**Modul 8 Praktikum Natural Language Processing**

**Sentiment Analysist Statistical Base**

**Implementing Sentiment Analysis**

Salah satu aplikasi industri NLP yang populer adalah Analisis Sentimen. Hal ini sangat penting dari sudut pandang bisnis untuk memahami bagaimana umpan balik, baik dari pelanggan produk, maupun opini publik.

**Problem**

Bagaimana Mengimplementasikan Sentimen Analisis

**Solution**

Mengimplementasikan library umum dan data emosi eksternal untuk membangun Analisis Sentimen

**How It Works**

ikuti langkah-langkah di bagian ini untuk menerapkan analisis sentimen pada kasus aplikasi android

1. Memahami mendefinisikan masalah bisnis

2. Mengidentifikasi sumber, pengumpulan, dan pemahaman data yang potensial

3. Implementasi kedalam program

Mount Google Drive

from google.colab import drive

drive.mount('/content/drive')

Import library

#import modul

import pandas as pd

import numpy as np

import re

import re as reg

import matplotlib.pyplot as plt

%matplotlib inline

Input Data Hotel

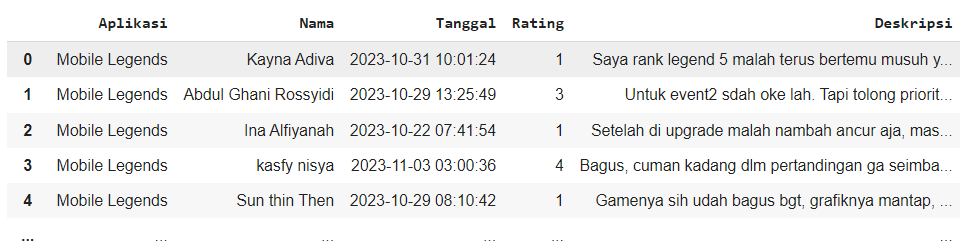
#input data hotel

dataset = pd.read\_csv('dataku\_lite.csv, sheet\_name='Sheet1')

#output data hotel

dataset

Output:



Preprocessing:

pip install sastrawi

Bersihkan kata

#Preprocessing

from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory

from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

slangs={'yg':'yang', 'tdk':'tidak', 'pd':'pada', 'mlh':'malah', 'jgn':'jangan', 'jg':'juga', 'tp':'tapi', 'blkg': 'belakang', 'dr':'dari', 'klo':'kalo', 'lg':'lagi'}

processed\_comments = []

for sentence in dataset['deskripsi']:

    # Remove all the special characters

    processed\_comment = re.sub(r'\W', ' ', str(sentence))

    # Converting to Lowercase

    processed\_comment = processed\_comment.lower()

    #Remove number

    processed\_comment = re.sub(r'\d+', ' ', processed\_comment)

    # remove all single characters

    processed\_comment = re.sub(r'\s+[a-zA-Z]\s+', ' ', processed\_comment)

    #remove duplicate character

    pattern=reg.compile(r"(.)\1{1,}",reg.DOTALL)

    processed\_comment=pattern.sub(r"\1",processed\_comment)

    #Corrected Slang words

    words = processed\_comment.split()

    rfrm=[slangs[word] if word in slangs else word for word in words]

    processed\_comment= " ".join(rfrm)

    #remove stopword

    factory = StopWordRemoverFactory()

    more\_stopword = ['tak', 'jd', 'per', 'nya'] #menambahkan stopword

    stopwords = factory.get\_stop\_words() + more\_stopword

    temp = [t for t in re.findall(r'\b[a-z]+-?[a-z]+\b',processed\_comment) if t not in stopwords]

    processed\_comment = ' '.join(temp)

    #stemming

    stemmer = StemmerFactory().create\_stemmer()

    processed\_comment = stemmer.stem(processed\_comment)

    # Substituting multiple spaces with single space

    processed\_comment = re.sub(r'\s+', ' ', processed\_comment, flags=re.I)

    processed\_comments.append(processed\_comment)

#output data Preprocessing

processed\_comments

Output:

