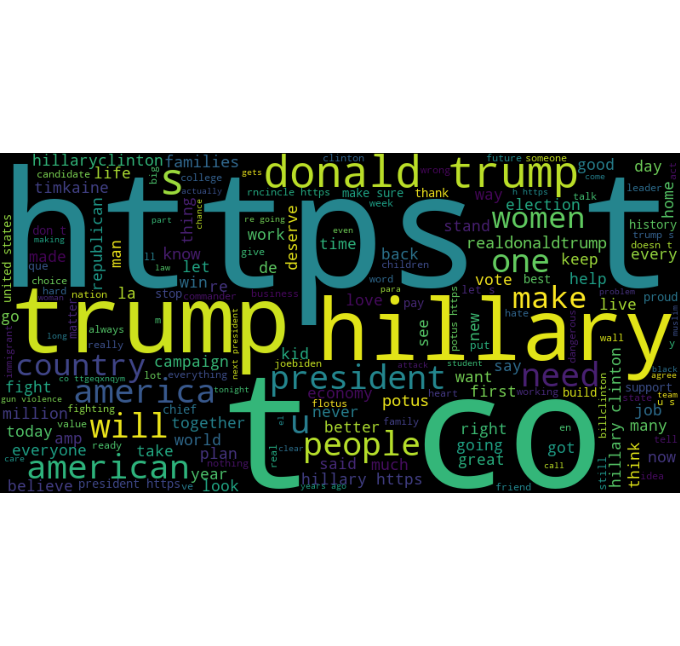
* nasi: TF = 1/4 = 0.25, IDF = log(3/3) = 0, TF-IDF = 0.25 \* 0 = 0
* adalah: TF = 1/4 = 0.25, IDF = log(3/1) = 0.477, TF-IDF = 0.25 \* 0.477 = 0.119
* makanan: TF = 1/4 = 0.25, IDF = log(3/1) = 0.477, TF-IDF = 0.25 \* 0.477 = 0.119
* pokok: TF = 1/4 = 0.25, IDF = log(3/1) = 0.477, TF-IDF = 0.25 \* 0.477 = 0.119

Preprocessing Text & Bag of Words

Preprocessing Text menggunakan Bahasa pemograman Py sebagai berikut:

|  |
| --- |
| # nge-ekstrak kata yang ada @  def extract\_words(df, c):      words = []      for t in df[c].tolist():          t = [x for x in t.split() if x.startswith('@')]          words += t      print(words[:10])  extract\_words(realDonaldTrump, 'text')  extract\_words(hillaryClinton, 'text')     # nge-ekstrak kata yang ada #  def extract\_words\_(df, c):      words = []      for t in df[c].tolist():          t = [x for x in t.split() if x.startswith('#')]          words += t      print(words[:10])  extract\_words(realDonaldTrump, 'text')  extract\_words(hillaryClinton, 'text')  # nge-ekstrak kata yang ada # (—)  def extract\_words\_(df, c):      words = []      for t in df[c].tolist():          t = [x for x in t.split() if x.startswith('—')]          words += t        print(words[:10])  extract\_words(realDonaldTrump, 'text')  extract\_words(hillaryClinton, 'text')  # ''' buang all tags (@, #, -) '''  def remove\_tags(t):      text = " ".join([x for x in t.split(" ") if not x.startswith("@")])      text = " ".join([x for x in text.split(" ") if not x.startswith("#")])      text = " ".join([x for x in text.split(" ") if not x.startswith("—")])      return text  num\_hillary\_tweets = len(hillaryClinton)  num\_trump\_tweets = len(realDonaldTrump)  print(f"Jumlah tweet dari Hillary Clinton: {num\_hillary\_tweets}")  print(f"Jumlah tweet dari Donald Trump: {num\_trump\_tweets}") |

Bag word yang didapat seperti berikut :



Similiarity menggunakan Bahasa pemograman Py:

|  |
| --- |
| # Fungsi untuk menghitung kata yang paling sering muncul  def most\_common\_words(df, column, num\_words=10):      text = " ".join(df[column].tolist())      words = [word.lower() for word in text.split() if word.lower() not in STOPWORDS and word not in pnc]      word\_counts = Counter(words)      return word\_counts.most\_common(num\_words) |