LAMPIRAN

Lampiran 1 - Daftar Riwayat Hidup

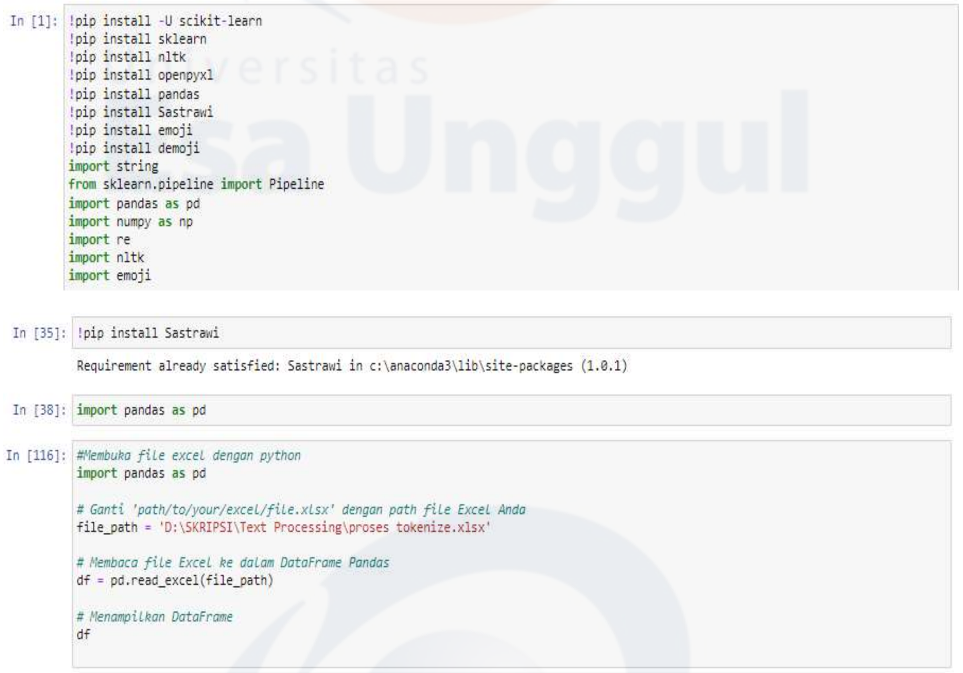
Data Pribadi

Nama	:	Muhamad Akbar Ramadhan
TTL	:	Tangerang, 02 Desember 2000
Jenis Kelamin	:	Laki-Laki
Agama	:	Islam
Kewarganegaraan	:	Indonesia
Alamat	:	Kp. Pos Sentul, Ds. Sentul Jaya, Kec. Balaraja, Kab. Tangerang, Banten
Nomor Telepon/HP	:	081318562529
E-mail	:	rama.akbar0101@gmail.com

Riwayat Pendidikan

Periode (Tahun)	Sekolah/Institusi	Jurusan	Jenjang Pendidikan
2007-2013	SDN Sentul Jaya 1	-	SD
2013-2016	SMPN 3 Balaraja	-	SMP
2016-2019	SMA Mandiri Balaraja	MIPA	SMA

Lampiran 2 – Kode Preprocessing

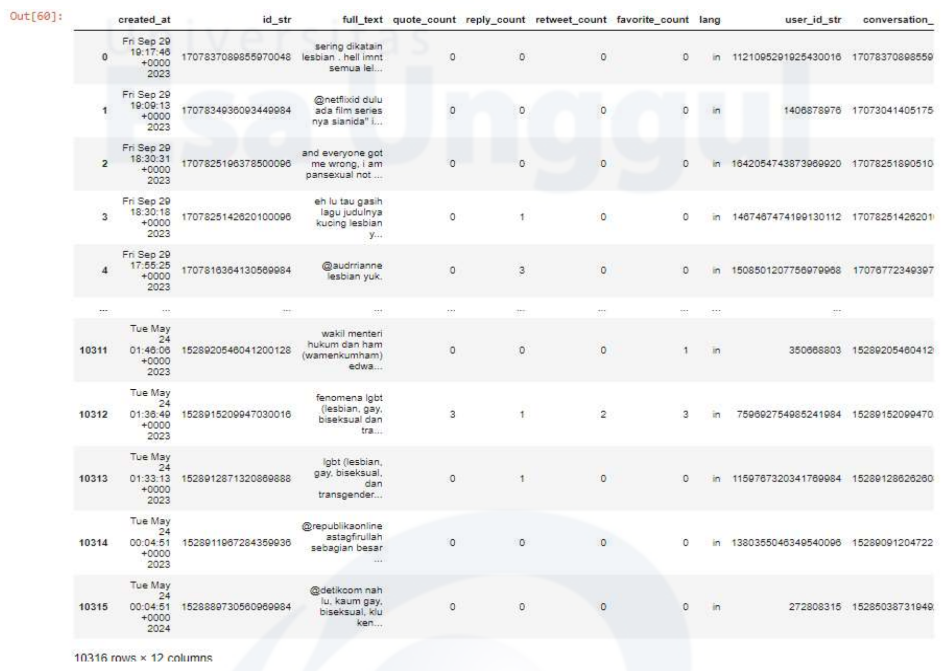


```
In [1]: !pip install -U scikit-learn
        !pip install sklearn
        !pip install nltk
        !pip install openpyxl
        !pip install pandas
        !pip install Sastrawi
        !pip install emoji
        !pip install demoji
        import string
        from sklearn.pipeline import Pipeline
        import pandas as pd
        import numpy as np
        import re
        import nltk
        import emoji
```

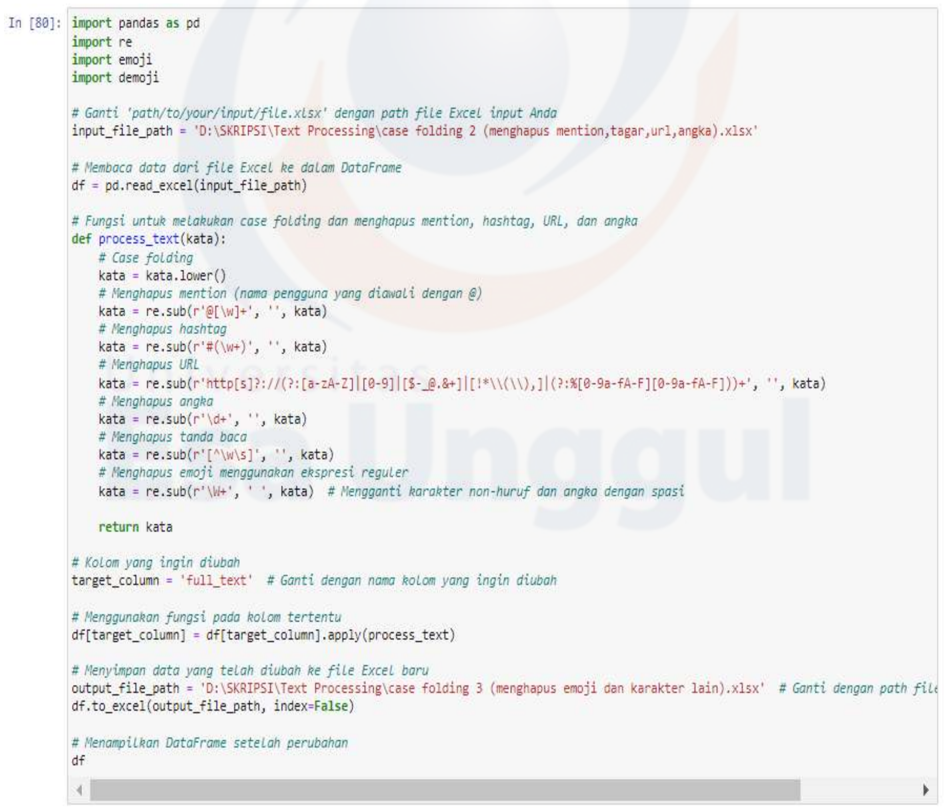
```
In [35]: !pip install Sastrawi

         Requirement already satisfied: Sastrawi in c:\anaconda3\lib\site-packages (1.0.1)
```

```
In [38]: import pandas as pd
```

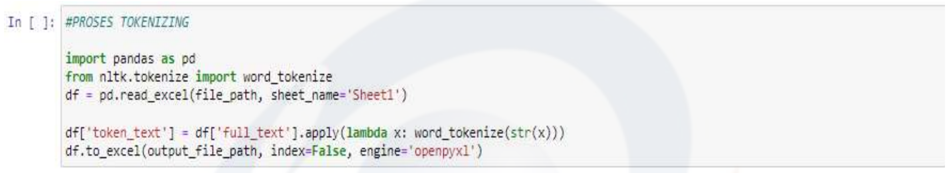
```
In [116]: #Membuka file excel dengan python
          import pandas as pd

          # Ganti 'path/to/your/excel/file.xlsx' dengan path file Excel Anda
          file_path = 'D:\SKRIPSI\Text Processing\proses tokenize.xlsx'

          # Membaca file Excel ke dalam DataFrame Pandas
          df = pd.read_excel(file_path)

          # Menampilkan DataFrame
          df
```



