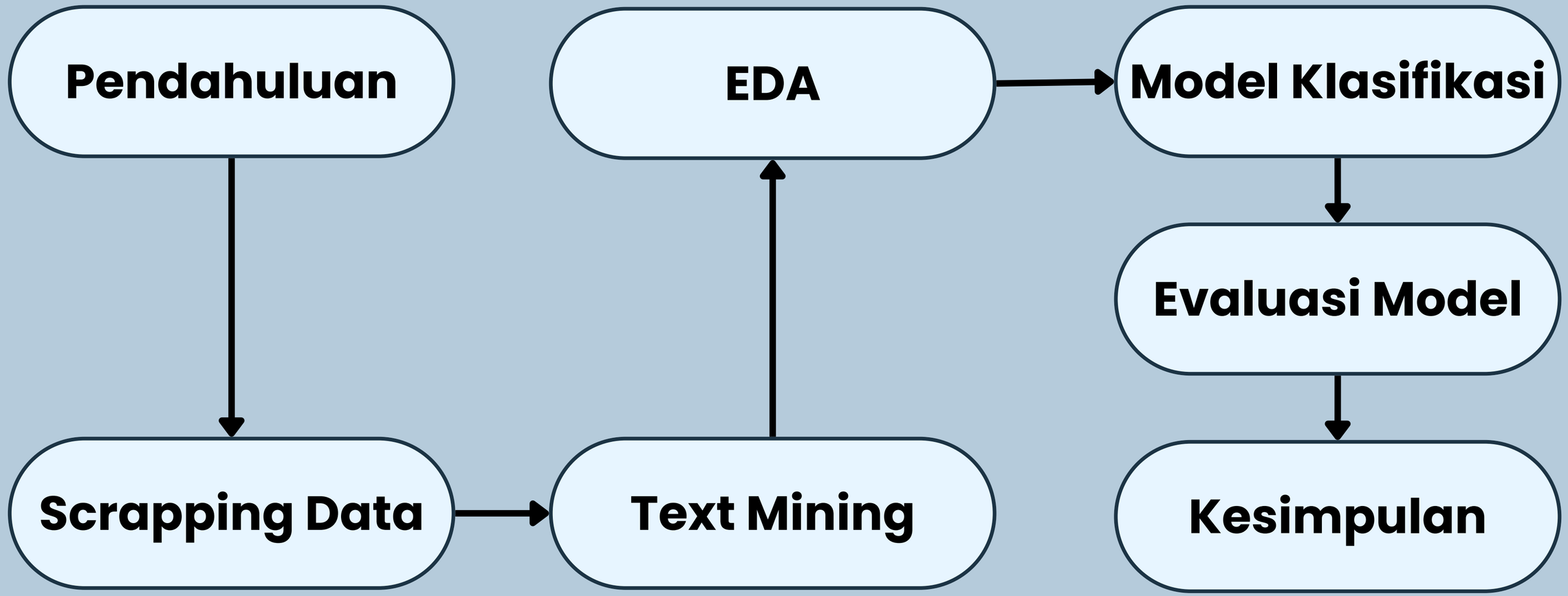
Bagus Tri Handoko

ANALISIS SENTIMEN TERHADAP REVIEW APLIKASI MOODLE DI GOOGLE PLAY STORE MENGGUNAKAN KLASIFIKASI NAIVE BAYES

TABLE OF CONTENT



PENDAHULUAN

E-learning, merupakan sebuah proses belajar dan mengajar, yang memanfaatkan media elektronik, secara khusus yaitu internet, sebagai sistem pembelajarannya. Secara umum, e-learning adalah sebuah proses pembelajaran berbasis elektronik. Salah satu platform E-learning yang populer adalah Moodle yang banyak di gunakan oleh instansi pendidikan.

Pada penelitian kali ini kelompok kami akan melakukan analisis sentimen terhadap review aplikasi Moodle menggunakan klasifikasi Multinomial Naive Bayes, tujuan penelitian ini untuk mengklasifikasi komentar positif dan negatif dari review pengguna Moodle.

SCRAPPING DATA

```
from google_play_scraper import app

import pandas as pd

import numpy as np
```

Untuk Scrapping data review aplikasi Moodle kami menggunakan library google_play_scraper.

```
#scrape jumlah ulasan yang diinginkan
from google_play_scraper import Sort, reviews

result, continuation_token = reviews(
    'com.moodle.moodlemobile',
    lang='id',
    country='id',
    sort=Sort.MOST_RELEVANT,
    count=2500,
    filter_score_with=None
)
```

Ada beberapa parameter yang digunakan yaitu :

- 'com.moodle.moodlemobile', adalah ID dari aplikasi moodle di play store
- lang = “id”, mengambil ulasan yang berbahasa Indonesia
- country “id”, menentukan negara asal ulasan yakni Indonesia
- sort=sort.MOST_RELEVANT, pengurutan ulasan berdasarkan relevansi
- count=2500, mengambil jumlah data ulasan sebanyak 2500 ulasan
- filter_score_with=None, tidak ada penyaringan score, sehingga semua ulasan diambil

SCRAPPING DATA

	reviewId	userName	userImage	content	score	thumbsUpCount	reviewCreatedVersion	at	replyContent	repliedAt	appVersion
0	2420f8ea-42c5-4ce3-8e56-b7a759caf72c	Widi Saputra	https://play-lh.googleusercontent.com/a-/ALV-U...	Gatau ni aplikasi gajelas gini, udah kirim sat...	1	2	4.2.0	2023-10-23 08:09:18	None	NaT	4.2.0
1	bc740079-4ba9-44fa-a895-567c922ba9c9	Kunthi	https://play-lh.googleusercontent.com/a/ACg8oc...	Tadinya bagus sih pas awal2. Simpel, cepet, mu...	2	111	3.9.5	2021-10-03 17:30:00	None	NaT	3.9.5
2	6cb6ab81-fd31-4988-9d82-a265931d5ace	Franciskus Damianus	https://play-lh.googleusercontent.com/a-/ALV-U...	Awalnya aplikasi ini sangat bagus, tapi setela...	1	11	3.9.5	2022-04-18 11:55:03	None	NaT	3.9.5
3	64a1e82f-36ec-4210-b28d-f24248fa4ec0	Fitrie Hasibuan	https://play-lh.googleusercontent.com/a-/ALV-U...	Entahlah kenapa aplikasi ini tidak sebagus dul...	2	10	4.2.0	2023-06-13 01:17:40	None	NaT	4.2.0
4	5a48d8d3-c6a8-453e-93a9-4069f2731997	Jamiro Noor Z.	https://play-lh.googleusercontent.com/a-/ALV-U...	Aplikasi jelek kurang cepet bahkan loading gan...	1	3	None	2020-06-08 01:43:23	None	NaT	None

IMPORT LIBRARY DAN DATASET

Import Library

```
import string
from sklearn.pipeline import Pipeline
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
```

executed in 2.01s, finished 06:51:30 2023-11-24

Import Dataset

```
df_awal = pd.read_csv("review_modle_awal.csv", sep=',', encoding='latin1')
df_awal.head(15)
```

executed in 50ms, finished 07:36:58 2023-11-24

TEXT MINING

Text PreProcessing

Ada beberapa tahap text preprocessing yang digunakan yakni: pemilihan fitur atau kolom, case folding, word normalization tokenization, stopword removal, stemming, dan labeling

Pembobotan TF IDF

Pembobotan pada setiap kata menggunakan TF IDF. TF IDF merupakan suatu teknik atau cara pemberian bobot pada suatu kata terhadap dokumen.

TEXT PREPROCESSING

Pemilihan Fitur / Kolom

```
In [29]: #membuang kolom / fitur yang akan di gunakan nantinya dan merename kolom
         df = df_awal[["content", "score"]].rename(columns={'score': 'rating', 'content': 'komen'})
         df.head()
```

executed in 12ms, finished 21:39:50 2023-11-19

Memilih kolom yang hanya digunakan, untuk mengolah data yakni kolom content dan score, lalu mengubah nama kolom score menjadi rating dan content menjadi komen

	komen	rating
0	Cukup membantu dalam belajar namun kekurangannya agak lemot	4
1	Sejauh ini bagus si cm kekuranganya pas upload foto jdi ngeblur dan jelek, jdnya pas saya ngirim tugas pas ke upload tulisan jdi burem dan jelek bgt bnrn tolong dong perbaiki kualitas foto yg di upload	3
2	Gatau ni aplikasi gajelas gini, udah kirim satu file tauÂ² jadi 6 file kan bingung tu, udah gitu jadi nilainya kurang gara2 ngeleg2 gitu, sumpah ga rekomend banget	1
3	Tadinya bagus sih pas awal2. Simpel, cepet, mudah dipahami juga dashboardnya. Tadinya mau ku kasih bintang 5, tapi udah hampir 3 Minggu ini malah bermasalah. Dipakek kirim tugas lemooottt bgt, pakai WiFi aja bisa sampe 10 menit. Padahal cuma 2 file tugasn dan jumlahnya cuma 3MB. Ga bisa dipakek absen juga, kayak ada pilihan absensinya tapi ga bisa dipilih dan disubmit. Mohon diperbaiki, kalau membaik saya tambah bintangnya ð□□□ð□□□ð□□□	2
4	Awalnya aplikasi ini sangat bagus, tapi setelah update jadi sagat jelek, pertama karena aplikasinya lemot kalau scroll (sering ada tulisan "aplikasi tidak menanggapi") atau download file, kedua tampilannya jadi tidak enak dilihat karena berantakan	1

TEXT PREPROCESSING

Case Folding dan Pembersihan Karakter

In [44]:
```python
# Case folding
import re

def casefolding(komen):
    # Konversi ke lowercase
    komen = komen.lower()

    # Hapus angka
    komen = re.sub(r'\d+', '', komen)

    # Hapus emoji
    komen = re.sub(r'[\U00010000-\U0010ffff]', '', komen)

    # Hapus karakter-karakter tertentu
    komen = komen.strip(" ")
    komen = re.sub(r'[?|$|.|!²_:")(-+,]', '', komen)
    komen = re.sub(r'[^\w\s]','',komen) # menghapus karakter tanda baca

    return komen

# Terapkan fungsi pada kolom 'komen'
df['case_folding'] = df['komen'].apply(casefolding)
df.head()
```

executed in 31ms, finished 22:52:58 2023-11-23

In [45]:
```python
print('Kata awal\t\t:',df['komen'].iloc[13])
print('Casefolding\t\t:',df['case_folding'].iloc[13])
```

executed in 6ms, finished 22:53:04 2023-11-23

```
Kata awal               : Sangat membantu dalam belajar jarak jauh, bisa diakses dimana saja.
Casefolding             : sangat membantu dalam belajar jarak jauh bisa diakses dimana saja
```

Case folding adalah mengubah seluruh huruf yang ada pada dokumen menjadi lower case semua ataupun upper case semua.

Pembersihan karakter pada dokumen ini mencangkup penghapusan angka, tanda baca, emoji, menghapus karakter dan simbol simbol yang tidak diperlukan dalam pemrosesan nantinya

TEXT PREPROCESSING

Word Normalization

```
In [46]: #word normalization
key_norm = pd.read_csv('key_norm.csv')

def text_normalize(komen):
    words = komen.split()
    normalized_text = ' '.join([key_norm[key_norm['singkat'] == word]['hasil'].values[0] if (key_norm['singkat'] == word).any() e
    normalized_text = str.lower(normalized_text)
    return normalized_text

df['word_normal'] = df['case_folding'].apply(text_normalize)
df.head()
```

executed in 6.61s, finished 22:53:17 2023-11-23

Word Normalization yaitu proses mengubah kata tidak baku atau non standar menjadi kata baku dari kumpulan *slangwords* yang tersedia.

```
In [47]: print('Kata awal\t\t:',df['komen'].iloc[23])
print('word normalize\t:',df['word_normal'].iloc[23])
```

executed in 7ms, finished 22:53:22 2023-11-23

```
Kata awal               : Bisa dengan mudah dan cepat serta tidak sering gangguan siknal alias krowdit or Lowbet
word normalize          : bisa dengan mudah dan cepat serta tidak sering gangguan sinyal atau ramai atau boros
```

Source kamus *Slangwords*
https://github.com/ksnugroho/klasifikasi-spam-sms/blob/master/data/key_norm.csv

TEXT PREPROCESSING

Tokenization

In [48]:
```
# Tokenization
def token(komen):
    nstr = komen.split(' ')
    dat = []
    a = -1
    for hu in nstr:
        a = a + 1
        if hu =='':
            dat.append(a)
    p = 0
    b = 0
    for q in dat:
        b = q - p
        del nstr[b]
        p = p + 1
    return nstr
df['tokenize'] = df['word_normal'].apply(token)
df.head()
```

executed in 24ms, finished 22:53:31 2023-11-23

Tokenization dalah membagi teks menjadi token-token atau bagian-bagian untuk memecah kalimat menjadi kepingan kata, contoh kalimat “aplikasinya bagus sekali”, menjadi “aplikasinya”,”bagus”,”sekali”.

In [49]:
```
print('Kata awal\t\t:',df['komen'].iloc[13])
print('Casefolding\t\t:',df['case_folding'].iloc[13])
print('Tokenize\t\t:',df['tokenize'].iloc[13])
```

executed in 6ms, finished 22:53:35 2023-11-23

```
Kata awal               : Sangat membantu dalam belajar jarak jauh, bisa diakses dimana saja.
Casefolding             : sangat membantu dalam belajar jarak jauh bisa diakses dimana saja
Tokenize                : ['sangat', 'membantu', 'dalam', 'belajar', 'jarak', 'jauh', 'bisa', 'diakses', 'dimana', 'saja']
```

TEXT PREPROCESSING

Stop-word Removal

```
In [50]: from nltk.corpus import stopwords

more_stopwords = ['aaaaaahhhhh', 'aah', 'aakkkk', 'woiii', 'abt', 'acong', 'ah','ahh','ahhh','ahhhh','aiti',
                  'aj','ajaaaa','acess','activationaktivasi','afk','akskbxjsnd','am','ampass','ampe','ampee','anjeeeengg',
                 'gg','aplikadi','jdi','appcmn','aplilasi','aplilasi','apayg','anjghapus','jd','yg','bamgettttt','bagustpi',
                  'bagusspngej','bagis','bagaimanana','aupport','appcmn','aplikasj','bangbang','bangkrruut','bamgettttt',
                  'centangnyaaintinnya','byzulpahri','bwangat','bngtttttttolong','bngetkalo','ckuakz','deddytolloll','danbagak',
                 'cuuuuuuttttitu','cuihh','cokkk','cok','coeg','ckuakz','dhlh','dg','deddytolloll','dijalankanjika','digerakkanma
                 'dhlh']

def stopword_removal(komen):
    filtering = set(stopwords.words('indonesian', 'english') + more_stopwords)
    df = [x for x in komen if x not in filtering]
    return df

df['stop_word'] = df['tokenize'].apply(stopword_removal)
df.head()
```

executed in 1.01s, finished 22:53:39 2023-11-23

```
In [51]: print('Kata awal\t\t:',df['komen'].iloc[13])
         print('Casefolding\t\t:',df['case_folding'].iloc[13])
         print('Tokenize\t\t:',df['tokenize'].iloc[13])
         print('stopword\t\t:',df['stop_word'].iloc[13])
```

executed in 8ms, finished 22:53:44 2023-11-23

```
Kata awal               : Sangat membantu dalam belajar jarak jauh, bisa diakses dimana saja.
Casefolding             : sangat membantu dalam belajar jarak jauh bisa diakses dimana saja
Tokenize                : ['sangat', 'membantu', 'dalam', 'belajar', 'jarak', 'jauh', 'bisa', 'diakses', 'dimana', 'saja']
stopword                : ['membantu', 'belajar', 'jarak', 'diakses', 'dimana']
```

Stop-word removal adalah penghapusan kata yang tidak bermakna, seperti kata penghubung dan, yang, bisa, dalam , dan lain-lain.

TEXT PREPROCESSING

Stemming

In [54]:
```python
# Stemming
from sklearn.pipeline import Pipeline
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

def stemming(komen):
    factory = StemmerFactory()
    stemmer = factory.create_stemmer()
    do = []
    for w in komen:
        dt = stemmer.stem(w)
        do.append(dt)
    d_clean = []
    d_clean =" ".join(do)
    print(d_clean)
    return d_clean
df['stemming'] = df['stop_word'].apply(stemming)
```

executed in 10m 46s, finished 23:05:29 2023-11-23

Stemming adalah proses yang digunakan dalam text mining untuk mengubah kata-kata dalam dokumen menjadi bentuk dasarnya. Seperti menghapus imbuhan yang ada dalam sebuah kata.

In [56]:
```python
print('Kata awal\t\t:',df['komen'].iloc[13])
print('Casefolding\t\t:',df['case_folding'].iloc[13])
print('Tokenize\t\t:',df['tokenize'].iloc[13])
print('stopword\t\t:',df['stop_word'].iloc[13])
print('Stemming\t\t:',df['stemming'].iloc[13])
```

executed in 8ms, finished 23:09:03 2023-11-23

```
Kata awal               : Sangat membantu dalam belajar jarak jauh, bisa diakses dimana saja.
Casefolding             : sangat membantu dalam belajar jarak jauh bisa diakses dimana saja
Tokenize                : ['sangat', 'membantu', 'dalam', 'belajar', 'jarak', 'jauh', 'bisa', 'diakses', 'dimana', 'saja']
stopword                : ['membantu', 'belajar', 'jarak', 'diakses', 'dimana']
Stemming                : bantu ajar jarak akses mana
```

TEXT PREPROCESSING

Labeling

```
# Memberikan pelabelan pada rating(1-3 = Negatif, 4-5 = Positif)
df_clear['label'] = df['rating'].map({1:'NEGATIF', 2:'NEGATIF', 3:'NEGATIF', 4:'POSITIF',5:'POSITIF'})
df_clear = df_clear.astype({'label' : 'category'})
df_clear = df_clear.astype({'komen' : 'string'})
df_clear.head()
```

executed in 22ms, finished 19:31:59 2023-11-20

Melakukan labeling untuk menjadi variable pengawas, parameter dalam pemberian label adalah dari jumlah rating, apabila ratingnya 1 - 3 maka diberi label negatif dan jika ratingnya 4 - 5 maka diberi label positif.

	komen	rating	label
0	bantu ajar kurang lot	4	POSITIF
1	bagus si cm kekuranganya pas upload foto jdi ngeblur jelek jdnya pas ngirim tugas pas upload tulis jdi burem jelek bgt bnrn tolong baik kualitas foto yg upload	3	NEGATIF
2	gatau ni aplikasi gajelas gin udah kirim file tau file bingung tu udah gitu nilai gara ngeleg gitu sumpah ga rekomend banget	1	NEGATIF
3	bagus sih pas simpel cepet mudah paham dashboardnya ku kasih bintang udah minggu masalah dipakek kirim tugas lemooottt bgt pakai wifi aja sampe menit file tugasn mb ga dipakek absen kayak pilih absensi ga pilih disubmit mohon baik baik bintang	2	NEGATIF
4	aplikasi bagus update sagat jelek aplikasi lot scroll tulis aplikasi tanggap download file tampil enak beranta	1	NEGATIF

TEXT PREPROCESSING

Pembobotan TF - IDF

```
In [12]: #Proses TF - IDF
         from sklearn.feature_extraction.text import CountVectorizer
         from sklearn.feature_extraction.text import TfidfVectorizer

         tf = TfidfVectorizer()
         text_tf = tf.fit_transform(df_clear['komen'].astype('U'))
         text_tf
```

executed in 65ms, finished 15:35:45 2023-11-21

```
Out[12]: <2500x3402 sparse matrix of type '<class 'numpy.float64'>'
                with 15794 stored elements in Compressed Sparse Row format>
```

TF-IDF (Term Frequency-Inverse Document Frequency) adalah suatu metode yang digunakan dalam pemrosesan teks dan pengelompokan dokumen. Fungsi utamanya adalah untuk memberikan bobot (weight) pada setiap kata dalam suatu dokumen, yang mencerminkan seberapa penting kata tersebut

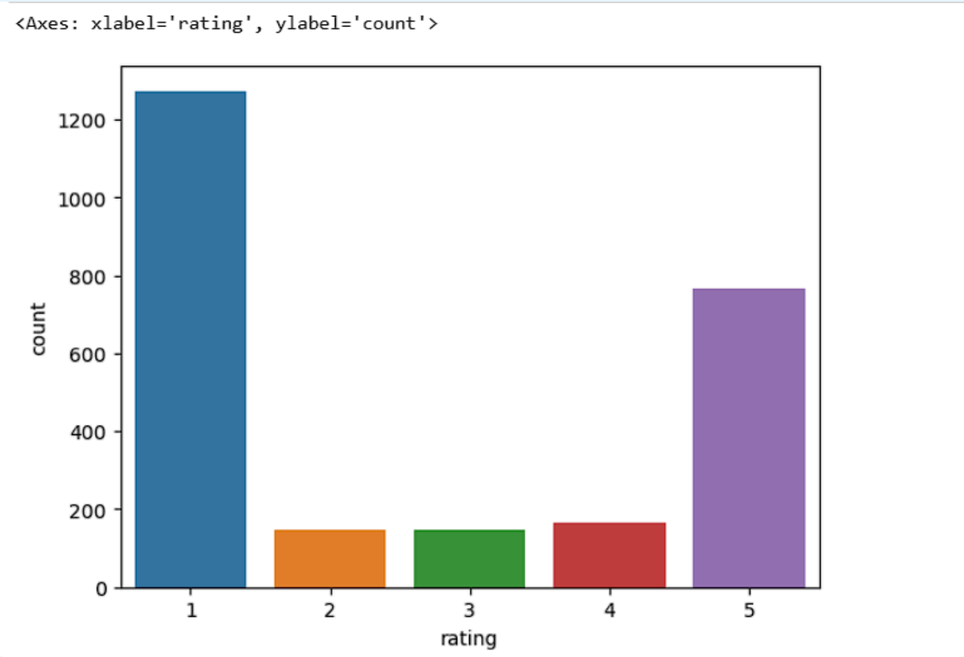
	nilai	kata
160	142.541242	aplikasi
235	119.201993	bagus
309	100.486408	bantu
2121	72.890950	nya
150	70.074290	apk
31	56.620970	ajar
2960	51.572843	tugas
283	50.884747	banget
948	50.824701	gak
259	50.006443	baik

EXPLORATORY DATA ANALYSIS (EDA)

```
# EDA (Expolatory Data Analysis)
import seaborn as sns
import matplotlib.pyplot as plt

sns.countplot(x='rating', data=df_clear)
```

executed in 1.60s, finished 15:35:38 2023-11-21



Melihat jumlah pada setiap rating menggunakan barplot , yang dimana :

- rating 1 = 1273
- rating 2 = 147
- rating 3 = 148
- rating 4 = 166
- rating 5 = 766

EXPLORATORY DATA ANALYSIS (EDA)

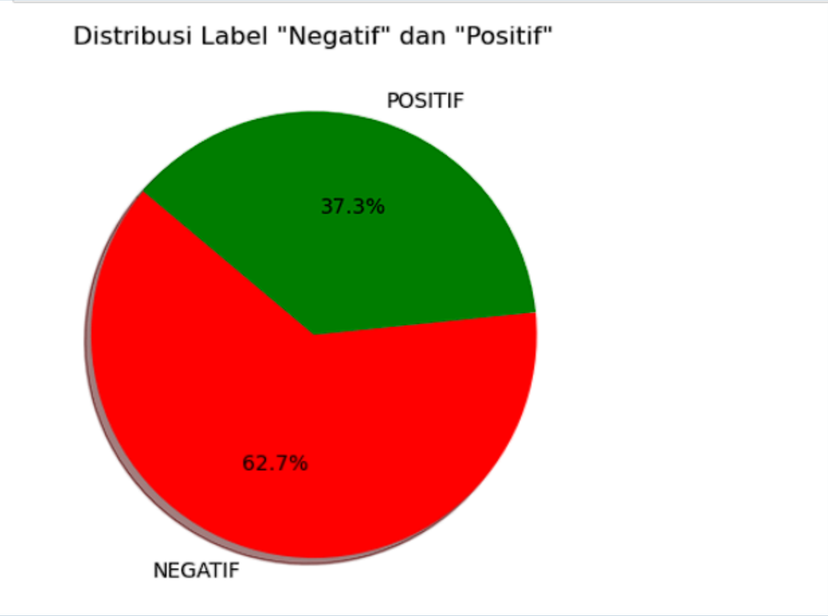
```
labels = df_clear['label'].value_counts().index
sizes = df_clear['label'].value_counts().values

# Membuat pie chart
plt.pie(sizes, labels=labels, colors=['red','green'], autopct='%1.1f%%', startangle=140, shadow= True)

# Menambahkan judul
plt.title('Distribusi Label "Negatif" dan "Positif"')

# Menampilkan pie chart
plt.show()
```

executed in 160ms, finished 15:35:41 2023-11-21



Membuat pie chart untuk melihat jumlah data komentar yang berlabel negatif dan positif.

MODEL KLASIFIKASI

Split data 80 : 20

```
In [75]: #Split data 80:20
         from sklearn.model_selection import train_test_split

         X_train, X_test, y_train, y_test = train_test_split(text_tf, df_clear['label'], test_size=0.2, random_state=40)
```

executed in 20ms, finished 00:09:29 2023-11-24

Melakukan splitting data menggunakan library train_test_split dengan perbandingan 80% untuk data training dan 20% untuk data testing dan random_state = 40.

```
In [77]: # Model klasifikasi dan evaluasi model
         from sklearn.naive_bayes import MultinomialNB
         from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
         from sklearn.metrics import classification_report
         from sklearn.metrics import confusion_matrix


         clf = MultinomialNB().fit(X_train, y_train)
         predic = clf.predict(X_test)
         predic
```

executed in 31ms, finished 00:11:23 2023-11-24

Membuat model klasifikasi dengan algoritma Multinominal Naive Bayes, dan membuat model untuk data training dan data testing. Dimana training model dengan variabel X_train dan y_train disimpan pada variable clf. Lalu data training digunakan untuk memprediksi label pada data testing.

MODEL KLASIFIKASI

Multinominal Naive Bayes

```
predic
```

executed in 31ms, finished 00:11:23 2023-11-24

Out[77]:
```
array(['POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'NEGATIF', 'POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'NEGATIF', 'POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'POSITIF', 'POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'POSITIF', 'POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'POSITIF',
       'POSITIF', 'NEGATIF', 'POSITIF', 'NEGATIF', 'POSITIF', 'NEGATIF',
       'POSITIF', 'POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'POSITIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'POSITIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'POSITIF', 'NEGATIF', 'POSITIF', 'NEGATIF', 'POSITIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'POSITIF', 'POSITIF',
       'NEGATIF', 'NEGATIF', 'POSITIF', 'POSITIF', 'NEGATIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'POSITIF', 'POSITIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'POSITIF',
       'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'NEGATIF', 'POSITIF', 'NEGATIF', 'NEGATIF',
       'NEGATIF', 'NEGATIF', 'NEGATIF', 'POSITIF', 'POSITIF', 'NEGATIF',
```

Menampilkan hasill prediksi label dari model.

In [76]:
```
#variabel pengawas dari model kalsifikasi
y_test
```

executed in 12ms, finished 00:11:00 2023-11-24

Out[76]:
```
2383    POSITIF
1297    POSITIF
2       NEGATIF
1647    NEGATIF
1356    NEGATIF
         ...
51      POSITIF
252     NEGATIF
1352    NEGATIF
923     NEGATIF
1939    POSITIF
Name: label, Length: 500, dtype: category
Categories (2, object): ['NEGATIF', 'POSITIF']
```

Menampilkan y_test sebagai variabel pengawas atau label sebenarnya dari model klasifikasi.

EVALUASI MODEL

Evaluasi Model Multinominal Naive Bayes

```
In [14]: predik_benar = (predic == y_test).sum()
         predik_salah = (predic != y_test).sum()

         print('jumlah benar : ', predik_benar)
         print('jumlah salah : ', predik_salah)
```

executed in 9ms, finished 07:38:15 2023-11-24

```
jumlah benar :  406
jumlah salah :  94
```

```
In [15]: print("MultinomialNB akurasi score : ", accuracy_score(y_test,predic))

         print(f'confusion_matrix : \n {confusion_matrix(y_test, predic)}')
         print("============================== \n")
         print(classification_report(y_test,predic, zero_division=0))
```

executed in 49ms, finished 07:38:46 2023-11-24

```
MultinomialNB akurasi score :  0.812
confusion_matrix :
 [[294  20]
 [ 74 112]]
==============================

              precision    recall  f1-score   support

     NEGATIF       0.80      0.94      0.86       314
     POSITIF       0.85      0.60      0.70       186

    accuracy                           0.81       500
   macro avg       0.82      0.77      0.78       500
weighted avg       0.82      0.81      0.80       500
```

Hasil dari evaluasi model yang di dapatkan dari algoritma Multinominal naive bayes adalah 406 untuk hasil prediksi benar dan 94 untuk hasil prediksi yang salah, total data prediksi yaitu 500.

Berdasarkan hasil evaluasi model nilai akurasi skor dari algoritma Multinominal Naive Bayes mencapai 0.812.

Confusion matrix dan classification report juga digunakan untuk mengevaluasi model.

KESIMPULAN

Berdasarkan hasil dari Analisis Sentimen Terhadap Review Aplikasi Moodle di Google Play Store menggunakan Algoritma Klasifikasi Naive Bayes mendapatkan hasil sentimen berlabel Negatif sebanyak 62.7% dan sentimen berlabel positif sebanyak 37.3% dari 2500 total data yang diambil.

Penggunaan Algoritma klasifikasi Multinominal Naive Bayes mendapatkan hasil akurasi skor sebesar 81% yang menunjukkan nilai cukup baik dalam memprediksi sentimen negatif dan positif, dengan hasil jumlah prediksi yang benar mencapai 406 dan jumlah prediksi yang salah mencapai 94 dari 500 data yang uji