Analisis Sentimen dengan *Logistic Regression* untuk Deteksi Kata pada Livin' *by* Mandiri

Ni Made Gita Satviki Nirmala[a1], Ngurah Agus Sanjaya ER[a2]

[a]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana
Jalan Raya Kampus UNUD, Bukit Jimbaran, Kuta Selatan, Badung, Bali, Indonesia
[1]gitanirmala726@gmail.com
[2]agus_sanjaya@unud.ac.id

Abstract

Livin by Mandiri is one of the most frequently used mobile banking. To find out the quality of the application, you can carry out sentiment analysis from reviews. The data taken from the Google Play Store was 6334 data from January 2022 to December 2022. The training data and test data used had a ratio of 80:20. This data goes through preprocessing and then TF-IDF weighting is carried out. After that, the analysis used logistic regression which produced 91.5% with C = 0.75. As well as getting negative sentiment results, namely precision 89%, recall 95%, f1-score 92%. Meanwhile, positive sentiment produces 94% precision, 88% recall, 91% f1-score. There is a word detection program that can help search for keywords including positive sentiment or negative sentiment from the Livin by Mandiri application.

Keywords: *Livin by Mandiri, Logistic Regression, Mobile Banking, Sentiment Analysis, TF-IDF, Word Detection*

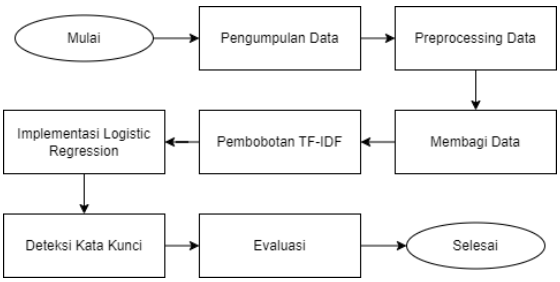
1. Pendahuluan

Bank mandiri merupakan salah satu bank terbesar di Indonesia yang sudah berdiri sejak 2 Oktober 1998 [1]. Dengan banyaknya kebutuhan nasabah dan perkembangan zaman yang begitu pesat, menyebabkan bank mandiri meluncurkan aplikasi *mobile banking* bernama Livin' by Mandiri untuk mempermudah para nasabah pada Oktober 2021 [2]. Dengan mudahnya pemakaian dari Livin' by Mandiri menyebabkan kenaikan pengunduhan pada aplikasi, diketahui hingga September 2023 telah terunduh sebanyak 32 juta kali dan pengguna aktif sebesar 21 juta pengguna [3]. Dengan banyaknya penggunaan pada aplikasi tentu saja menimbulkan komentar kritik maupun saran yang diberikan pada *google play store*. Terhitung hingga 6 Mei 2024 sudah terdapat 548 ribu ulasan yang diberikan dengan rata-rata bintang 4. Dengan banyaknya data tersebut pada penelitian ini ingin menganalisis mengenai sentimen baik positif maupun negatif pada ulasan Livin' by Mandiri. Pada penelitian ini akan menggunakan metode algoritma yang dipergunakan untuk analisis sentimen. Sebelumnya pada tahun 2022 sudah ada penelitian serupa yang diteliti oleh Muhammad Zaki Hariansyah dan Siswanto dengan jurnal berjudul "Implementasi Metode Multinomial Naïve Bayes pada Analisis Sentimen Terhadap Layanan Aplikasi Livin by Mandiri" dengan memiliki tujuan agar memperoleh umpan balik dan dapat meningkatkan kualitas layanan aplikasi dengan mengetahui keluhan secara realtime [4]. Penelitian dilakukan dengan menggunakan data cuitan twitter sebanyak 1182 data, yang kemudian dianalisa dengan algoritma naïve bayes mendapatkan skor akurasi sebesar 93%, *precision* 90%, *recall* 93%, dan *F1-score* 91% dengan rata-rata skor sebesar 92% [4]. Serta terdapat penelitian serupa lainnya pada tahun 2023 yang ditulis oleh Alfajri dkk dengan jurnal berjudul "Perbandingan Analisis Sentimen Menggunakan Naïve Bayes, *Lexicon-Based*, dan *Hybrid* pada Ulasan Pengguna Aplikasi Livin' by Mandiri" dengan tujuan membandingkan performa analisis sentimen berbasis Naive Bayes, *Lexicon-based*, dan *Hybrid* dalam mengklasifikasikan sentimen ulasan pengguna [5]. Penelitian dilakukan dengan menggunakan data ulasan *google play store* pada aplikasi Livin' by Mandiri, dengan menghasilkan metode naïve bayes memiliki akurasi tertinggi yaitu *F1-score* sebesar 0,96 [5]. Sehingga pada penelitian ini menggunakan algoritma logistic regression dalam menganalisis sentimen dengan bantuan