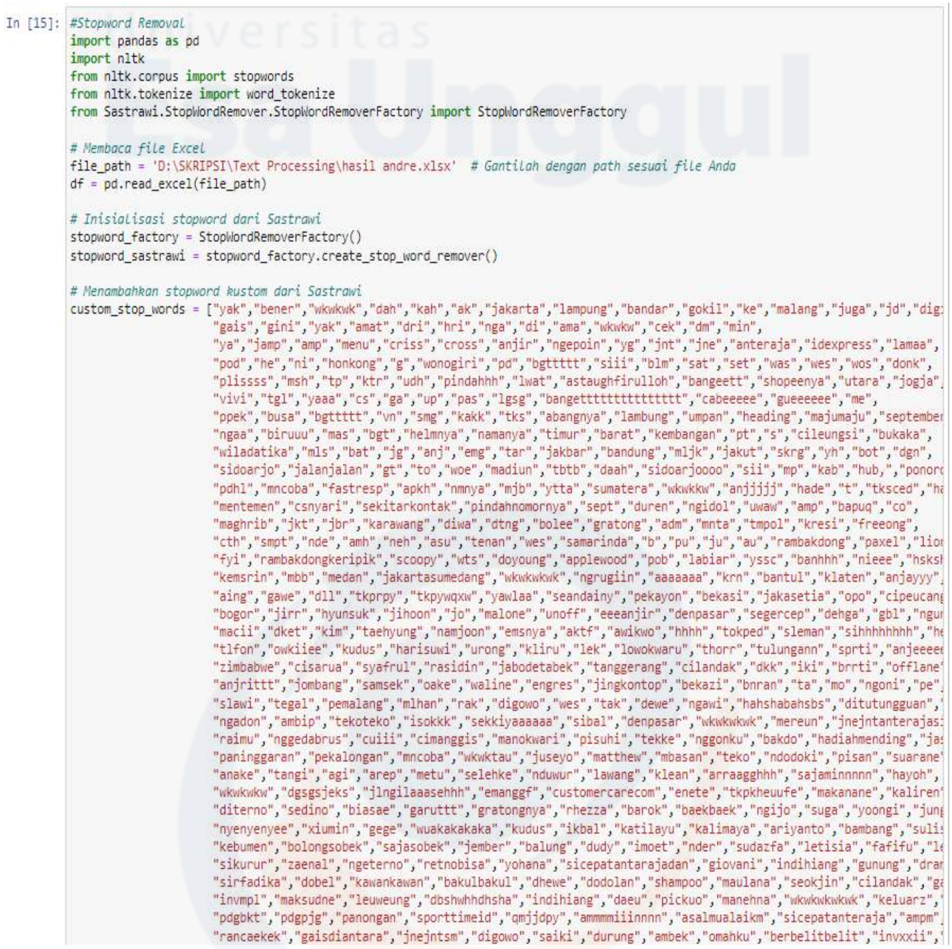
```
Out[60]:
          created_at               id_str          full_text  quote_count  reply_count  retweet_count  favorite_count  lang          user_id_str  conversation_
    0     Fri Sep 29 19:17:46 +0000 2023  1707837089855970048  sering dikatain lesbian . hell imnt semua lel...  0  0  0  0  in  1121095291925430016  17078370898559
    1     Fri Sep 29 19:09:13 +0000 2023  1707834936093449984  @netflixid dulu ada film series nya sianida" i...  0  0  0  0  in           1406878976  17073041405175
    2     Fri Sep 29 18:30:31 +0000 2023  1707825196378500096  and everyone got me wrong, i am pansexual not ...  0  0  0  0  in  1642054743873969920  17078251890510
    3     Fri Sep 29 18:30:18 +0000 2023  1707825142620100096  eh lu tau gasih lagu judulnya kucing lesbian y...  0  1  0  0  in  1467467474199130112  17078251426201
    4     Fri Sep 29 17:55:25 +0000 2023  1707816364130569984  @audrrianne lesbian yuk.  0  3  0  0  in  1508501207756979968  17076772349397
  ...                ...                  ...                ...          ...          ...            ...             ...   ...                  ...
10311     Tue May 24 01:46:06 +0000 2023  1528920546041200128  wakil menteri hukum dan ham (wamenkumham) edwa...  0  0  0  1  in            350668803  15289205460412
10312     Tue May 24 01:36:49 +0000 2023  1528915209947030016  fenomena lgbt (lesbian, gay, biseksual dan tra...  3  1  2  3  in   759692754985241984  15289152099470
10313     Tue May 24 01:33:13 +0000 2023  1528912871320869888  lgbt (lesbian, gay, biseksual, dan transgender...  0  1  0  0  in  1159767320341769984  15289128626260
10314     Tue May 24 00:04:51 +0000 2023  1528911967284359936  @republikaonline astagfirullah sebagian besar ...  0  0  0  0  in  1380355046349540096  15289091204722
10315     Tue May 24 00:04:51 +0000 2024  1528889730560969984  @detikcom nah lu, kaum gay, biseksual, klu ken...  0  0  0  0  in            272808315  15285038731949

10316 rows × 12 columns
```



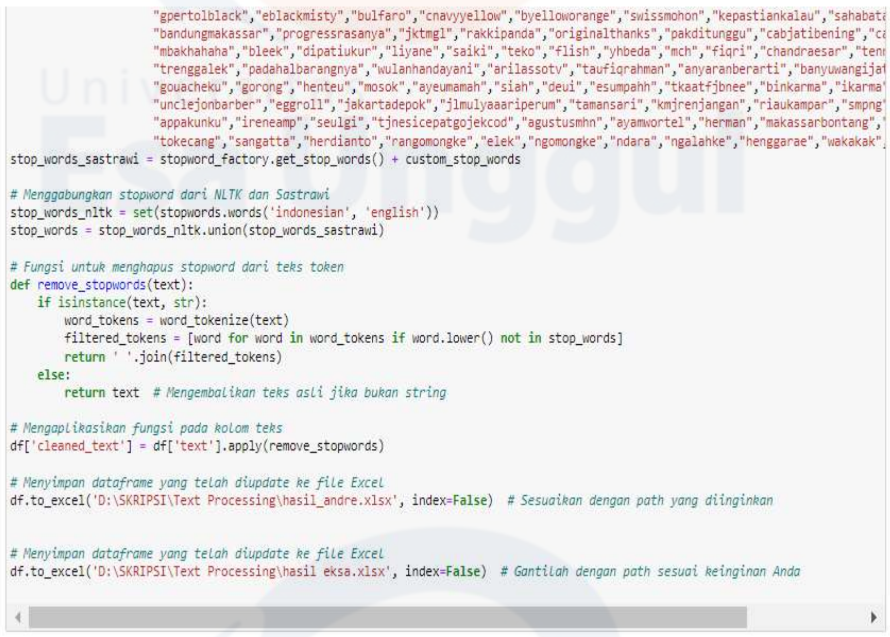
```
In [80]: import pandas as pd
         import re
         import emoji
         import demoji

         # Ganti 'path/to/your/input/file.xlsx' dengan path file Excel input Anda
         input_file_path = 'D:\SKRIPSI\Text Processing\case folding 2 (menghapus mention,tagar,url,angka).xlsx'

         # Membaca data dari file Excel ke dalam DataFrame
         df = pd.read_excel(input_file_path)

         # Fungsi untuk melakukan case folding dan menghapus mention, hashtag, URL, dan angka
         def process_text(kata):
             # Case folding
             kata = kata.lower()
             # Menghapus mention (nama pengguna yang diawali dengan @)
             kata = re.sub(r'@[\w]+', '', kata)
             # Menghapus hashtag
             kata = re.sub(r'#(\w+)', '', kata)
             # Menghapus URL
             kata = re.sub(r'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[!*\\(\\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+', '', kata)
             # Menghapus angka
             kata = re.sub(r'\d+', '', kata)
             # Menghapus tanda baca
             kata = re.sub(r'[^\w\s]', '', kata)
             # Menghapus emoji menggunakan ekspresi reguler
             kata = re.sub(r'\W+', ' ', kata)  # Mengganti karakter non-huruf dan angka dengan spasi

             return kata

         # Kolom yang ingin diubah
         target_column = 'full_text'  # Ganti dengan nama kolom yang ingin diubah

         # Menggunakan fungsi pada kolom tertentu
         df[target_column] = df[target_column].apply(process_text)

         # Menyimpan data yang telah diubah ke file Excel baru
         output_file_path = 'D:\SKRIPSI\Text Processing\case folding 3 (menghapus emoji dan karakter lain).xlsx'  # Ganti dengan path file

         df.to_excel(output_file_path, index=False)

         # Menampilkan DataFrame setelah perubahan
         df
```



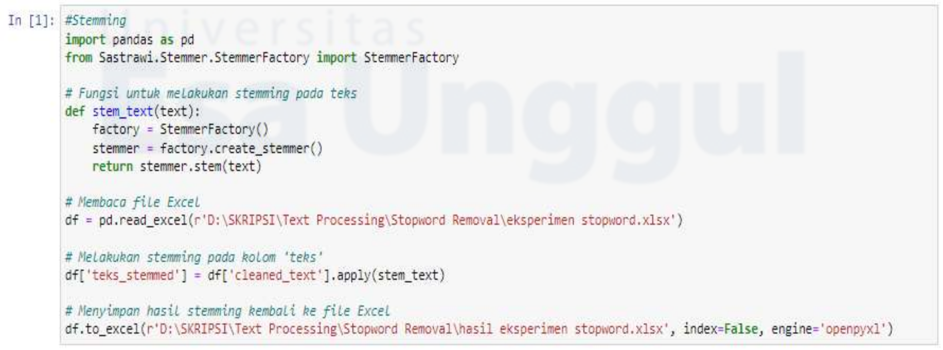
```
In [ ]: #PROSES TOKENIZING

        import pandas as pd
        from nltk.tokenize import word_tokenize
        df = pd.read_excel(file_path, sheet_name='Sheet1')

        df['token_text'] = df['full_text'].apply(lambda x: word_tokenize(str(x)))
        df.to_excel(output_file_path, index=False, engine='openpyxl')
```



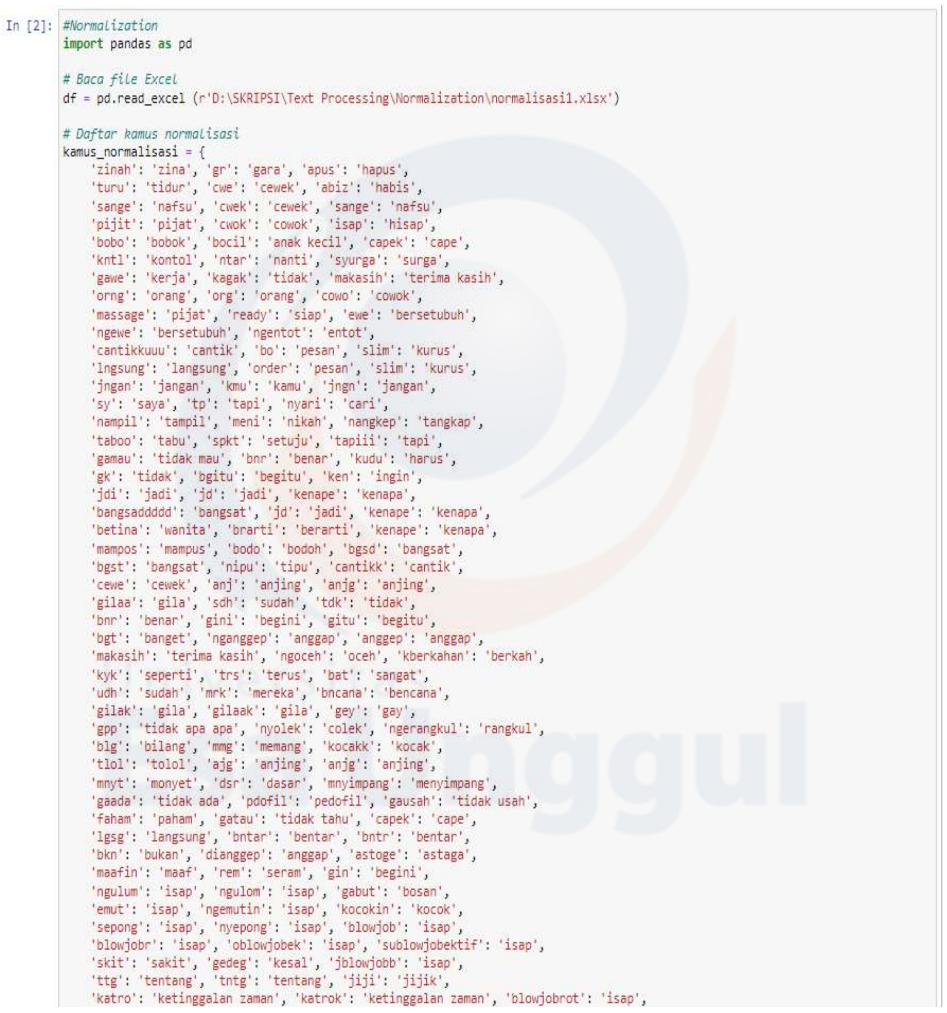
```
In [15]: #Stopword Removal
         import pandas as pd
         import nltk
         from nltk.corpus import stopwords
         from nltk.tokenize import word_tokenize
         from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory

         # Membaca file Excel
         file_path = 'D:\SKRIPSI\Text Processing\hasil andre.xlsx'  # Gantilah dengan path sesuai file Anda
         df = pd.read_excel(file_path)

         # Inisialisasi stopword dari Sastrawi
         stopword_factory = StopWordRemoverFactory()
         stopword_sastrawi = stopword_factory.create_stop_word_remover()

         # Menambahkan stopword kustom dari Sastrawi
         custom_stop_words = ["yak","bener","wkwkwk","dah","kah","ak","jakarta","lampung","bandar","gokil","ke","malang","juga","jd","dig
                              "gais","gini","yak","amat","dri","hri","nga","di","ama","wkwkw","cek","dm","min",
                              "ya","jamp","amp","menu","criss","cross","anjir","ngepoin","yg","jnt","jne","anteraja","idexpress","lamaa",
                              "pod","he","ni","honkong","g","wonogiri","pd","bgttttt","siii","blm","sat","set","was","wes","wos","donk",
                              "plissss","msh","tp","ktr","udh","pindahhh","lwat","astaughfirulloh","bangeett","shopeenya","utara","jogja",
                              "vivi","tgl","yaaa","cs","ga","up","pas","lgsg","bangettttttttttttttt","cabeeeee","gueeeeee","me",
                              "ppek","busa","bgttttt","vn","smg","kakk","tks","abangnya","lambung","umpan","heading","majumaju","september
                              "ngaa","biruuu","mas","bgt","helmnya","namanya","timur","barat","kembangan","pt","s","cileungsi","bukaka",
                              "wiladatika","mls","bat","jg","anj","emg","tar","jakbar","bandung","mljk","jakut","skrg","yh","bot","dgn",
                              "sidoarjo","jalanjalan","gt","to","woe","madiun","tbtb","daah","sidoarjoooo","sii","mp","kab","hub,","ponoro
                              "pdhl","mncoba","fastresp","apkh","nmnya","mjb","ytta","sumatera","wkwkkw","anjjjjj","hade","t","tksced","ha
                              "mentemen","csnyari","sekitarkontak","pindahnomornya","sept","duren","ngidol","uwaw","amp","bapuq","co",
                              "maghrib","jkt","jbr","karawang","diwa","dtng","bolee","gratong","adm","mnta","tmpol","kresi","freeong",
                              "cth","smpt","nde","amh","neh","asu","tenan","wes","samarinda","b","pu","ju","au","rambakdong","paxel","lion
                              "fyi","rambakdongkeripik","scoopy","wts","doyoung","applewood","pob","labiar","yssc","banhhh","nieee","hsksh
                              "kemsrin","mbb","medan","jakartasumedang","wkwkwkwk","ngrugiin","aaaaaaa","krn","bantul","klaten","anjayyy",
                              "aing","gawe","dll","tkprpy","tkpywqxw","yawlaa","seandainy","pekayon","bekasi","jakasetia","opo","cipeucang
                              "bogor","jirr","hyunsuk","jihoon","jo","malone","unoff","eeeanjir","denpasar","segercep","dehga","gbl","ngur
                              "macii","dket","kim","taehyung","namjoon","emsnya","aktf","awikwo","hhhh","tokped","sleman","sihhhhhhhh","he
                              "tlfon","owkiiee","kudus","harisuwi","urong","kliru","lek","lowokwaru","thorr","tulungann","sprti","anjeeeee
                              "zimbabwe","cisarua","syafrul","rasidin","jabodetabek","tanggerang","cilandak","dkk","iki","brrti","offlane"
                              "anjrittt","jombang","samsek","oake","waline","engres","jingkontop","bekazi","bnran","ta","mo","ngoni","pe",
                              "slawi","tegal","pemalang","mlhan","rak","digowo","wes","tak","dewe","ngawi","hahshabahsbs","ditutungguan","
                              "ngadon","ambip","tekoteko","isokkk","sekkiyaaaaaa","sibal","denpasar","wkwkwkwk","mereun","jnejntanterajas
                              "raimu","nggedabrus","cuiii","cimanggis","manokwari","pisuhi","tekke","nggonku","bakdo","hadiahmending","jas
                              "paninggaran","pekalongan","mncoba","wkwktau","juseyo","matthew","mbasan","teko","ndodoki","pisan","suarane"
                              "anake","tangi","agi","arep","metu","selehke","nduwur","lawang","klean","arraagghhh","sajaminnnnn","hayoh",
                              "wkwkwkw","dgsgsjeks","jlngilaaasehhh","emanggf","customercarecom","enete","tkpkheuufe","makanane","kaliren"
                              "diterno","sedino","biasae","garuttt","gratongnya","rhezza","barok","baekbaek","ngijo","suga","yoongi","jung
                              "nyenyenyee","xiumin","gege","wuakakakaka","kudus","ikbal","katilayu","kalimaya","ariyanto","bambang","sulis
                              "kebumen","bolongsobek","sajasobek","jember","balung","dudy","imoet","nder","sudazfa","letisia","fafifu","le
                              "sikurur","zaenal","ngeterno","retnobisa","yohana","sicepatantarajadan","giovani","indihiang","gunung","dran
                              "sirfadika","dobel","kawankawan","bakulbakul","dhewe","dodolan","shampoo","maulana","seokjin","cilandak","ga
                              "invmpl","maksudne","leuweung","dbshwhhdhsha","indihiang","daeu","pickuo","manehna","wkwkwkwkwk","keluarz","
                              "pdgbkt","pdgpjg","panongan","sporttimeid","qmjjdpy","ammmmiiinnnn","asalmualaikm","sicepatanteraja","ampm",
                              "rancaekek","gaisdiantara","jnejntsm","digowo","saiki","durung","ambek","omahku","berbelitbelit","invxxii","
```



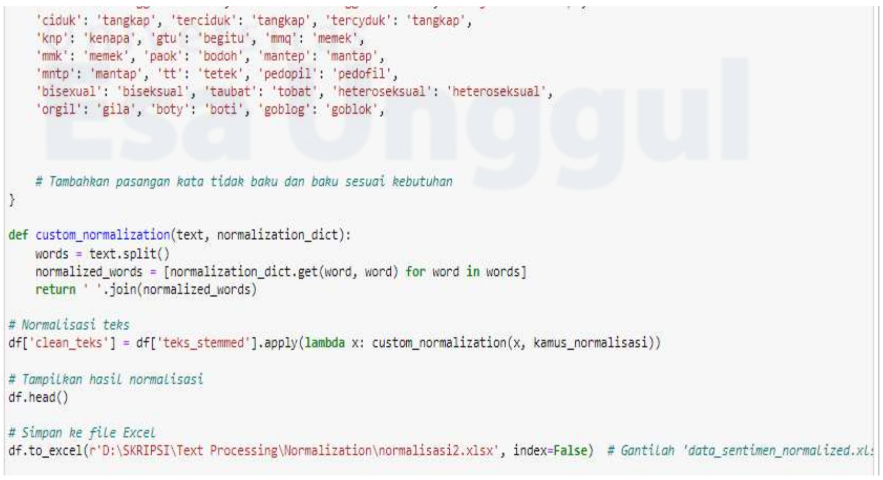
```
                              "gpertolblack","eblackmisty","bulfaro","cnavyyellow","byelloworange","swissmohon","kepastiankalau","sahabat
                              "bandungmakassar","progressrasanya","jktmgl","rakkipanda","originalthanks","pakditunggu","cabjatibening","c
                              "mbakhahaha","bleek","dipatiukur","liyane","saiki","teko","flish","yhbeda","mch","fiqri","chandraesar","tenn
                              "trenggalek","padahalbarangnya","wulanhandayani","arilassotv","taufiqrahman","anyaranberarti","banyuwangija
                              "gouacheku","gorong","henteu","mosok","ayeumamah","siah","deui","esumpahh","tkaatfjbnee","binkarma","ikarma"
                              "unclejonbarber","eggroll","jakartadepok","jlmulyaaariperum","tamansari","kmjrenjangan","riaukampar","smpng"
                              "appakunku","ireneamp","seulgi","tjnesicepatgojekcod","agustusmhn","ayamwortel","herman","makassarbontang","
                              "tokecang","sangatta","herdianto","rangomongke","elek","ngomongke","ndara","ngalahke","henggarae","wakakak",
         stop_words_sastrawi = stopword_factory.get_stop_words() + custom_stop_words

         # Menggabungkan stopword dari NLTK dan Sastrawi
         stop_words_nltk = set(stopwords.words('indonesian', 'english'))
         stop_words = stop_words_nltk.union(stop_words_sastrawi)

         # Fungsi untuk menghapus stopword dari teks token
         def remove_stopwords(text):
             if isinstance(text, str):
                 word_tokens = word_tokenize(text)
                 filtered_tokens = [word for word in word_tokens if word.lower() not in stop_words]
                 return ' '.join(filtered_tokens)
             else:
                 return text  # Mengembalikan teks asli jika bukan string

         # Mengaplikasikan fungsi pada kolom teks
         df['cleaned_text'] = df['text'].apply(remove_stopwords)

         # Menyimpan dataframe yang telah diupdate ke file Excel
         df.to_excel('D:\SKRIPSI\Text Processing\hasil_andre.xlsx', index=False)  # Sesuaikan dengan path yang diinginkan


         # Menyimpan dataframe yang telah diupdate ke file Excel
         df.to_excel('D:\SKRIPSI\Text Processing\hasil eksa.xlsx', index=False)  # Gantilah dengan path sesuai keinginan Anda
```



```
In [1]: #Stemming
        import pandas as pd
        from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

        # Fungsi untuk melakukan stemming pada teks
        def stem_text(text):
            factory = StemmerFactory()
            stemmer = factory.create_stemmer()
            return stemmer.stem(text)

        # Membaca file Excel
        df = pd.read_excel(r'D:\SKRIPSI\Text Processing\Stopword Removal\eksperimen stopword.xlsx')

        # Melakukan stemming pada kolom 'teks'
        df['teks_stemmed'] = df['cleaned_text'].apply(stem_text)

        # Menyimpan hasil stemming kembali ke file Excel
        df.to_excel(r'D:\SKRIPSI\Text Processing\Stopword Removal\hasil eksperimen stopword.xlsx', index=False, engine='openpyxl')
```

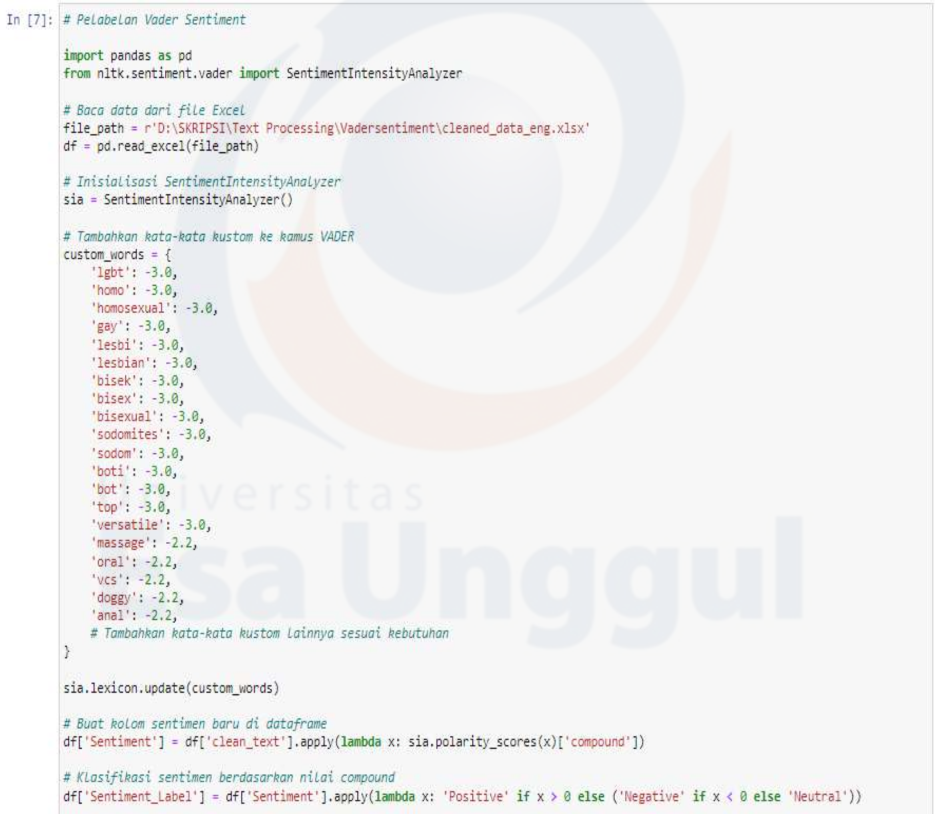


```
In [2]: #Normalization
        import pandas as pd

        # Baca file Excel
        df = pd.read_excel (r'D:\SKRIPSI\Text Processing\Normalization\normalisasi1.xlsx')

        # Daftar kamus normalisasi
        kamus_normalisasi = {
            'zinah': 'zina', 'gr': 'gara', 'apus': 'hapus',
            'turu': 'tidur', 'cwe': 'cewek', 'abiz': 'habis',
            'sange': 'nafsu', 'cwek': 'cewek', 'sange': 'nafsu',
            'pijit': 'pijat', 'cwok': 'cowok', 'isap': 'hisap',
            'bobo': 'bobok', 'bocil': 'anak kecil', 'capek': 'cape',
            'kntl': 'kontol', 'ntar': 'nanti', 'syurga': 'surga',
            'gawe': 'kerja', 'kagak': 'tidak', 'makasih': 'terima kasih',
            'orng': 'orang', 'org': 'orang', 'cowo': 'cowok',
            'massage': 'pijat', 'ready': 'siap', 'ewe': 'bersetubuh',
            'ngewe': 'bersetubuh', 'ngentot': 'entot',
            'cantikkuuu': 'cantik', 'bo': 'pesan', 'slim': 'kurus',
            'lngsung': 'langsung', 'order': 'pesan', 'slim': 'kurus',
            'jngan': 'jangan', 'kmu': 'kamu', 'jngn': 'jangan',
            'sy': 'saya', 'tp': 'tapi', 'nyari': 'cari',
            'nampil': 'tampil', 'meni': 'nikah', 'nangkep': 'tangkap',
            'taboo': 'tabu', 'spkt': 'setuju', 'tapiii': 'tapi',
            'gamau': 'tidak mau', 'bnr': 'benar', 'kudu': 'harus',
            'gk': 'tidak', 'bgitu': 'begitu', 'ken': 'ingin',
            'jdi': 'jadi', 'jd': 'jadi', 'kenape': 'kenapa',
            'bangsaddddd': 'bangsat', 'jd': 'jadi', 'kenape': 'kenapa',
            'betina': 'wanita', 'brarti': 'berarti', 'kenape': 'kenapa',
            'mampos': 'mampus', 'bodo': 'bodoh', 'bgsd': 'bangsat',
            'bgst': 'bangsat', 'nipu': 'tipu', 'cantikk': 'cantik',
            'cewe': 'cewek', 'anj': 'anjing', 'anjg': 'anjing',
            'gilaa': 'gila', 'sdh': 'sudah', 'tdk': 'tidak',
            'bnr': 'benar', 'gini': 'begini', 'gitu': 'begitu',
            'bgt': 'banget', 'nganggep': 'anggap', 'anggep': 'anggap',
            'makasih': 'terima kasih', 'ngoceh': 'oceh', 'kberkahan': 'berkah',
            'kyk': 'seperti', 'trs': 'terus', 'bat': 'sangat',
            'udh': 'sudah', 'mrk': 'mereka', 'bncana': 'bencana',
            'gilak': 'gila', 'gilaak': 'gila', 'gey': 'gay',
            'gpp': 'tidak apa apa', 'nyolek': 'colek', 'ngerangkul': 'rangkul',
            'blg': 'bilang', 'mmg': 'memang', 'kocakk': 'kocak',
            'tlol': 'tolol', 'ajg': 'anjing', 'anjg': 'anjing',
            'mnyt': 'monyet', 'dsr': 'dasar', 'mnyimpang': 'menyimpang',
            'gaada': 'tidak ada', 'pdofil': 'pedofil', 'gausah': 'tidak usah',
            'faham': 'paham', 'gatau': 'tidak tahu', 'capek': 'cape',
            'lgsg': 'langsung', 'bntar': 'bentar', 'bntr': 'bentar',
            'bkn': 'bukan', 'dianggep': 'anggap', 'astoge': 'astaga',
            'maafin': 'maaf', 'rem': 'seram', 'gin': 'begini',
            'ngulum': 'isap', 'ngulom': 'isap', 'gabut': 'bosan',
            'emut': 'isap', 'ngemutin': 'isap', 'kocokin': 'kocok',
            'sepong': 'isap', 'nyepong': 'isap', 'blowjob': 'isap',
            'blowjobr': 'isap', 'oblowjobek': 'isap', 'sublowjobektif': 'isap',
            'skit': 'sakit', 'gedeg': 'kesal', 'jblowjobb': 'isap',
            'ttg': 'tentang', 'tntg': 'tentang', 'jiji': 'jijik',
            'katro': 'ketinggalan zaman', 'katrok': 'ketinggalan zaman', 'blowjobrot': 'isap',
```

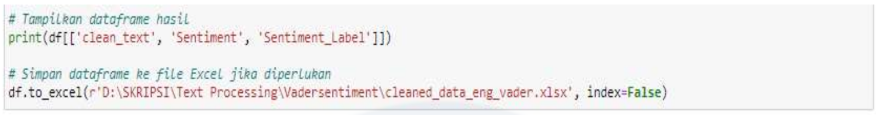


```
    'ciduk': 'tangkap', 'terciduk': 'tangkap', 'tercyduk': 'tangkap',
    'knp': 'kenapa', 'gtu': 'begitu', 'mmq': 'memek',
    'mmk': 'memek', 'paok': 'bodoh', 'mantep': 'mantap',
    'mntp': 'mantap', 'tt': 'tetek', 'pedopil': 'pedofil',
    'bisexual': 'biseksual', 'taubat': 'tobat', 'heteroseksual': 'heteroseksual',
    'orgil': 'gila', 'boty': 'boti', 'goblog': 'goblok',


    # Tambahkan pasangan kata tidak baku dan baku sesuai kebutuhan
}

def custom_normalization(text, normalization_dict):
    words = text.split()
    normalized_words = [normalization_dict.get(word, word) for word in words]
    return ' '.join(normalized_words)

# Normalisasi teks
df['clean_teks'] = df['teks_stemmed'].apply(lambda x: custom_normalization(x, kamus_normalisasi))

# Tampilkan hasil normalisasi
df.head()

# Simpan ke file Excel
df.to_excel(r'D:\SKRIPSI\Text Processing\Normalization\normalisasi2.xlsx', index=False)  # Gantilah 'data_sentimen_normalized.xl
```

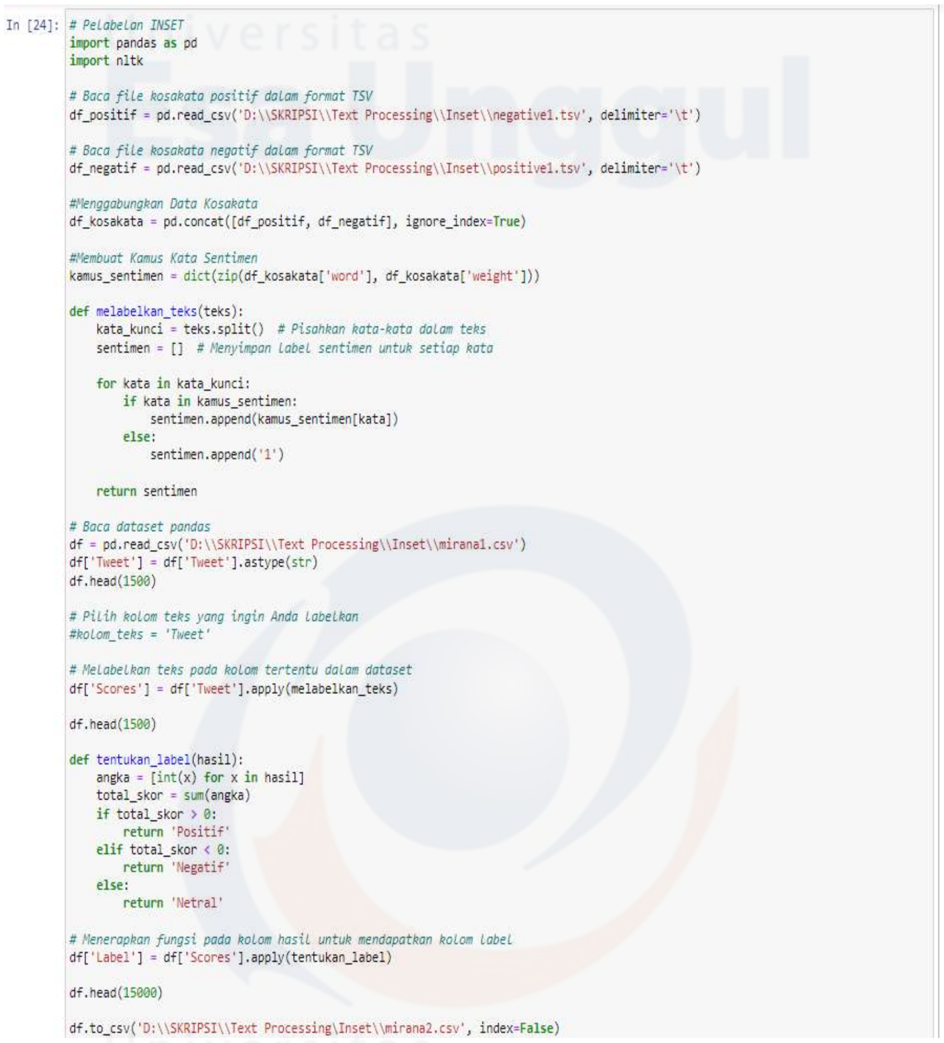
Lampiran 3 – Pelabelan



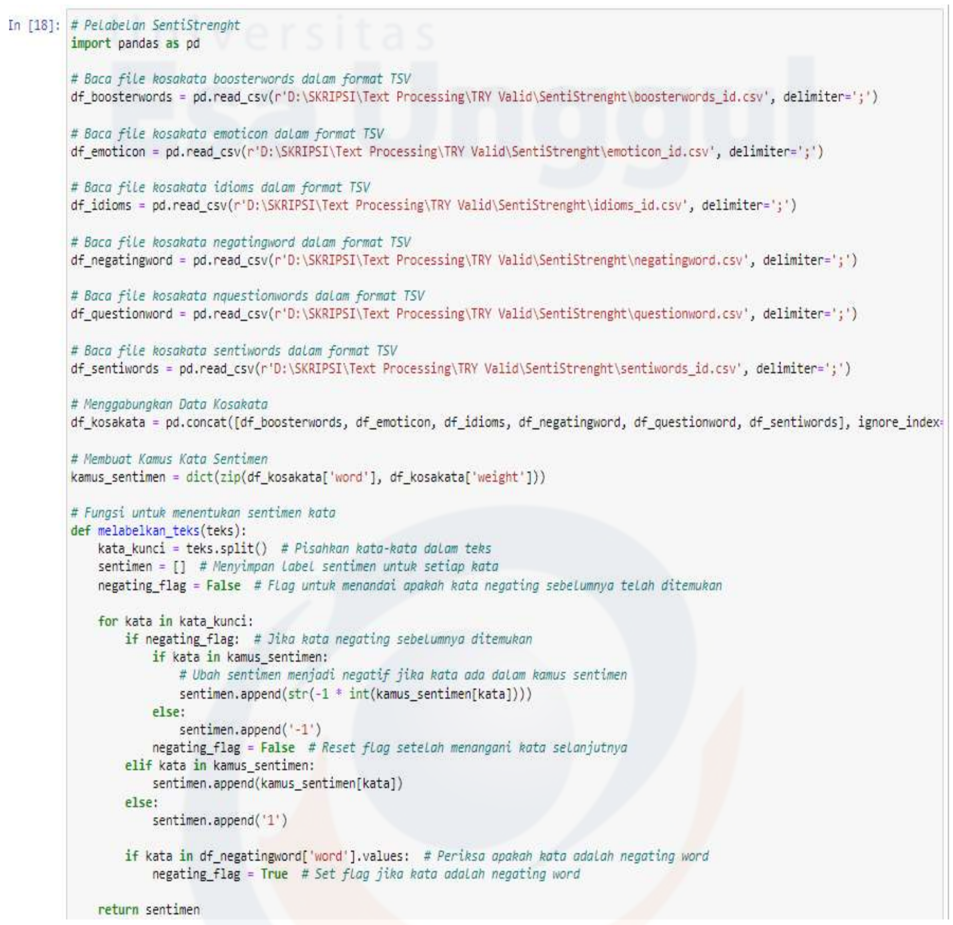
```
In [7]: # Pelabelan Vader Sentiment

        import pandas as pd
        from nltk.sentiment.vader import SentimentIntensityAnalyzer

        # Baca data dari file Excel
        file_path = r'D:\SKRIPSI\Text Processing\Vadersentiment\cleaned_data_eng.xlsx'
        df = pd.read_excel(file_path)

        # Inisialisasi SentimentIntensityAnalyzer
        sia = SentimentIntensityAnalyzer()

        # Tambahkan kata-kata kustom ke kamus VADER
        custom_words = {
            'lgbt': -3.0,
            'homo': -3.0,
            'homosexual': -3.0,
            'gay': -3.0,
            'lesbi': -3.0,
            'lesbian': -3.0,
            'bisek': -3.0,
            'bisex': -3.0,
            'bisexual': -3.0,
            'sodomites': -3.0,
            'sodom': -3.0,
            'boti': -3.0,
            'bot': -3.0,
            'top': -3.0,
            'versatile': -3.0,
            'massage': -2.2,
            'oral': -2.2,
            'vcs': -2.2,
            'doggy': -2.2,
            'anal': -2.2,
            # Tambahkan kata-kata kustom lainnya sesuai kebutuhan
        }

        sia.lexicon.update(custom_words)

        # Buat kolom sentimen baru di dataframe
        df['Sentiment'] = df['clean_text'].apply(lambda x: sia.polarity_scores(x)['compound'])

        # Klasifikasi sentimen berdasarkan nilai compound
        df['Sentiment_Label'] = df['Sentiment'].apply(lambda x: 'Positive' if x > 0 else ('Negative' if x < 0 else 'Neutral'))
```



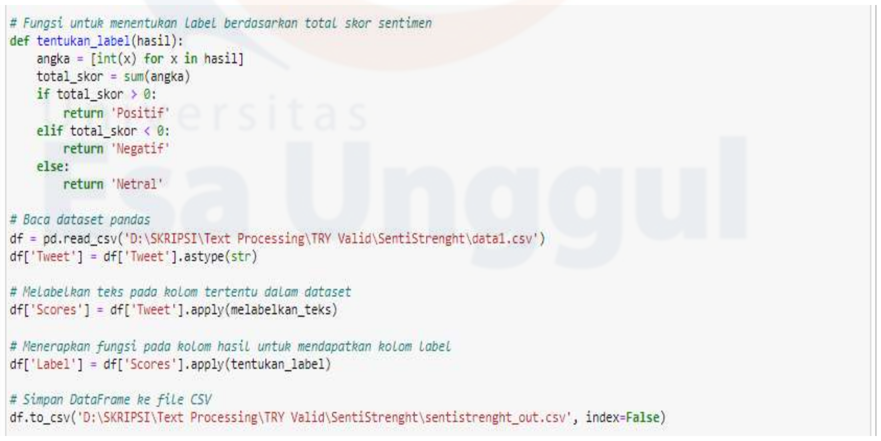
```
# Tampilkan dataframe hasil
print(df[['clean_text', 'Sentiment', 'Sentiment_Label']])

# Simpan dataframe ke file Excel jika diperlukan
df.to_excel(r'D:\SKRIPSI\Text Processing\Vadersentiment\cleaned_data_eng_vader.xlsx', index=False)
```



```
In [24]: # Pelabelan INSET
         import pandas as pd
         import nltk

         # Baca file kosakata positif dalam format TSV
         df_positif = pd.read_csv('D:\\SKRIPSI\\Text Processing\\Inset\\negative1.tsv', delimiter='\t')

         # Baca file kosakata negatif dalam format TSV
         df_negatif = pd.read_csv('D:\\SKRIPSI\\Text Processing\\Inset\\positive1.tsv', delimiter='\t')

         #Menggabungkan Data Kosakata
         df_kosakata = pd.concat([df_positif, df_negatif], ignore_index=True)

         #Membuat Kamus Kata Sentimen
         kamus_sentimen = dict(zip(df_kosakata['word'], df_kosakata['weight']))

         def melabelkan_teks(teks):
             kata_kunci = teks.split()  # Pisahkan kata-kata dalam teks
             sentimen = []  # Menyimpan label sentimen untuk setiap kata

             for kata in kata_kunci:
                 if kata in kamus_sentimen:
                     sentimen.append(kamus_sentimen[kata])
                 else:
                     sentimen.append('1')

             return sentimen

         # Baca dataset pandas
         df = pd.read_csv('D:\\SKRIPSI\\Text Processing\\Inset\\mirana1.csv')
         df['Tweet'] = df['Tweet'].astype(str)
         df.head(1500)

         # Pilih kolom teks yang ingin Anda labelkan
         #kolom_teks = 'Tweet'

         # Melabelkan teks pada kolom tertentu dalam dataset
         df['Scores'] = df['Tweet'].apply(melabelkan_teks)

         df.head(1500)

         def tentukan_label(hasil):
             angka = [int(x) for x in hasil]
             total_skor = sum(angka)
             if total_skor > 0:
                 return 'Positif'
             elif total_skor < 0:
                 return 'Negatif'
             else:
                 return 'Netral'

         # Menerapkan fungsi pada kolom hasil untuk mendapatkan kolom label
         df['Label'] = df['Scores'].apply(tentukan_label)

         df.head(15000)

         df.to_csv('D:\\SKRIPSI\\Text Processing\Inset\\mirana2.csv', index=False)
```

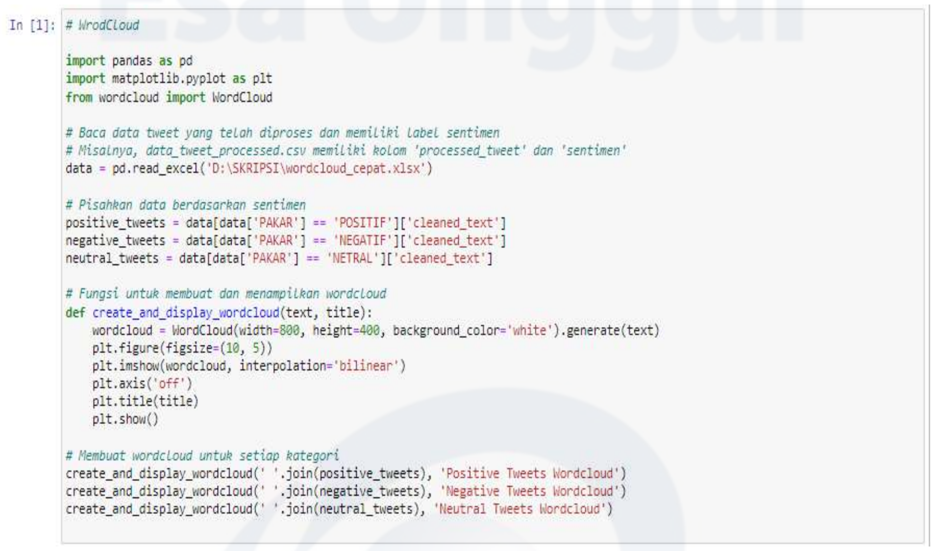


```
In [18]: # Pelabelan SentiStrenght
         import pandas as pd

         # Baca file kosakata boosterwords dalam format TSV
         df_boosterwords = pd.read_csv(r'D:\SKRIPSI\Text Processing\TRY Valid\SentiStrenght\boosterwords_id.csv', delimiter=';')

         # Baca file kosakata emoticon dalam format TSV
         df_emoticon = pd.read_csv(r'D:\SKRIPSI\Text Processing\TRY Valid\SentiStrenght\emoticon_id.csv', delimiter=';')

         # Baca file kosakata idioms dalam format TSV
         df_idioms = pd.read_csv(r'D:\SKRIPSI\Text Processing\TRY Valid\SentiStrenght\idioms_id.csv', delimiter=';')

         # Baca file kosakata negatingword dalam format TSV
         df_negatingword = pd.read_csv(r'D:\SKRIPSI\Text Processing\TRY Valid\SentiStrenght\negatingword.csv', delimiter=';')

         # Baca file kosakata nquestionwords dalam format TSV
         df_questionword = pd.read_csv(r'D:\SKRIPSI\Text Processing\TRY Valid\SentiStrenght\questionword.csv', delimiter=';')

         # Baca file kosakata sentiwords dalam format TSV
         df_sentiwords = pd.read_csv(r'D:\SKRIPSI\Text Processing\TRY Valid\SentiStrenght\sentiwords_id.csv', delimiter=';')

         # Menggabungkan Data Kosakata
         df_kosakata = pd.concat([df_boosterwords, df_emoticon, df_idioms, df_negatingword, df_questionword, df_sentiwords], ignore_index

         # Membuat Kamus Kata Sentimen
         kamus_sentimen = dict(zip(df_kosakata['word'], df_kosakata['weight']))

         # Fungsi untuk menentukan sentimen kata
         def melabelkan_teks(teks):
             kata_kunci = teks.split()  # Pisahkan kata-kata dalam teks
             sentimen = []  # Menyimpan label sentimen untuk setiap kata
             negating_flag = False  # Flag untuk menandai apakah kata negating sebelumnya telah ditemukan

             for kata in kata_kunci:
                 if negating_flag:  # Jika kata negating sebelumnya ditemukan
                     if kata in kamus_sentimen:
                         # Ubah sentimen menjadi negatif jika kata ada dalam kamus sentimen
                         sentimen.append(str(-1 * int(kamus_sentimen[kata])))
                     else:
                         sentimen.append('-1')
                     negating_flag = False  # Reset flag setelah menangani kata selanjutnya
                 elif kata in kamus_sentimen:
                     sentimen.append(kamus_sentimen[kata])
                 else:
                     sentimen.append('1')

                 if kata in df_negatingword['word'].values:  # Periksa apakah kata adalah negating word
                     negating_flag = True  # Set flag jika kata adalah negating word

             return sentimen
```



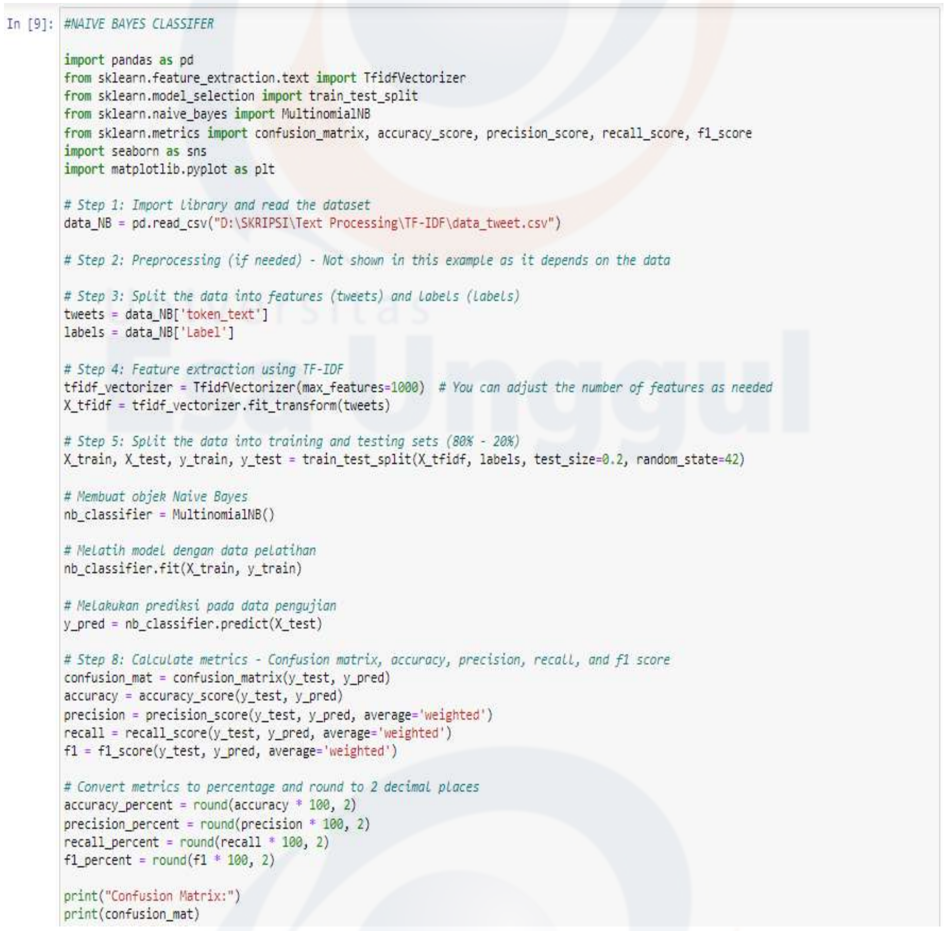
```
# Fungsi untuk menentukan label berdasarkan total skor sentimen
def tentukan_label(hasil):
    angka = [int(x) for x in hasil]
    total_skor = sum(angka)
    if total_skor > 0:
        return 'Positif'
    elif total_skor < 0:
        return 'Negatif'
    else:
        return 'Netral'

# Baca dataset pandas
df = pd.read_csv('D:\SKRIPSI\Text Processing\TRY Valid\SentiStrenght\data1.csv')
df['Tweet'] = df['Tweet'].astype(str)

# Melabelkan teks pada kolom tertentu dalam dataset
df['Scores'] = df['Tweet'].apply(melabelkan_teks)

# Menerapkan fungsi pada kolom hasil untuk mendapatkan kolom label
df['Label'] = df['Scores'].apply(tentukan_label)

# Simpan DataFrame ke file CSV
df.to_csv('D:\SKRIPSI\Text Processing\TRY Valid\SentiStrenght\sentistrenght_out.csv', index=False)
```

Lampiran 4 - WordCloud

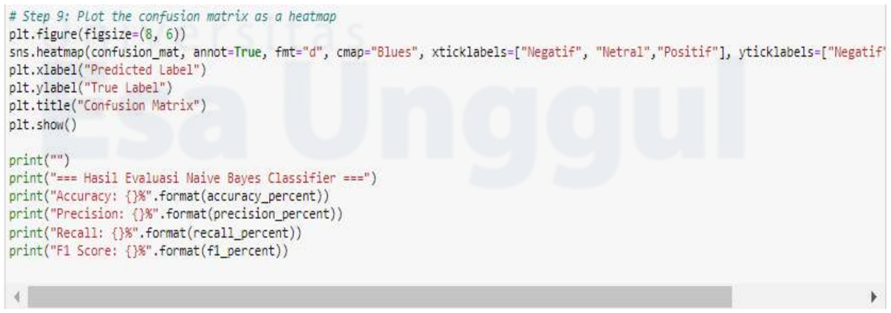


```
In [1]: # WrodCloud

        import pandas as pd
        import matplotlib.pyplot as plt
        from wordcloud import WordCloud

        # Baca data tweet yang telah diproses dan memiliki label sentimen
        # Misalnya, data_tweet_processed.csv memiliki kolom 'processed_tweet' dan 'sentimen'
        data = pd.read_excel('D:\SKRIPSI\wordcloud_cepat.xlsx')

        # Pisahkan data berdasarkan sentimen
        positive_tweets = data[data['PAKAR'] == 'POSITIF']['cleaned_text']
        negative_tweets = data[data['PAKAR'] == 'NEGATIF']['cleaned_text']
        neutral_tweets = data[data['PAKAR'] == 'NETRAL']['cleaned_text']

        # Fungsi untuk membuat dan menampilkan wordcloud
        def create_and_display_wordcloud(text, title):
            wordcloud = WordCloud(width=800, height=400, background_color='white').generate(text)
            plt.figure(figsize=(10, 5))
            plt.imshow(wordcloud, interpolation='bilinear')
            plt.axis('off')
            plt.title(title)
            plt.show()

        # Membuat wordcloud untuk setiap kategori
        create_and_display_wordcloud(' '.join(positive_tweets), 'Positive Tweets Wordcloud')
        create_and_display_wordcloud(' '.join(negative_tweets), 'Negative Tweets Wordcloud')
        create_and_display_wordcloud(' '.join(neutral_tweets), 'Neutral Tweets Wordcloud')
```