A text on a screen

Description automatically generated

Export Data Bersih Kedalam Excell

pip install xlsxwriter

#SAVE HASIL PREPROCESSING

import xlsxwriter

workbook = xlsxwriter.Workbook('hasilpreprocessing.xlsx', {'nan\_inf\_to\_errors': True})

worksheet=workbook.add\_worksheet()

row=0

col=0

x=dataset

hasilakhir=list(zip(x.aplikasi, x.nama,x.tanggal,x.rating,x.deskripsi,processed\_comments))

worksheet.write(row, col, "aplikasi")

worksheet.write(row, col+1, "nama")

worksheet.write(row, col+2, "tanggal")

worksheet.write(row, col+3, "rating")

worksheet.write(row, col+4, "deskripsi")

worksheet.write(row, col+5, "CleanReview")

row+=1

for a,b,c,d,e,f in (hasilakhir):

    worksheet.write(row, col, a)

    worksheet.write(row, col+1, b)

    worksheet.write(row, col+2, c)

    worksheet.write(row, col+3, d)

    worksheet.write(row, col+4, e)

    worksheet.write(row, col+5, f)

    row+=1

workbook.close()

Jika Preprocessing Sudah Ada, Maka Proses Diawali Dari Sini

dataset = pd.read\_excel('hasilpreprocessing.xlsx', sheet\_name='Sheet1')

dataset['Number\_of\_words'] = dataset['CleanReview'].apply(lambda x:len(str(x).split()))

dataset.drop(dataset[dataset["Number\_of\_words"]<2].index, inplace = True)

Load Dictionary

dictionary=pd.read\_excel('NRC.xlsx', sheet\_name='dict')

#Load to memory

positive=[]

negative=[]

anger=[]

anticipation=[]

disgust=[]

fear=[]

joy=[]

sadness=[]

surprise=[]

trust=[]

teksbaru=''

for i in range(1,len(dictionary)):

    kata=dictionary.iloc[i,1]

    #print(kata)

    if (dictionary.iloc[i,6])==1:

        positive.append(kata)

    if (dictionary.iloc[i,7])==1:

        negative.append(kata)

    if (dictionary.iloc[i,8])==1:

        anger.append(kata)

    if (dictionary.iloc[i,9])==1:

        anticipation.append(kata)

    if (dictionary.iloc[i,10])==1:

        disgust.append(kata)

    if (dictionary.iloc[i,11])==1:

        fear.append(kata)

    if (dictionary.iloc[i,12])==1:

        joy.append(kata)

    if (dictionary.iloc[i,13])==1:

        sadness.append(kata)

    if (dictionary.iloc[i,14])==1:

        surprise.append(kata)

    if (dictionary.iloc[i,15])==1:

        trust.append(kata)

Lakukan Ekstraksi Fitur

#Preprocessing

processed\_comments=dataset['CleanReview']

print('Panjang :',len(processed\_comments))

datasetangkapos=[]

for kalimat in processed\_comments:

    baris=[]

    baris.append(kalimat)

    #ekstraksi fitur

    f\_sentimenpositif=0

    f\_sentimennegatif=0

    f\_anger=0

    f\_fear=0

    f\_disgust=0

    f\_sadness=0

    f\_surprise=0

    f\_joy=0

    f\_trust=0

    y=kalimat

    # Untuk setiap kata (teks) dalam kalimat, dilakukan iterasi untuk mencocokkan dengan kata-kata dalam set emosi dan sentimen positif/negatif.

    for teks in y.split():

        # Iterasi melalui indeks kata-kata dalam kamus kata positif (positive)

        for j in range(0,len(positive)-1):

            # Jika kata dalam kalimat sama dengan kata positif

            if teks==positive[j]:

                # (frekuensi sentimen positif) akan ditambah 1.

                f\_sentimenpositif+=1

        for j in range(0,len(negative)-1):

            if teks==negative[j]:

                f\_sentimennegatif+=1

        for j in range(0,len(anger)-1):

            if teks==anger[j]:

                f\_anger+=1

        for j in range(0,len(fear)-1):

            if teks==fear[j]:

                f\_fear+=1

        for j in range(0,len(disgust)-1):

            if teks==disgust[j]:

                f\_disgust+=1

        for j in range(0,len(sadness)-1):

            if teks==sadness[j]:

                f\_sadness+=1

        for j in range(0,len(joy)-1):

            if teks==joy[j]:

                f\_joy+=1

        for j in range(0,len(surprise)-1):

            if teks==surprise[j]:

                f\_surprise+=1

        for j in range(0,len(trust)-1):

            if teks==trust[j]:

                f\_trust+=1

    # Menghitung skor sentimen dengan mengurangkan frekuensi kata-kata negatif dari kata-kata positif.

    # Hasilnya, l\_sentimen akan menjadi indikator sentimen keseluruhan dari kalimat.

    l\_sentimen=f\_sentimenpositif-f\_sentimennegatif

    # mencari nilai max dari kelas emosi

    totalemotion=max([f\_anger,f\_fear,f\_disgust,f\_sadness,f\_surprise,f\_joy,f\_trust])

    # deklarasi variabel untuk menyimpan nilai sentimen

    l\_sentimenne=0 # Indikator sentimen negatif.

    l\_sentimenp=0 # Indikator sentimen positif.

    l\_sentimenn=0 # Indikator sentimen netral.

    # Jika skor sentimen positif dan negatif sama, maka kalimat dianggap netral.

    if f\_sentimenpositif-f\_sentimennegatif==0:

        # Skor sentimen negatif diatur menjadi 0 (netral).

        l\_sentimenne=0

    # Jika skor sentimen positif dan negatif tidak sama

    else:

        # Jika skor sentimen positif lebih besar dari skor sentimen negatif, maka kalimat dianggap positif.

        if f\_sentimenpositif-f\_sentimennegatif>0:

            # Skor sentimen positif diatur menjadi 1.

            l\_sentimenp=1

        # Jika skor sentimen positif lebih kecil dari skor sentimen negatif, maka kalimat dianggap negatif.

        else:

            # Skor sentimen negatif diatur menjadi -1.

            l\_sentimenn=-1

    if totalemotion==0:

        totalemotion=1;

    l\_anger=f\_anger/totalemotion

    l\_disgust=f\_disgust/totalemotion

    l\_fear=f\_fear/totalemotion

    l\_sadness=f\_sadness/totalemotion

    l\_surprise=f\_surprise/totalemotion

    l\_joy=f\_joy/totalemotion

    l\_trust=f\_trust/totalemotion

    baris.append(l\_sentimenp)                            #dataset(3)

    baris.append(l\_sentimenn)                            #dataset(3)

    baris.append(l\_anger)                            #dataset(4)

    baris.append(l\_fear)                            #dataset(5)

    baris.append(l\_disgust)                            #dataset(6)

    baris.append(l\_sadness)                            #dataset(7)

    baris.append(l\_surprise)                            #dataset(8)

    baris.append(l\_joy)                            #dataset(9)

    baris.append(l\_trust)                            #dataset(10)

    datasetangkapos.append(baris)

Buat Dataframe Untuk Menyimpan Nilai

datasetanotated=pd.DataFrame(datasetangkapos, columns=['Review','Positif','Negatif','Anger','Fear','Disgust','Sadness','Surprise','Joy','Trust'])

datasetanotated['aplikasi']=dataset['aplikasi']

datasetanotated

Output:

A screenshot of a computer

Description automatically generated

Rename nama kolom agar bisa diakses

datasetfitur = datasetanotated.drop('Review', 1)

grouped\_app = dataset\_fitur.rename(columns={'Mobile Legends': 'MobileLegends'})

grouped\_app = grouped\_app.rename(columns={'Arena of Valor': 'ArenaOfValor'})

Buang Kolom Review dan lakukan groping berdasarkan kolomnya

datasetfitur = datasetanotated.drop('Review', 1)

grouped\_obwis = datasetfitur.groupby("aplikasi").mean().abs()

# Lakukan Transpose

grouped\_obwis=grouped\_obwis.T

Output:

A table of numbers and letters

Description automatically generated

Cek nilai unik pada total aplikasi

komenaplikasi=datasetanotated.aplikasi.unique()

komenaplikasi

Output:



Lakukan Visualisasi

Barchart

grouped\_app.MobileLegends.plot(kind='bar')

A graph of negative emotions

Description automatically generated

grouped\_app.ArenaOfValor.plot(kind='bar')