pembobotan TF-IDF. Bukan hanya itu pada penelitian ini juga menghasilkan program untuk dapat memeriksa kata kunci yang berhubungan dengan Livin' by Mandiri dan mengetahui apakah kata kunci tersebut memiliki sentimen positif atau sentimen negatif.

2. Metode Penelitian

Penelitian yang dilakukan melalui beberapa tahapan yang dilakukan agar mendapatkan hasil yang baik dan maksimal. Berikut merupakan gambaran dari tahapan yang dilakukan pada penelitian ini.



Gambar 1. Tahapan Penelitian

2.1 Pengumpulan Data

Dataset yang dipergunakan dalam penelitian ini merupakan ulasan pengguna aplikasi Livin' by Mandiri pada *google play store* [6]. Dataset merupakan data sekunder yang diambil dari *Kaggle* yang pada awalnya terdapat 7057 data berbahasa Indonesia. Dari data tersebut dilabeli sesuai bintang dengan bintang 4 dan 5 dilabeli 'Positif', bintang 3 dilabeli 'Netral', dan bintang 1 dan 2 dilabeli 'Negatif'. Lalu data 'Netral' dihilangkan dari dataset, sehingga hanya tersisa data berlabel 'Positif' dan 'Negatif'. Pada data tersebut dilakukan penyeimbangan data dengan perbandingan 50:50 sehingga dataset akhir yang dimiliki yaitu 6344 data. Data ulasan tersebut merupakan ulasan dari Januari 2022 hingga Desember 2022.

2.2 *Preprocessing* Data

Pada *preprocessing* data dilakukan beberapa tahapan yang dilakukan untuk membersihkan data agar sesuai dengan yang dapat dibaca dan diolah komputer. Pada penelitian ini melakukan beberapa tahapan dari memeriksa data *null* data hingga stemming.

a. Cek *NULL*
Preprocessing data yang dilakukan pertama kali yaitu memeriksa dari dataset apakah terdapat data yang *null* atau tidak. Bila memang terdapat data yang *null* maka akan terdapat proses yang dilakukan untuk menghapus data tersebut.

b. *Cleaning* Data
Pada *cleaning* data terdapat beberapa tahapan yang dilakukan untuk mendapatkan hasil yang maksimal untuk menghilangkan hal-hal yang tidak penting pada data. *Cleaning* pada bagian awal menghilangkan komentar yang berisikan *hashtag*, *gmai*l, *link* suatu *web*, menghilangkan angka, dan menghilangkan karakter khusus. *Cleaning* dilakukan dengan menghilangkan hal-hal tersebut dan digantikan dengan spasi kosong. Lalu *cleaning* kedua yaitu menghapus emoji yang terdapat pada komentar dan diganti dengan spasi. Serta yang terakhir menghilangkan kata yang berulang agar muncul hanya sekali.

c. *Casefolding*
Proses ketiga yaitu melakukan *casefolding*, yaitu tahapan yang menyebabkan komentar menjadi huruf kecil semua atau lebih sering dikenal dengan *lowercase*. Sehingga hasil akhirnya menyebabkan data tidak ada perbedaan bila kata tersebut sama dikarenakan tidak adanya kapital pada kata tertentu.

d. *Tokenizing*
Proses keempat melakukan *tokenizing* yaitu memecah komentar menjadi penggalan-penggalan kata yang lebih kecil atau dapat diartikan memenggal kata pada komentar. Pada *tokenizing* spasi tidak dihitung sebagai sebuah kata sehingga tidak ada pemenggalan untuk spasi.

e. Formalisasi
Pada formalisasi dilakukan perubahan kata-kata terhadap kata tidak baku maupun kata yang memiliki kesalahan dalam penulisan. Pada penelitian ini menggunakan *file* txt yang berisikan rangkaian kata tidak baku atau kesalahan penulisan menjadi kata yang baku dan penulisan yang benar. Pada kamus ini berisikan beberapa kata yang dicatat dan dibenahi sesuai dengan data komentar yang terdapat pada *dataset*. Dikarenakan dalam memeriksa 6339 data terlalu banyak, maka peneliti hanya memeriksa kata yang tidak baku dan kesalahan penulisan dari 2000 data, yang menurut peneliti sudah dapat mendefinisikan sebagian besar kesalahan pada data.

f. *Stopword Removal*
Bagian *stopword removal* merupakan tahapan yang dilakukan untuk menghilangkan kata yang tidak penting atau terlalu sering muncul. Contoh kata yang biasanya dihilangkan yaitu kata untuk merepresentasikan diri atau orang lain seperti “saya”, “kamu”, dan sebagainya.

g. *Stemming*
Stemming pada penelitian ini merupakan tahapan paling akhir yang dilakukan pada *preprocessing* data. Tahapan ini bertujuan untuk mengubah kata menjadi kata dasar atau menghilangkan imbuhan yang terdapat pada suatu kata. Bagian ini dapat dikatakan bagian paling penting dikarenakan mempengaruhi kata yang menjadi kategori bagi sentimen negatif maupun sentimen positif.

2.3 Pembobotan TF-IDF

Pembobotan TF-IDF atau yang memiliki kepanjangan *Term Frequency Inverse Document* merupakan salah satu metode pembobotan yang sering digunakan pada penelitian. TF-IDF menjadi alasan sering digunakan dikarenakan pemakaiannya yang mudah namun lumayan kompleks sehingga dapat meningkatkan keakuratan dari suatu algoritma yang dipergunakan. Pembobotan TF-IDF berfokus terhadap transformasi data dari data tekstual ke dalam bentuk data numerik untuk dilakukan pembobotan pada setiap kata atau fitur [7]. Bila ditelaah dari setiap katanya maka TF dapat diartikan sebagai frekuensi kemunculan kata pada setiap dokumen [7]. Sedangkan DF berarti frekuensi dokumen yang mengandung kata tersebut, IDF merupakan *inverse* dari nilai DF [7]. Adapun persamaan yang digunakan dalam TF-IDF dapat dilihat pada persamaan 1 [8].

$$W(d,t) = TF(d,t) \times \log\left(\frac{N}{df(t)}\right) \quad (1)$$

$$TF(d,t) = \frac{jumlah\ kata\ dalam\ dokumen}{total\ kata\ dalam\ dokumen} \quad (2)$$

2.4 *Logistic Regression*

Logistic regression merupakan suatu algoritma yang berhubungan dengan teori matematika yaitu probabilitas [9]. Algoritma ini memprediksi probabilitas diskrit dengan kinerja yang unggul [10]. Dalam pengaplikasian algoritma ini menggunakan fungsi *logistic* untuk menentukan probabilitas yang menghasilkan keluaran biner yaitu 0 atau 1 [10]. Pada algoritma ini menggunakan fungsi sigmoid dalam memprediksi probabilitas. Sehingga dapat diketahui persamaan dari *logistic regression* seperti pada persamaan 2 [10].

$$y = \frac{1}{1+e^{-x}} \quad (3)$$

2.5 Evaluasi

Evaluasi merupakan tahapan pada penelitian dengan menunjukkan hasil dari *testing* yang dilakukan dalam model yang menghitung performa. Model tersebut ada *precision, recall, accuracy*, dan *F1-score*. Perhitungan tersebut didapatkan dari perhitungan *confusion matrix* sehingga mendapatkan persamaan sebagai berikut [8]:

$$precision = \frac{TP}{TP+FP} \quad (4)$$

$$recall = \frac{TP}{TP+FN} \quad (5)$$

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (6)$$

$$F1-score = 2 \times \frac{precision \times recall}{precision+recall} \quad (7)$$

3. Hasil dan Pembahasan

3.1 Pengumpulan Data

Pada pengumpulan data diawali mengambil data dari Kaggle lalu menyortir data sesuai tanggal yang diinginkan. Pada penelitian ini mengambil data dari Januari 2022 hingga Desember 2022. Lalu data dilabeli dengan berdasarkan bintang yaitu bintang 3 dilabeli 'Netral', bintang 2 dan 1 dilabeli 'Negatif', dan bintang 4 dan 5 dilabeli 'Positif'. Lalu data yang dilabeli dibagi menjadi 50:50 yaitu data berlabel positif dengan data berlabel negatif, sedangkan untuk data berlabel netral dihilangkan. Sehingga mendapatkan data seperti pada tabel 1.

Tabel 1. Dataset

dated	rating	label	review
2022-12-23 21:05:15	1	Negatif	Tak bisa diupdate, payah sekali
2022-12-23 20:22:03	5	Positif	Simple and smart
2022-12-23 20:15:23	1	Negatif	Aplikasi apa ini dikit2 update dikit2 update emng ngebantu tapi jangan keseringan di update dong
2022-12-23 20:09:38	5	Positif	Good joooob
2022-12-23 19:07:33	1	Negatif	Gimana si, malah makin parah di update

3.2 Preprocessing Data

Dari data tersebut selanjutnya dilakukan *cleaning data* dengan menghilangkan karakter-karakter spesial, angka, emoji, *link web*, dan menghilangkan kata yang berulang. Sehingga hasil proses cleaning data dapat dilihat pada gambar 2.

p-ISSN: 2986-3929
e-ISSN: 3032-1948

label	review	cleaning	hapusEmoji	double
Negatif	Tak bisa diupdate, payah sekali	Tak bisa diupdate payah sekali	Tak bisa diupdate payah sekali	Tak bisa diupdate payah sekali
Positif	Simple and smart	Simple and smart	Simple and smart	Simple and smart
Negatif	Aplikasi apa ini dikit2 update dikit2 update e...	Aplikasi apa ini dikit update dikit update e...	Aplikasi apa ini dikit update dikit update e...	Aplikasi apa ini dikit update dikit update e...
Positif	Good joooob	Good joooob	Good joooob	Good job
Negatif	Gimana si, malah makin parah di update	Gimana si malah makin parah di update	Gimana si malah makin parah di update	Gimana si malah makin parah di update

Gambar 2. *Cleaning* Data

Selanjutnya melakukan tahapan *casefolding* atau merubah semua data ke bentuk huruf kecil semua atau *lowercase*. Sehingga tidak ada kata yang sama memiliki arti yang beda bila terdapat perbedaan huruf besar atau kecil. Data yang telah melewati *casefolding* dapat dilihat pada gambar 3.

```
0                        tak bisa diupdate  payah sekali
1                                    simple and smart
2       aplikasi apa ini dikit  update dikit  update e...
3                                          good job
4                  gimana si  malah makin parah di update
                              ...
6339    masih mending mandiri sebelum ini  ui nya ribe...
6340    sangat membantu sistem pembayaran dan memperce...
6341    pembaharuan aplikasi ini hari gk bisa bisa ini...
6342                                          lemot
6343                            taii aplksi ssh di buka
Name: caseFolding, Length: 6344, dtype: object
```

Gambar 3. *Case Folding*

Setelah melalui *casefolding*, data selanjutnya diproses untuk dilakukan *tokenizing*. *Tokenizing* berguna untuk membagi kata per kata sehingga dapat dibaca oleh program perkata tidak perkalimat. Pada *tokenizing* spasi tidak dimasukkan atau dapat dikatakan dihilangkan. Untuk hasil *tokenizing* dapat dilihat pada gambar 4.

```
0                   [tak, bisa, diupdate, payah, sekali]
1                                [simple, and, smart]
2       [aplikasi, apa, ini, dikit, update, dikit, upd...
3                                          [good, job]
4           [gimana, si, malah, makin, parah, di, update]
                                ...
6339    [masih, mending, mandiri, sebelum, ini, ui, ny...
6340    [sangat, membantu, sistem, pembayaran, dan, me...
6341    [pembaharuan, aplikasi, ini, hari, gk, bisa, b...
6342                                           [lemot]
6343                        [taii, aplksi, ssh, di, buka]
Name: tokenizing, Length: 6344, dtype: object
```

Gambar 4. *Tokenizing*

Tahapan selanjutnya yaitu formalisasi mengubah kata tidak baku atau gaul serta kata yang terdapat penulisan menjadi benar. Dalam formalisasi terdapat kamus yang merepresentasikan kata-kata gaul atau tidak baku tersebut yang diambil dari 2000 data. Peneliti merasa dari 2000 data sudah dapat merepresentasikan kata tidak baku yang ada dalam data. Berikut merupakan kamus dari kata tersebut yang terdapat pada gambar 5.

p-ISSN: 2986-3929
e-ISSN: 3032-1948


```
{"tak":"tidak", "simple":"sederhana", "and":"dan", "smart":"pintar",
"dikit":"sedikit", "update":"diperbarui", "emng":"emang", "good":"bagus",
"apkikasinya":"aplikasi", "gk":"tidak", "bbrp":"beberapa",
"kluar":"keluar", "lgi":"lagi", "kudu":"harus", "ma":"dengan",
"dftr":"daftar", "tp":"tapi", "utk":"untuk", "jd":"jadi", "lemot":"lambat",
"okla":"oke", "penguna":"pengguna", "dilema":"ragu", "kemrn":"kemarin",
"pdhal":"padahal", "sy":"saya", "sdh":"sudah", "mamfaatnya":"manfaatnya",
"mantab":"mantap", "bgt":"banget", "mlh":"malah", "lg":"lagi",
"jdnya":"jadinya", "trus":"terus", "jgn":"jangan", "donk":"dong",
"kalu":"kalau", "d":"di", "pening":"pusing", "tup":"top", "lbh":"lebih",
"smoga":"semoga", "smngkin":"semakin", "ajh":"aja", "mantak":"minta",
"hilng":"hilang", "terpasa":"terpaksa", "bnk":"bank", "apk":"aplikasi",
"dsuruh":"disuruh", "mengenakn":"mengenakan", "klo":"kalo",
"dcancel":"dibatalkan", "esmosi":"emosi", "sdikit":"sedikit", "bsa":"bisa",
"dgn":"dengan", "trima":"terima", "kirkm":"kirimkan", "skrg":"sekarang",
"apus":"hapus", "quota":"kuota", "aptudate":"update", "ga":"gak",
"dll":"lainnya", "pki":"pakai", "tks":"thanks", "dech":"deh",
```


Gambar 5. Kamus Formalisasi

Dari gambar 5 dapat dilihat merupakan Sebagian dari kata singkatan atau kata tidak baku yang terdapat dalam data. Dari kamus tersebut dibaca dan mengganti data sehingga menghasilkan hasil seperti pada gambar 6.

```
0                  [tidak, bisa, diupdate, payah, sekali]
1                               [sederhana, dan, pintar]
2       [aplikasi, apa, ini, sedikit, diperbarui, sedi...
3                                          [bagus, job]
4       [gimana, si, malah, makin, parah, di, diperbarui]
                                ...
6339    [masih, mending, mandiri, sebelum, ini, tampil...
6340    [sangat, membantu, sistem, pembayaran, dan, me...
6341    [pembaharuan, aplikasi, ini, hari, tidak, bisa...
6342                                               [lambat]
6343                  [sialan, aplikasi, susah, di, buka]
Name: formalisasi, Length: 6344, dtype: object
```

Gambar 6. Formalisasi

Selanjutnya melakukan tahapan *stopword removal* yaitu menghilangkan kata-kata yang sering muncul atau tidak penting, contohnya “dan”, “saya”, “di”, dan sebagainya. Penghilangan kata-kata tersebut dipergunakan agar arti dari komentar tersebut diketahui bahwa memiliki sentimen negatif atau positif. Hasil dari *stopword removal* dapat dilihat pada gambar 7.

```
0                                     [diupdate, payah]
1                                   [sederhana, pintar]
2       [aplikasi, diperbarui, diperbarui, emang, ngeb...
3                                           [bagus, job]
4                         [gimana, si, parah, diperbarui]
                                ...
6339    [mending, mandiri, tampilan, nya, ribet, gak, ...
6340    [membantu, sistem, pembayaran, mempercepat, pe...
6341                          [pembaharuan, aplikasi, ya]
6342                                               [lambat]
6343                      [sialan, aplikasi, susah, buka]
Name: stopwordRemoval, Length: 6344, dtype: object
```

Gambar 7. *Stopword Removal*

Tahap terakhir yaitu melakukan *stemming* yang merubah kata menjadi kata dasar, atau dapat diartikan sebagai menghilangkan imbuhan pada kata sehingga hanya tediri dari kata dasar. Hasil dari proses *stemming* dapat dilihat pada gambar 8.

p-ISSN: 2986-3929
e-ISSN: 3032-1948

```
0                                    [diupdate, payah]
1                                  [sederhana, pintar]
2       [aplikasi, baru, baru, emang, ngebantu, sering...
3                                          [bagus, job]
4                                [gimana, si, parah, baru]
                                 ...
6339    [mending, mandiri, tampil, nya, ribet, gak, us...
6340         [bantu, sistem, bayar, cepat, putar, dagang]
6341                               [baharu, aplikasi, ya]
6342                                             [lambat]
6343                          [sial, aplikasi, susah, buka]
Name: stemming, Length: 6344, dtype: object
```

Gambar 8. *Stemming*

3.3 Ekstrasi Fitur

Pada bagian ini yaitu melakukan ekstraksi fitur TF-IDF setelah data melalui tahapan preprocessing. Pada bagian ini juga terdapat pembagian data testing dan data uji, dengan perbandingan data 80:20. Pembagian juga dibedakan dari 2 hal yaitu nilai x dan nilai y, hal ini dikarenakan *logistic regression* menggunakan perhitungan matematis. Nilai x sendiri diambil dari hasil stemming dan nilai y diambil dari label. Pada perhitungan pembobotan menggunakan bantuan dari *library TfidfVectorizer*. Sehingga potongan program pembobotan dapat dilihat pada gambar 9.

```
X = ulasan['stemming']
Y = ulasan['label']

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=64)

def dummy_fun(doc):
    return doc

vectorizer = TfidfVectorizer(analyzer='word',
    tokenizer=dummy_fun,
    preprocessor=dummy_fun,
    token_pattern=None)
x_train = vectorizer.fit_transform(x_train)
x_test = vectorizer.transform(x_test)
```

Gambar 9. Pembobotan TF-IDF

3.4 Pengaplikasian Logistic Regression

Pada pengaplikasian algoritma *logistic regression* menggunakan bantuan dari library LogisticRegression. Pada pengaplikasiannya terdapat c yang merepresentasikan regularisasi, semakin rendah nilai c maka semakin kuat regularisasi dan berarti model akan cenderung menjadi lebih sederhana. Hasil dari pengaplikasian program algoritma *logistic regression* dapat dilihat pada gambar 10.

```
C_values = [0.01, 0.05, 0.25, 0.5, 0.75, 1]

for c in C_values:
    lr = LogisticRegression(C=c)
    lr.fit(x_train, y_train)
    print('Accuracy for C=%s: %s' % (c, accuracy_score(y_test, lr.predict(x_test))))
```

```
Accuracy for C=0.01: 0.8289992119779354
Accuracy for C=0.05: 0.8589440504334122
Accuracy for C=0.25: 0.9062253743104807
Accuracy for C=0.5: 0.913317572892041
Accuracy for C=0.75: 0.9156816390858944
Accuracy for C=1: 0.9141055949566588
```

Gambar 10. *Logistic Regression*

p-ISSN: 2986-3929
e-ISSN: 3032-1948

Dapat dilihat dari hasil program tersebut akurasi tertinggi dihasilkan dari C = 0.75 yang menghasilkan 0,915 atau bila dipersentasekan menjadi 91,5%. Sedangkan nilai terendah didapatkan dari C = 0.01 yang menghasilkan 0.828 atau 82,8%.

3.5 Deteksi Kata

Pada bagian ini merupakan suatu program yang dapat digunakan untuk mengetahui suatu kata kunci memiliki sentimen positif atau sentimen negatif. Kata kunci dapat dikategorikan ke salah satu sentimen tersebut atau tidak keduanya dilihat dari hasil pembobotan TF-IDF yang sebelumnya dilakukan. Serta kata yang dimasukkan hanya sebuah kata tidak bisa kalimat. Bila kita memasukkan kata yang tidak berhubungan dengan aplikasi maka akan ada pemberitahuan bahwa kata tersebut tidak ada dalam fitur. Hasil deteksi kata dapat dilihat pada gambar 11, gambar 12, dan gambar 13.

```
Masukkan kata yang ingin dianalisis sentimennya: cepat
Sentimen positif terhadap kata 'cepat'
```

Gambar 11. Deteksi Kata Positif

```
Masukkan kata yang ingin dianalisis sentimennya: transfer
Sentimen negatif terhadap kata 'transfer'
```

Gambar 12. Deteksi Kata Negatif

```
Masukkan kata yang ingin dianalisis sentimennya: buah
Kata 'buah' tidak ditemukan dalam fitur
```

Gambar 13. Deteksi Kata Salah

3.6 Evaluasi

Pada evaluasi sendiri menunjukkan hasil *precision, recall, f1-score*, dan *accuracy* terhadap hasil data uji kita yang telah dibandingkan dengan hasil *testing* yang dilakukan. Sehingga mendapatkan hasil seperti gambar 14.

	precision	recall	f1-score
Negatif	0.89	0.95	0.92
Positif	0.94	0.88	0.91
accuracy			0.91
macro avg	0.92	0.91	0.91
weighted avg	0.92	0.91	0.91

Gambar 14. Evaluasi

Dari gambar 14 dapat dilihat hasil untuk sentimen negatif yaitu *precision* 89%, *recall* 95%, *f1-score* 92%. Sedangkan untuk sentimen positif menghasilkan *precision* 94%, *recall* 88%, *f1-score* 91%. Sehingga hasil akurasi keseluruhan mendapatkan 91% yang berarti analisis sentimen dengan metode algoritma *logistic regression* dan pembobotan TF-IDF sudah baik. Serta bila dilihat dari *wordcloud* mengenai kata yang sering muncul terhadap sentimen positif dan sentimen negatif dapat dilihat dari gambar 15 dan gambar 16.

p-ISSN: 2986-3929
e-ISSN: 3032-1948

Gambar 15. *Wordcloud* Positif

Gambar 16. *Wordcloud* Negatif

4. Kesimpulan

Berdasarkan penelitian yang dilakukan, maka mendapatkan kesimpulan bahwa algoritma *logistic regression* menghasilkan akurasi yang baik dalam analisis sentimen ulasan Livin by Mandiri dari ulasan *google play store*. Data yang diperoleh yaitu sebanyak 6344 data yang kemudian melakukan tahapan *labeling* serta membagi data agar seimbang yang selanjutnya dilakukan proses *preprocessing* data. Dari hasil *preprocessing* tersebut dilakukan pembobotan TF-IDF yang dilakukan untuk membantu perhitungan algoritma. Data dibagi menjadi data uji dan data latih dengan perbandingan 80:20, yaitu 80% data latih dan 20% data uji. Data kemudian diuji dengan metode algoritma *logistic regression* yang menghasilkan nilai C = 0.75 tertinggi yautu memiliki akurasi 91,5%. Serta menghasilkan untuk sentimen negatif yaitu *precision* 89%, *recall* 95%, *f1-score* 92%. Sedangkan untuk sentimen positif menghasilkan *precision* 94%, *recall* 88%, *f1-score* 91%. Terdapat pula program deteksi kata yang dapat memeriksa suatu kata kunci termasuk kedalam sentimen positif atau sentimen negatif sesuai dengan hasil testing. Saran untuk peneliti selanjutnya diharapkan untuk dapat memaksimalkan dalam proses *preprocessing* data serta dapat melakukan dengan algoritma lain ataupun metode *deep learning*.

Daftar Pustaka

[1] Bank Mandiri, “Bank Mandiri’s Transformation.” Accessed: May 06, 2024. [Online]. Available: https://www.bankmandiri.co.id/en/tentang-kami#:~:text=Bank%20Mandiri%20didirikan%20pada%202,yang%20dilaksanakan%20oleh%20pemerintah%20Indonesia.

[2] Bank Mandiri, “Permudah Transaksi Lifestyle Di Livin’, Bank Mandiri Luncurkan Fitur Livin’ Sukha,” 2020. Accessed: May 06, 2024. [Online]. Available: https://www.bankmandiri.co.id/en/news-detail?primaryKey=57363825&backUrl=/web/guest/news#:~:text=Sejak%20diluncurkan%20pada%20Oktober%202021,ritel%20yang%20menawarkan%20berbagai%20keunggulan.

[3] Bank Mandiri, “Fokus Penuhi Kebutuhan Nasabah, Bank Mandiri Sabet Penghargaan Marketing Company of The Year 2023 Versi AMF,” 2023. Accessed: May 06, 2024. [Online]. Available: https://www.bankmandiri.co.id/en/news-detail?primaryKey=219483836&backUrl=/search%3Fkeywords%3Dlaporan%20tahunan%26searchCategory%3D0

[4] M. Z. Hariansyah and S. Siswanto, “Implementasi Metode Multinomial Naive Bayes pada Analisis Sentimen Terhadap Layanan Aplikasi Livin by Mandiri,” in *Prosiding Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)*, 2022, pp. 517–524.

[5] N. R. Alfajri, “Perbandingan Analisis Sentimen Menggunakan Naive Bayes, Lexicon-Based, dan Hybrid pada Ulasan Pengguna Aplikasi Livin’by Mandiri,” 2023, Accessed: May 06, 2024. [Online]. Available: https://www.kaggle.com/datasets/itanium/livin-by-mandiri-app-reviews

[6] GHIFFARI AHMADIJAYA, “Livin’ by Mandiri App Reviews.” 2023.

[7] J. A. Septian, T. M. Fachrudin, and A. Nugroho, “Analisis Sentimen Pengguna Twitter Terhadap Polemik Persepakbolaan Indonesia Menggunakan Pembobotan TF-IDF dan K-Nearest Neighbor,” *INSYST: Journal of Intelligent System and Computation*, vol. 1, no. 1, pp. 43–49, 2019.

[8] P. W. A. Wibawa and C. R. A. Pramartha, “Analisis Sentimen pada Teks Berbahasa Bali Menggunakan Metode Multinomial Naive Bayes dengan TF-IDF dan BoW,” *Jurnal Nasional Teknologi Informasi dan Aplikasinya*, vol. 2, no. 1, pp. 37–46, 2023.

[9] B. B. Tangkere, “Analisis Performa Logistic Regression dan Support Vector Classification untuk Klasifikasi Email Phising,” *Jurnal Ekonomi Manajemen Sistem Informasi*, vol. 5, no. 4, pp. 442–450, 2024.

[10] Vijay Kanade, “What Is Logistic Regression? Equation, Assumptions, Types, and Best Practices.” Accessed: May 08, 2024. [Online]. Available: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/