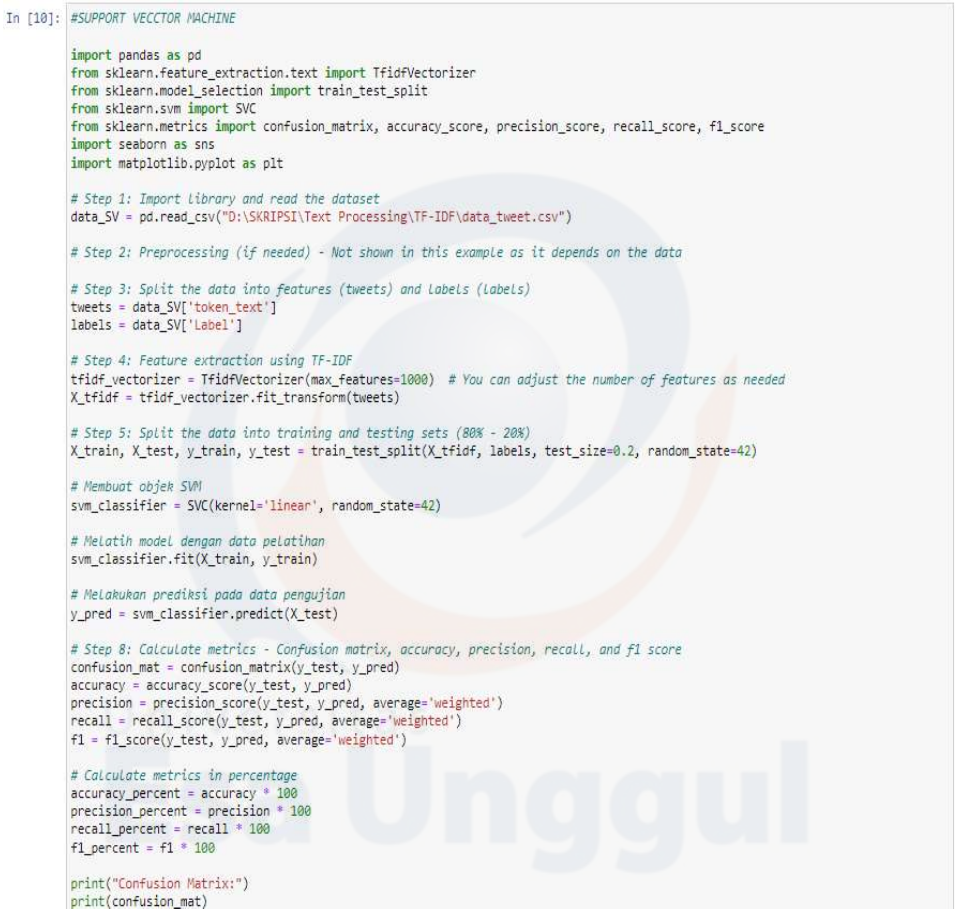
Lampiran 5 – Klasifikasi



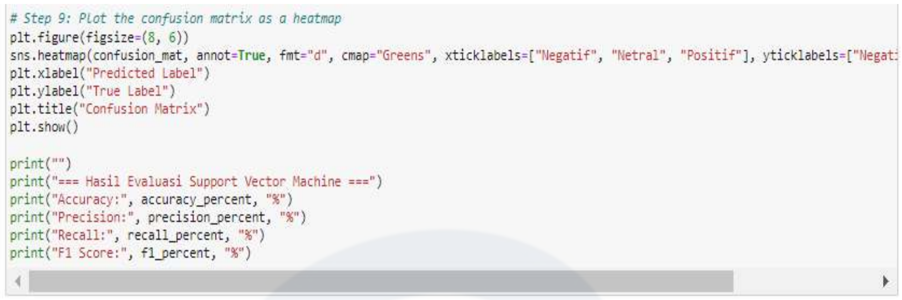
```
In [9]: #NAIVE BAYES CLASSIFER

        import pandas as pd
        from sklearn.feature_extraction.text import TfidfVectorizer
        from sklearn.model_selection import train_test_split
        from sklearn.naive_bayes import MultinomialNB
        from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
        import seaborn as sns
        import matplotlib.pyplot as plt

        # Step 1: Import library and read the dataset
        data_NB = pd.read_csv("D:\SKRIPSI\Text Processing\TF-IDF\data_tweet.csv")

        # Step 2: Preprocessing (if needed) - Not shown in this example as it depends on the data

        # Step 3: Split the data into features (tweets) and labels (labels)
        tweets = data_NB['token_text']
        labels = data_NB['Label']

        # Step 4: Feature extraction using TF-IDF
        tfidf_vectorizer = TfidfVectorizer(max_features=1000)  # You can adjust the number of features as needed
        X_tfidf = tfidf_vectorizer.fit_transform(tweets)

        # Step 5: Split the data into training and testing sets (80% - 20%)
        X_train, X_test, y_train, y_test = train_test_split(X_tfidf, labels, test_size=0.2, random_state=42)

        # Membuat objek Naive Bayes
        nb_classifier = MultinomialNB()

        # Melatih model dengan data pelatihan
        nb_classifier.fit(X_train, y_train)

        # Melakukan prediksi pada data pengujian
        y_pred = nb_classifier.predict(X_test)

        # Step 8: Calculate metrics - Confusion matrix, accuracy, precision, recall, and f1 score
        confusion_mat = confusion_matrix(y_test, y_pred)
        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred, average='weighted')
        recall = recall_score(y_test, y_pred, average='weighted')
        f1 = f1_score(y_test, y_pred, average='weighted')

        # Convert metrics to percentage and round to 2 decimal places
        accuracy_percent = round(accuracy * 100, 2)
        precision_percent = round(precision * 100, 2)
        recall_percent = round(recall * 100, 2)
        f1_percent = round(f1 * 100, 2)

        print("Confusion Matrix:")
        print(confusion_mat)
```



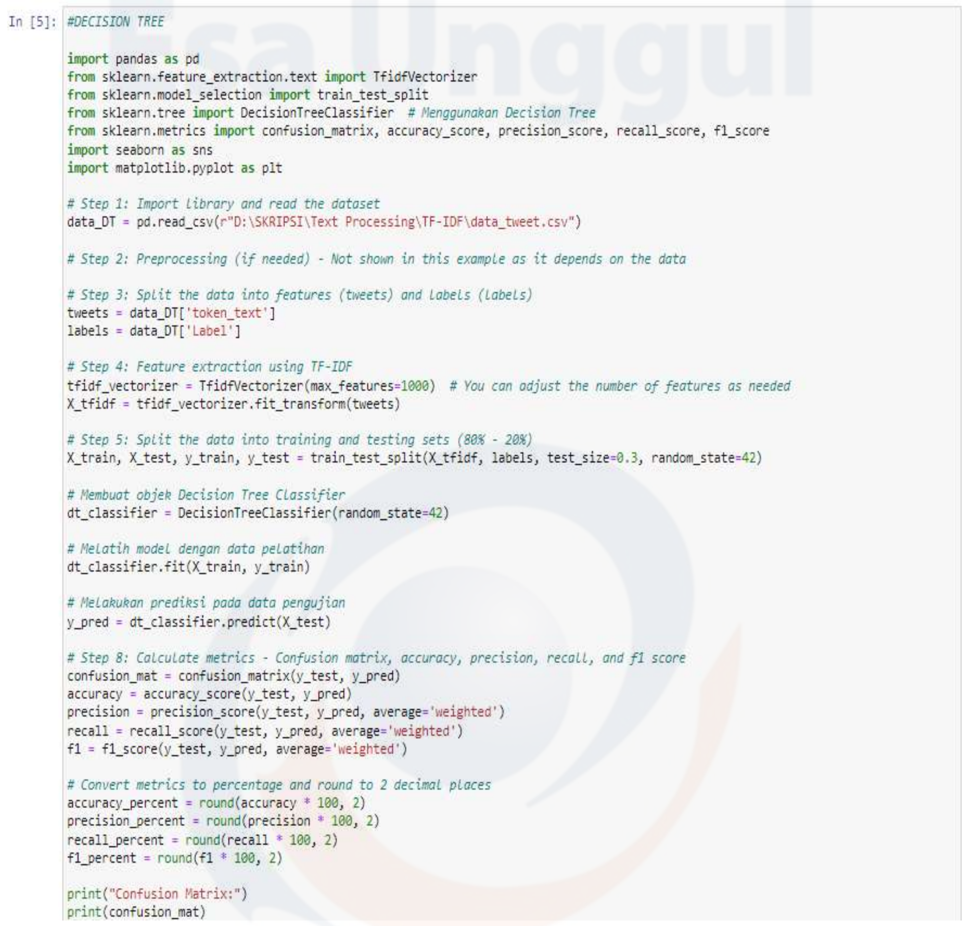
```
# Step 9: Plot the confusion matrix as a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(confusion_mat, annot=True, fmt="d", cmap="Blues", xticklabels=["Negatif", "Netral","Positif"], yticklabels=["Negatif"
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.show()

print("")
print("=== Hasil Evaluasi Naive Bayes Classifier ===")
print("Accuracy: {}%".format(accuracy_percent))
print("Precision: {}%".format(precision_percent))
print("Recall: {}%".format(recall_percent))
print("F1 Score: {}%".format(f1_percent))
```



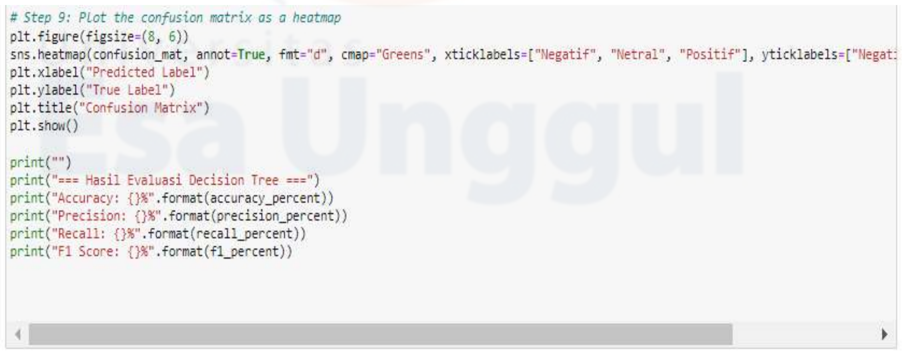
```
In [10]: #SUPPORT VECCTOR MACHINE

import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
import seaborn as sns
import matplotlib.pyplot as plt

# Step 1: Import library and read the dataset
data_SV = pd.read_csv("D:\SKRIPSI\Text Processing\TF-IDF\data_tweet.csv")

# Step 2: Preprocessing (if needed) - Not shown in this example as it depends on the data

# Step 3: Split the data into features (tweets) and labels (labels)
tweets = data_SV['token_text']
labels = data_SV['Label']

# Step 4: Feature extraction using TF-IDF
tfidf_vectorizer = TfidfVectorizer(max_features=1000)  # You can adjust the number of features as needed
X_tfidf = tfidf_vectorizer.fit_transform(tweets)

# Step 5: Split the data into training and testing sets (80% - 20%)
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, labels, test_size=0.2, random_state=42)

# Membuat objek SVM
svm_classifier = SVC(kernel='linear', random_state=42)

# Melatih model dengan data pelatihan
svm_classifier.fit(X_train, y_train)

# Melakukan prediksi pada data pengujian
y_pred = svm_classifier.predict(X_test)

# Step 8: Calculate metrics - Confusion matrix, accuracy, precision, recall, and f1 score
confusion_mat = confusion_matrix(y_test, y_pred)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

# Calculate metrics in percentage
accuracy_percent = accuracy * 100
precision_percent = precision * 100
recall_percent = recall * 100
f1_percent = f1 * 100

print("Confusion Matrix:")
print(confusion_mat)
```



```
# Step 9: Plot the confusion matrix as a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(confusion_mat, annot=True, fmt="d", cmap="Greens", xticklabels=["Negatif", "Netral", "Positif"], yticklabels=["Negati
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.show()

print("")
print("=== Hasil Evaluasi Support Vector Machine ===")
print("Accuracy:", accuracy_percent, "%")
print("Precision:", precision_percent, "%")
print("Recall:", recall_percent, "%")
print("F1 Score:", f1_percent, "%")
```

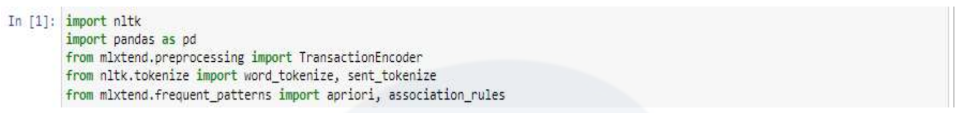


```
In [5]: #DECISION TREE

        import pandas as pd
        from sklearn.feature_extraction.text import TfidfVectorizer
        from sklearn.model_selection import train_test_split
        from sklearn.tree import DecisionTreeClassifier  # Menggunakan Decision Tree
        from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
        import seaborn as sns
        import matplotlib.pyplot as plt

        # Step 1: Import library and read the dataset
        data_DT = pd.read_csv(r"D:\SKRIPSI\Text Processing\TF-IDF\data_tweet.csv")

        # Step 2: Preprocessing (if needed) - Not shown in this example as it depends on the data

        # Step 3: Split the data into features (tweets) and labels (labels)
        tweets = data_DT['token_text']
        labels = data_DT['Label']

        # Step 4: Feature extraction using TF-IDF
        tfidf_vectorizer = TfidfVectorizer(max_features=1000)  # You can adjust the number of features as needed
        X_tfidf = tfidf_vectorizer.fit_transform(tweets)

        # Step 5: Split the data into training and testing sets (80% - 20%)
        X_train, X_test, y_train, y_test = train_test_split(X_tfidf, labels, test_size=0.3, random_state=42)

        # Membuat objek Decision Tree Classifier
        dt_classifier = DecisionTreeClassifier(random_state=42)

        # Melatih model dengan data pelatihan
        dt_classifier.fit(X_train, y_train)

        # Melakukan prediksi pada data pengujian
        y_pred = dt_classifier.predict(X_test)

        # Step 8: Calculate metrics - Confusion matrix, accuracy, precision, recall, and f1 score
        confusion_mat = confusion_matrix(y_test, y_pred)
        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred, average='weighted')
        recall = recall_score(y_test, y_pred, average='weighted')
        f1 = f1_score(y_test, y_pred, average='weighted')

        # Convert metrics to percentage and round to 2 decimal places
        accuracy_percent = round(accuracy * 100, 2)
        precision_percent = round(precision * 100, 2)
        recall_percent = round(recall * 100, 2)
        f1_percent = round(f1 * 100, 2)

        print("Confusion Matrix:")
        print(confusion_mat)
```

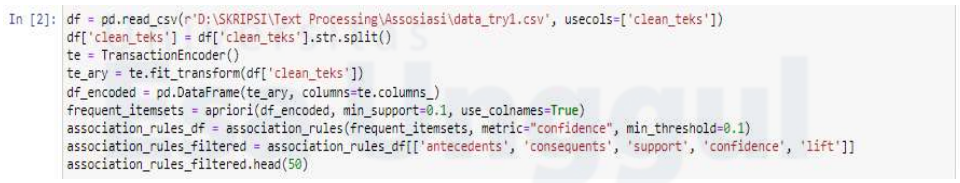


```
# Step 9: Plot the confusion matrix as a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(confusion_mat, annot=True, fmt="d", cmap="Greens", xticklabels=["Negatif", "Netral", "Positif"], yticklabels=["Negati
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.show()

print("")
print("=== Hasil Evaluasi Decision Tree ===")
print("Accuracy: {}%".format(accuracy_percent))
print("Precision: {}%".format(precision_percent))
print("Recall: {}%".format(recall_percent))
print("F1 Score: {}%".format(f1_percent))
```

Lampiran 6 – Kode Identifikasi Relasi Kata

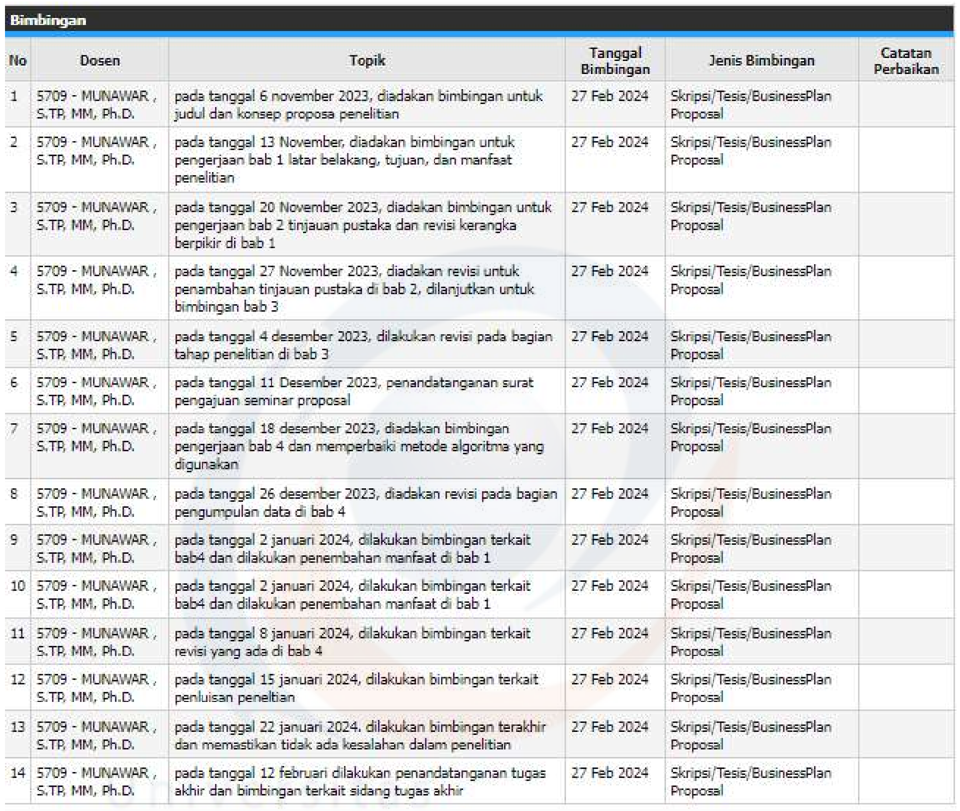


```
In [1]: import nltk
        import pandas as pd
        from mlxtend.preprocessing import TransactionEncoder
        from nltk.tokenize import word_tokenize, sent_tokenize
        from mlxtend.frequent_patterns import apriori, association_rules
```



```
In [2]: df = pd.read_csv(r'D:\SKRIPSI\Text Processing\Assosiasi\data_try1.csv', usecols=['clean_teks'])
        df['clean_teks'] = df['clean_teks'].str.split()
        te = TransactionEncoder()
        te_ary = te.fit_transform(df['clean_teks'])
        df_encoded = pd.DataFrame(te_ary, columns=te.columns_)
        frequent_itemsets = apriori(df_encoded, min_support=0.1, use_colnames=True)
        association_rules_df = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.1)
        association_rules_filtered = association_rules_df[['antecedents', 'consequents', 'support', 'confidence', 'lift']]
        association_rules_filtered.head(50)
```

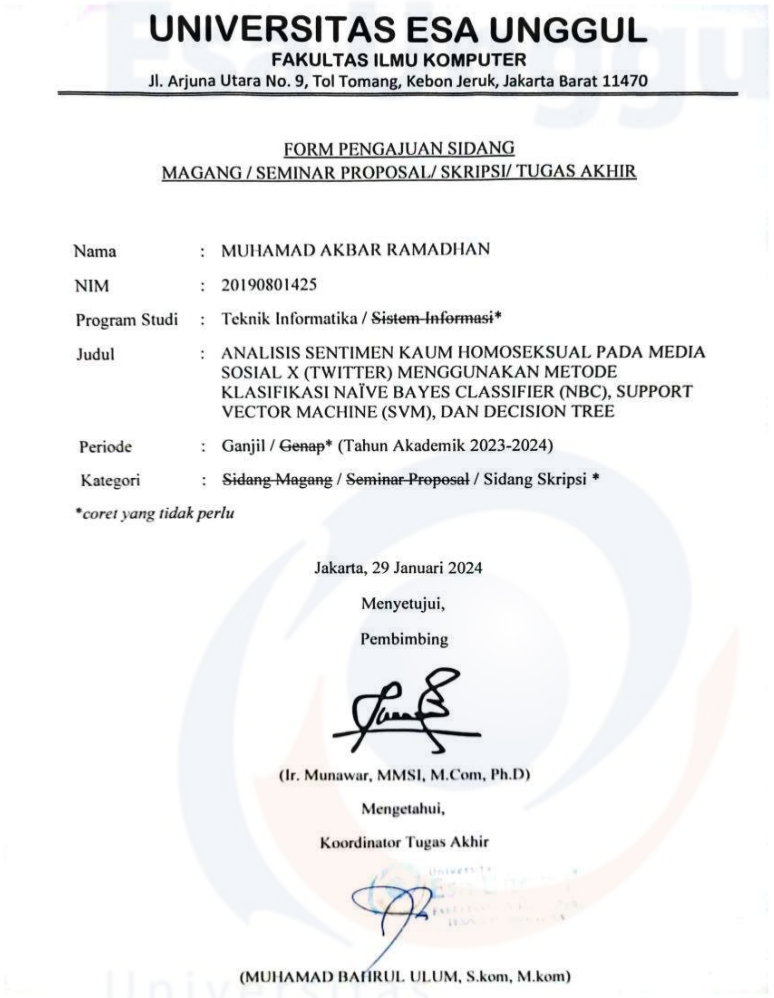
Lampiran 7 – Daftar Bimbingan



Bimbingan

No	Dosen	Topik	Tanggal Bimbingan	Jenis Bimbingan	Catatan Perbaikan
1	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 6 november 2023, diadakan bimbingan untuk judul dan konsep proposa penelitian	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
2	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 13 November, diadakan bimbingan untuk pengerjaan bab 1 latar belakang, tujuan, dan manfaat penelitian	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
3	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 20 November 2023, diadakan bimbingan untuk pengerjaan bab 2 tinjauan pustaka dan revisi kerangka berpikir di bab 1	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
4	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 27 November 2023, diadakan revisi untuk penambahan tinjauan pustaka di bab 2, dilanjutkan untuk bimbingan bab 3	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
5	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 4 desember 2023, dilakukan revisi pada bagian tahap penelitian di bab 3	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
6	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 11 Desember 2023, penandatanganan surat pengajuan seminar proposal	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
7	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 18 desember 2023, diadakan bimbingan pengerjaan bab 4 dan memperbaiki metode algoritma yang digunakan	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
8	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 26 desember 2023, diadakan revisi pada bagian pengumpulan data di bab 4	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
9	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 2 januari 2024, dilakukan bimbingan terkait bab4 dan dilakukan penembahan manfaat di bab 1	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
10	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 2 januari 2024, dilakukan bimbingan terkait bab4 dan dilakukan penembahan manfaat di bab 1	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
11	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 8 januari 2024, dilakukan bimbingan terkait revisi yang ada di bab 4	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
12	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 15 januari 2024, dilakukan bimbingan terkait penluisan peneltian	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
13	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 22 januari 2024, dilakukan bimbingan terakhir dan memastikan tidak ada kesalahan dalam penelitian	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	
14	5709 - MUNAWAR , S.TP, MM, Ph.D.	pada tanggal 12 februari dilakukan penandatanganan tugas akhir dan bimbingan terkait sidang tugas akhir	27 Feb 2024	Skripsi/Tesis/BusinessPlan Proposal	

Lampiran 8 – Lembar Pengajuan Sidang



UNIVERSITAS ESA UNGGUL
FAKULTAS ILMU KOMPUTER
Jl. Arjuna Utara No. 9, Tol Tomang, Kebon Jeruk, Jakarta Barat 11470

FORM PENGAJUAN SIDANG
MAGANG / SEMINAR PROPOSAL/ SKRIPSI/ TUGAS AKHIR

Nama	:	MUHAMAD AKBAR RAMADHAN
NIM	:	20190801425
Program Studi	:	Teknik Informatika / ~~Sistem Informasi~~*
Judul	:	ANALISIS SENTIMEN KAUM HOMOSEKSUAL PADA MEDIA SOSIAL X (TWITTER) MENGGUNAKAN METODE KLASIFIKASI NAÏVE BAYES CLASSIFIER (NBC), SUPPORT VECTOR MACHINE (SVM), DAN DECISION TREE
Periode	:	Ganjil / ~~Genap~~* (Tahun Akademik 2023-2024)
Kategori	:	~~Sidang Magang~~ / ~~Seminar Proposal~~ / Sidang Skripsi *

**coret yang tidak perlu*

Jakarta, 29 Januari 2024

Menyetujui,

Pembimbing

[illegible]

(Ir. Munawar, MMSI, M.Com, Ph.D)

Mengetahui,

Koordinator Tugas Akhir

[illegible]

(MUHAMAD BAHRUL ULUM, S.kom, M.kom)