A graph of negative emotions

Description automatically generated

grouped\_app.Wildrift.plot(kind='bar')

A graph of negative emotions

Description automatically generated

grouped\_app.iloc[0:2,:].plot(kind='bar', figsize=(10,5));

plt.rcParams["figure.figsize"] = [7.50, 3.50]

plt.rcParams["figure.autolayout"] = True

d = {'Column 1': [i for i in range(10)],

      'Column 2': [i \* i for i in range(10)]}

#plt.legend(bbox\_to\_anchor=(1.0, 1.0))

plt.legend(loc='center left', bbox\_to\_anchor=(0.5, 0.8),  prop={'size': 15})

plt.ylim([0, 1])

plt.xticks(fontsize = 20, rotation=0)

plt.yticks(fontsize = 20)

plt.xlabel('Sentiment', fontsize=18)

plt.ylabel('Percentage comment x100', fontsize=18)

plt.show()

A graph of different colored bars

Description automatically generated

grouped\_app.iloc[2:6,:].plot(kind='bar', figsize=(15,5));

plt.rcParams["figure.figsize"] = [4.50, 3.50]

plt.rcParams["figure.autolayout"] = True

d = {'Column 1': [i for i in range(10)],

      'Column 2': [i \* i for i in range(10)]}

#plt.legend(bbox\_to\_anchor=(1.0, 1.0))

plt.legend(loc='center left', bbox\_to\_anchor=(0.5, 0.8),  prop={'size': 15})

plt.ylim([0, 1])

plt.xticks(fontsize = 20, rotation=0)

plt.yticks(fontsize = 20)

plt.xlabel('Emosi', fontsize=18)

plt.ylabel('Persentase jumlah komentar (x100)', fontsize=18)

plt.show()

A graph of different colored bars

Description automatically generated

grouped\_app.iloc[4:6,:].plot(kind='bar', figsize=(15,5));

plt.rcParams["figure.figsize"] = [4.50, 3.50]

plt.rcParams["figure.autolayout"] = True

d = {'Column 1': [i for i in range(10)],

      'Column 2': [i \* i for i in range(10)]}

#plt.legend(bbox\_to\_anchor=(1.0, 1.0))

plt.legend(loc='center left', bbox\_to\_anchor=(0.5, 0.8),  prop={'size': 15})

plt.ylim([0, 1])

plt.xticks(fontsize = 20, rotation=0)

plt.yticks(fontsize = 20)

plt.xlabel('Emosi', fontsize=18)

plt.ylabel('Persentase jumlah komentar (x100)', fontsize=18)

plt.show()

A graph with different colored squares

Description automatically generated

grouped\_app.iloc[6:9,:].plot(kind='bar', figsize=(15,5));

plt.rcParams["figure.figsize"] = [4.50, 3.50]

plt.rcParams["figure.autolayout"] = True

d = {'Column 1': [i for i in range(10)],

      'Column 2': [i \* i for i in range(10)]}

#plt.legend(bbox\_to\_anchor=(1.0, 1.0))

plt.legend(loc='upper left', bbox\_to\_anchor=(1, 1),  prop={'size': 15})

plt.ylim([0, 1])

plt.xticks(fontsize = 20, rotation=0)

plt.yticks(fontsize = 20)

plt.xlabel('Emosi', fontsize=18)

plt.ylabel('Persentase jumlah komentar (x100)', fontsize=18)

plt.show()

A graph of different colored bars

Description automatically generated

WordCloud

from sklearn.feature\_extraction.text import CountVectorizer,TfidfVectorizer, HashingVectorizer

def get\_top\_n\_words(corpus, n=None):

    vec = CountVectorizer().fit(corpus)

    bag\_of\_words = vec.transform(corpus)

    sum\_words = bag\_of\_words.sum(axis=0)

    words\_freq = [(word, sum\_words[0, idx]) for word, idx in vec.vocabulary\_.items()]

    words\_freq =sorted(words\_freq, key = lambda x: x[1], reverse=True)

    return words\_freq[:n]

Mobile Legends

A close up of words

Description automatically generated

Wild Rift

A close up of words

Description automatically generated

AoV

A close up of words

Description automatically generated

Referensi:

<https://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm>