**Biodata mahasiswa**

NIM : 2112500422

Nama : Dhoni Khariri

Nomor Absen : 6

**Nomor 1.**

Lakukan crawling data twitter dengan kata kunci penting yang sesuai dengan passion Anda. Crawling data twitter bisa dilakukan dengan menggunakan link Google Colab Helmi Satria update 20 Mei 2024.

<https://colab.research.google.com/drive/11cyL2GVKIZyY1-O4RNrQnsz6XtNikmNV?usp=sharing>

Ada dianggap sudah menyelesaikan nomor ini setelah menyimpan data hasil crawling di dalam github Anda. **Berikan link github nya**.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Nomor 2.**

Buat file baru di Google Colabs atau Jupyter Notebook dengan cara mengunggah / load file dataset ke Google Colabs, gunakan **data kunci yang anda pilih.**

Berikan perintah untuk (1) load library pandas, (2) unduh dataset dengan wget, dan (3) membaca dataset dan menampilkannya.

|  |  |
| --- | --- |
| **Perintah** | import pandas as pd  !wget -O jihyo.csv  df = pd.read\_csv("jihyo.csv", delimiter=',', encoding='latin1')  df.head(150) |
| **Hasil** |  |

**Nomor 3**. Berapa jumlah baris datanya? Jalankan perintah berikut ini dan tampilkan hasilnya!

|  |  |
| --- | --- |
| **Perintah** | df.count() |
| **Hasil** |  |

**Nomor 4**. Tampilkan apakah terdapat data yang kosong (NULL). Jalankan perintah sebagai berikut dan tampilkan hasilnya! Apakah ada data yang kosong?

|  |  |
| --- | --- |
| **Perintah** | df.isna().sum() |
| **Hasil** |  |

**Nomor 5**. Periksa apakah terdapat data ‘fulltext’ yang duplikat / sama? Jalankan perintah sebagai berikut dan tampilkan hasilnya! Ada berapa yang duplikat?

|  |  |
| --- | --- |
| **Perintah** | df['full\_text'].duplicated().sum() |
| **Hasil** |  |

**Nomor 6**. Jika pada langkah 3 ternyata ditemukan data yang duplikat, maka kita perlu membuang duplikasi data. Jalankan perintah berikut ini, lalu cek kembali apakah datanya sudah tidak ada duplikat? Simpan datanya ke variabel ‘df2’. Berapa baris isi dari data df2 ?

|  |  |
| --- | --- |
| **Perintah** | df2 = df.drop\_duplicates(subset=["full\_text"])  df2 |
| **Hasil** |  |

**Nomor 7**. Tampilkan data yang panjang ‘fulltext’ kurang dari 4 karakter. Hapus data tersebut! Selanjutnya simpan hasilnya ke variabel ‘df3’. Berapa jumlah baris datanya sekarang?

|  |  |
| --- | --- |
| **Perintah** | df[df['full\_text'].str.len()<4]  df3 = df[df['full\_text'].str.len() >= 4]  df3 |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** |  |
| **Hasil** |  |

**Nomor 8**. Pada dataframe df3, ubah seluruh teks ‘fulltext’ menjadi huruf kecil semua (lowercase), simpan hasilnya pada kolom baru dengan nama ‘lowercase’. Jalankan perintah berikut ini dan tampilkan datanya!

|  |  |
| --- | --- |
| **Perintah** | df3['casefolding'] = df3['full\_text'].str.lower();  df3 |
| **Hasil** |  |

**Nomor 9**. Lakukan pembersihan teks pada kolom ‘casefolding’ dan simpan ke kolom baru ‘cleanedtext’. Pembersihan dilakukan dengan langkah: (1) menghilangkan url, (2) menghilangkan @, (3) menghilangkan tanda baca, (4) menghilangkan karakter pindah baris, tab, dan backspace, (5) menghilangkan spasi berlebih, dan (6) menghilangkan angka. Silakan buat dan jalankan fungsi sbb

|  |  |
| --- | --- |
| **Perintah** | import re  def cleaning\_text(text):      # menghilangkan url      url\_pattern = re.compile(r'https?://\S+|www\.\S+')      text =  url\_pattern.sub(r'', text)      # menghilangkah @, spt pada alamat email      text = re.sub(r'@[\w]\*', ' ', text)      # menghilangkan tanda baca      punctuations = '''!()-[]{};:'"\,<>./?@#$%^&\*\_~'''      for x in text.lower():          if x in punctuations:              text = text.replace(x, " ")      # menghilangkan newline (\n), tab (\t) dan backspace      text = text.replace('\\t'," ") .replace('\\n'," ").replace('\\u'," ").replace('\\',"")      # menghilangkan angka      text = re.sub(r"\d+", "", text)      # menghilangkan spasi berlebih      text = text.strip()      return text  df3['cleanedtext'] = df3['casefolding'].apply(cleaning\_text)  df3.head(150) |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** |  |
| **Hasil** |  |

**Nomor 10**. Definisikan stopword, buat fungsi untuk menerapkan penghapusan stopword, lalu terapkan pada data yang sudah dibersihkan di langkah sebelumnya. Simpan ke kolom baru ‘stopwordremoved’

|  |  |
| --- | --- |
| **Perintah** |  |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** |  |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** | import nltk  from nltk.corpus import stopwords  nltk.download('stopwords')  nltk.download('punkt')  import requests  sastrawi\_stopword = "https://raw.githubusercontent.com/onlyphantom/elangdev/master/elang/word2vec/utils/stopwords-list/sastrawi-stopwords.txt"  #bentuk stopword list  stopwords\_l = stopwords.words('indonesian')  response = requests.get(sastrawi\_stopword)  stopwords\_l += response.text.split('\n')  custom\_st = '''  yg yang dgn ane smpai bgt gua gwa si tu ama utk udh btw  ntar lol ttg emg aj aja tll sy sih kalo nya trsa mnrt nih  '''  # bikin dictionary untuk memastikan stopword bersifat unik  st\_words = set(stopwords\_l)  custom\_stopword = set(custom\_st.split())  # gabungkan dengan stopword custom  stop\_words = st\_words | custom\_stopword  #cetak sampel stopword  print(f'Stopwords: {list(stop\_words)[:5]}')  # remove stopwords function  from nltk import word\_tokenize, sent\_tokenize  def remove\_stopword(text, stop\_words=stop\_words):      word\_tokens = word\_tokenize(text)      filtered\_sentence = [w for w in word\_tokens if not w in stop\_words]      return ' '.join(filtered\_sentence)  df3['stopwordremoved'] = df3['cleanedtext'].apply(remove\_stopword)  df3.head(150) |
| **Hasil** |  |

**Nomor 11**. Install library Sastrawi. Lalu definisikan fungsi untuk melakukan stemming menggunakan library sastrawi. Terapkan fungsi stemming pada kolom text langkah sebelumnya dan simpan di kolom baru (‘stemmingtext’)

|  |  |
| --- | --- |
| **Perintah** |  |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** | !pip install PySastrawi |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** | # stemming  from Sastrawi.Stemmer.StemmerFactory import StemmerFactory  def stemming(text):      factory = StemmerFactory()      stemmer = factory.create\_stemmer()      return stemmer.stem(text)  df3['stemmingtext'] = df3['stopwordremoved'].apply(stemming)  df3.head(150) |
| **Hasil** |  |

**Nomor 12**. Definisikan fungsi tokenize untuk menghasilkan token. Lalu, terapkan fungsi tersebut untuk menghasilkan token dan simpan di kolom baru pada dataframe (‘tokenize’)

|  |  |
| --- | --- |
| **Perintah** |  |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** | # tokenization  def tokenize(text):      return word\_tokenize(text)  df3['tokenize'] = df3['stemmingtext'].apply(tokenize)  df3.head(150) |
| **Hasil** |  |

**Nomor 13**. Buat wordcloud

|  |  |
| --- | --- |
| **Perintah** | import matplotlib.pyplot as plt  from wordcloud import WordCloud  # bentuk wordcloud dari kolom 'stemmingtext'  data = df3['stemmingtext']  all\_text = ' '.join(word for word in data)  wordcloud = WordCloud(colormap='Blues', width=1000, height=800, mode='RGBA', background\_color='white').generate(all\_text)  plt.figure(figsize=(20,10), dpi=80)  plt.imshow(wordcloud, interpolation='bilinear')  plt.axis("off")  plt.margins(x=0, y=0)  plt.show() |
| **Hasil** |  |

**Nomor 14**. Simpan data yang sudah dibersihkan ke file yang **sesuai dengan kata kunci Anda**

|  |  |
| --- | --- |
| **Perintah** | f3.to\_csv('jihyo-cleaned.csv', index=False) |
| **Hasil** |  |

**Nomor 15. Pilihlah metode yang ingin anda buat. Apakah menggunakan metode Naïve Bayes atau Logistic Regression atau NLP Classification**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lakukan tahapan yang ada untuk penyelesaiannya. Tuliskan perintah yang Anda gunakan. Buat perintah setiap tahapannya.**

|  |  |
| --- | --- |
| **Perintah** | import pandas as pd  from sklearn.feature\_extraction.text import TfidfVectorizer  from sklearn.cluster import KMeans  # Baca dataset dari file CSV  file\_path = 'jihyo-cleaned.csv'  df = pd.read\_csv(file\_path)  # Pastikan dataset memiliki kolom 'text'  if 'full\_text' not in df.columns:      raise ValueError("Dataset harus memiliki kolom 'full\_text'")  # Ambil dokumen dari kolom 'text'  documents = df['full\_text'].tolist()  # Inisialisasi TfidfVectorizer dengan stop words bahasa Inggris  vectorizer = TfidfVectorizer(stop\_words='english')  X = vectorizer.fit\_transform(documents)  # Tentukan jumlah cluster  true\_k = 2  # Inisialisasi dan fit model KMeans  model = KMeans(n\_clusters=true\_k, init='k-means++', max\_iter=100, n\_init=1)  model.fit(X)  print("Top terms per cluster:")  order\_centroids = model.cluster\_centers\_.argsort()[:, ::-1]  terms = vectorizer.get\_feature\_names\_out()  for i in range(true\_k):      print("Cluster %d:" % i)      for ind in order\_centroids[i, :10]:          print(' %s' % terms[ind])      print()  print("\n")  print("Prediction")  # Contoh prediksi dokumen baru  new\_documents = ["Saya suka nonton badminton.", "Raisa adalah artis keren dari Indonesia"]  for doc in new\_documents:      Y = vectorizer.transform([doc])      prediction = model.predict(Y)      print(f"Dokumen: {doc} => Cluster {prediction[0]}") |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** | order\_centroids |
| **Hasil** |  |

|  |  |
| --- | --- |
| **Perintah** | model.labels\_ |
| **Hasil** |  |

**~ Selamat Mengerjakan ~**

|  |  |  |
| --- | --- | --- |
| Validasi Ketua Program Studi / Dosen Koordinator | Tanggal | Jam |
|  |  |  |
| Acuan Soal | (1)SAP (2)Pustaka (3)Pertemuan Kuliah (4)ISO | |