Implementasi *Logistic Regression* dan SMOTE dalam Analisis Sentimen Ulasan Wondr by BNI

Komang Krisna Jaya Nova Antara[a1], Ngurah Agus Sanjaya ER[a2]

[a]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana
Jalan Raya Kampus Udayana, Bukit Jimbaran, Kuta Selatan, Badung, Bali, Indonesia
[1]krisnajayanova66@email.com
[2]agus_sanjaya@unud.ac.id

Abstract

Rapid innovation in Indonesia's digital banking, evidences by digital transactions reaching Rp7,492.93 trillion by September 2024 and the launch of Wondr by BNI by PT. Bank Negara Indonesia, which has 5.3 million active users and 397 thousand reviews on Google Play Store by June 2025, presents a challenge for manual sentiment analysis of reviews due to its inefficiency. This study addresses this issue by employing a machine learning approach, utilizing the Logistic Regression algorithm for sentiment analysis. A total of 8000 review data from Kaggle were used, with sentiment labeled based on rating scores (1-3 negative, 4-5 positive). The methodology included data preprocessing, feature weighting using Term Frequency-Inverse Document Frequency (TF-IDF), and balancing training data with Synthetic Minority Oversampling (SMOTE). The Logistic Regression model was trained after parameter optimization via grid search, yielding the optimal combination of C=1, penalty='l2', and solver='newton-cg'. Evaluation using a confusion matrix revealed an overall accuracy of 93.54%. For negative sentiment, the model achieved 71.89% precision, 91.5% recall, and 80.52% F1-score, while for positive sentiment, it reached 98.48% precision, 93.89% recall, and 96.13% F1-score.

Keywords: *Sentiment Analysis, Wondr by BNI, Logistic Regression, SMOTE, Digital Banking*

1. Pendahuluan

Pesatnya perkembangan teknologi telah mendorong inovasi di berbagai sektor, termasuk industri perbankan digital. Di Indonesia, aktivitas transaksi perbankan digital mengalami peningkatan yang signifikan. Berdasarkan data Bank Indonesia (2024), nilai transaksi *digital banking* mencapai Rp7.492,93 triliun pada September 2024, tumbuh sebesar 54,89% dibandingkan periode yang sama pada tahun sebelumnya [1]. Salah satu institusi perbankan yang berperan aktif dalam perkembangan ini adalah PT. Bank Negara Indonesia yang terbentuk pada 5 Juli 1946 [2]. Pada Juli 2024, PT. Bank Negara Indonesia meluncurkan Wondr by BNI sebagai aplikasi *mobile banking* mereka yang merupakan pengganti BNI Mobile Banking dan telah mencapai 5,3 juta pengguna aktif [1]. Layanan *mobile banking* Wondr by BNI ini diharapkan dapat meningkatkan efisiensi transaksi sekaligus memberikan layanan dengan kualitas yang lebih baik lagi untuk para nasabah. Namun, seiring dengan meningkatnya popularitas *mobile banking,* muncul pula berbagai tantangan dalam pengalaman pengguna, sebagaimana terlihat dari ulasan di *Google Play Store*. Terhitung hingga 7 Juni 2025 sudah terdapat 397 ribu ulasan yang diberikan dengan *rating* rata-rata 4,8. Ulasan pengguna ini dapat menjadi sumber informasi yang sangat berharga bagi pihak bank dalam meningkatkan layanan. Namun, melakukan analisis manual terhadap ribuan ulasan sangat tidak efisien. Oleh karena itu, diperlukan pendekatan berbasis *machine learning* guna mengotomatisasi proses analisis sentimen pada ulasan aplikasi Wondr by BNI.

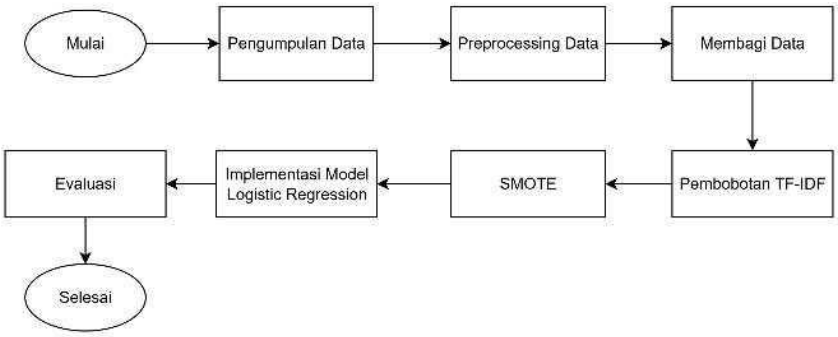
Analisis sentimen adalah teknik untuk mengidentifikasi pola dalam data tekstual untuk mendapatkan informasi [3]. Salah satu algoritma yang umum digunakan untuk analisis sentimen adalah *Logistic Regression*. Dalam penelitian sebelumnya, telah dilakukan perbandingan antara algoritma *Support Vector Machine* dengan *Logistic Regression* dalam melakukan analisis

sentimen pengguna aplikasi retail di android. Hasil pengujian menunjukan algoritma *Logistic Regression* menunjukkan kinerja yang lebih baik dengan akurasi mencapai 87% [4]. Penelitian lain juga membandingkan algoritma *Logistic Regression* dengan algoritma *Naïve Bayes* untuk analisis sentimen metaverse. Hasilnya, algoritma *Naïve Bayes* menghasilkan akurasi 90%, sedangkan algoritma *Logistic Regression* sedikit lebih unggul dengan akurasi 91% [5]. Selain itu, untuk mengatasi ketidakseimbangan data, dilakukan optimasi menggunakan *Synthetic Minority Oversampling* (SMOTE). Algoritma *Logistic Regression* dengan menggunakan SMOTE menunjukkan peningkatan akurasi sebesar 4% dibandingkan ketika tidak menggunakan SMOTE [5].

Sehingga algoritma *Logistic Regression* akan digunakan pada penelitian ini untuk melakukan analisis sentimen terhadap ulasan pengguna aplikasi Wondr by BNI di *Google Play Store*. Selain itu, pada penelitian ini juga menerapkan metode SMOTE dalam mengatasi ketidakseimbangan data dalam proses analisis sentimen.

2. Metode Penelitian

Penelitian ini dilaksanakan melalui serangkaian tahapan yang tersusun secara sistematis untuk memperoleh hasil yang optimal dan maksimal.



Gambar 1. Tahapan Penelitian

Gambar 1 menjelaskan alur penelitian yang meliputi pengumpulan data ulasan aplikasi Wondr by BNI. Setelah data didapatkan, selanjutnya data melalui tahap *peprocessing* sebelum digunakan pada model. Kemudian dilakukan pembagian data untuk data latih dan data uji. Selanjutnya akan dilakukan tahap ekstraksi fitur dengan metode *Term Frequency-Inverse Document Frequency* (TF-IDF). Tahap selanjutnya adalah meyeimbangkan data latih dengan *Synthetic Minority Oversampling* (SMOTE) dan dilanjutkan dengan implementasi model *Logistic Regression* dengan data latih yang sudah disiapkan. Tahapan terakhir adalah evaluasi, yaitu tahap untuk mengetahui tingkat akurasi yang dihasilkan oleh model.

2.1. Metode Perolehan Data

Data ulasan aplikasi Wondr by BNI di platform *Google Play Store* yang digunakan pada penelitian kali ini diambil dari *Kaggle* yang terdiri dari 8000 data. *Dataset* ini tidak terdapat label sentimen, namun terdapat fitur *score* yang dapat diasumsikan sebagai representasi sentimen, dimana *score* 1-3 merupakan sentimen negatif, dan *score* 4-5 merupakan sentimen positif.

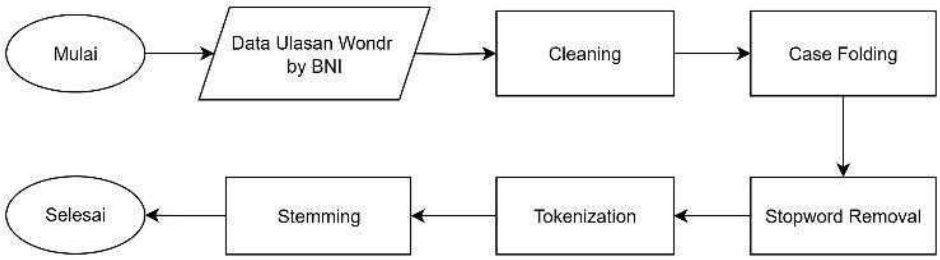
Tabel 1. Karakteristik *Dataset*

No	Fitur	Tipe	Deskripsi
1	username	Nominal	Label atau nama pengguna.
2	score	Ordinal	Skor rating (1–5), menunjukkan tingkat kepuasan.

No	Fitur	Tipe	Deskripsi
3	at	Datetime	Tanggal dan waktu ulasan dibuat.
4	content	Unstructured Text	Isi ulasan pengguna.

2.2. Data *Preprocessing*

Dalam tahap data *preprocessing,* serangkaian tahapan dilakukan untuk mempersiapkan *dataset* sebelum digunakan pada tahap-tahap selanjutnya. Adapun tahapan-tahapan dari data *preprocessing* pada penelitian ini terlampir pada Gambar 2.



Gambar 2. Tahapan Data *Preprocessing*

a. *Cleaning*
Cleaning data adalah tahapan dalam *preprocessing* data yang bertujuan untuk menghilangkan elemen yang tidak diperlukan dalam data, seperti data duplikat, data kosong, karakter maupun tanda baca, URL, *HTML tags*, emoji, dan angka [5].

b. *Case Folding*
Tahap *Case folding* bertujuan untuk mengatur seluruh teks dalam sebuah dokumen dengan huruf kecil agar konsisten dan mempermudah pencarian [6].

c. *Stopword Removal*
Tahap *Stopword removal* berjutuan untuk menghilangkan kata-kata yang tidak memberikan kontribusi dan tidak memiliki pengaruh signifikan terhadap makna dalam sebuah kalimat [1]. Tahap ini bertujuan untuk menyederhanakan teks dan berfokus pada informasi yang relevan.

d. *Tokenization*
Tahap *Tokenization* merupakan proses untuk memecah suatu kalimat atau paragraf menjadi kata-kata individu atau token, dengan menggunakan karakter spasi sebagai pemisah [5]. Tahap ini memiliki tujuan untuk menyederhanakan teks dan mengubahnya menjadi representasi yang lebih terstruktur sehingga dapat diolah oleh sistem.

e. *Stemming*
Tahap *Stemming* bertujuan untuk mengganti suatu kata berimbuhan menjadi bentuk kata dasar [5].

2.3. Pembobotan TF-IDF (*Term Frequency-Inverse Document Frequency*)

Term Frequency-Inverse Document Frequency (TF-IDF) adalah jenis pembobotan yang populer dan sering digunakan karena mampu memberikan akurasi dan *recall* yang relatif tinggi [7]. Pembobotan TF-IDF adalah metode yang dikenal baik untuk mengevaluasi pentingnya sebuah kata dalam sebuah dokumen [8]. TF-IDF terdiri dari 2 bagian, yaitu TF (*Term Frequency*) dan IDF (*Inverse Document Frequency*), TF adalah jumlah kemunculan tiap kata dalam sebuah dokumen, semakin banyak kata muncul dalam tiap dokumen maka semakin besar nilai TF. Sedangkan IDF adalah jumlah nilai dokumen pada tiap kata yang berbanding terbalik, yaitu apabila suatu kata

jarang muncul pada sebuah dokumen, maka nilai IDF lebih besar daripada kata yang sering muncul [9]. Pembobotan kata dengan TF-IDF memiliki tahapan-tahapan dengan rumus matematik sebagai berikut:

a. Persamaan untuk menghitung TF:

$$TF(t_k, d_j) = f(t_k, d_j) \quad (1)$$

b. Persamaan untuk menghitung IDF:

$$IDF(t_k) = log \frac{D}{df(t)} \quad (2)$$

c. Persamaan untuk menghitung TF-IDF:

$$TF\ IDF(t_k, d_j) = TF(t_k, d_j).IDF(t_k) \quad (3)$$

2.4. *Synthetic Minority Oversampling* (SMOTE)

Synthetic Minority Oversampling atau SMOTE adalah suatu metode untuk mengatasi tantangan ketidakseimbangan distribusi kelas pada data dengan cara membuat data sintetis pada kelas minoritas [10]. Metode ini bekerja dengan membuat data sintetis di sepanjang garis penghubung antara titik-titik data yang terdapat dalam kelas minoritas dengan tujuan meningkatkan jumlah sampel kelas minoritas tanpa menghasilkan replika yang identik atau titik data yang terlalu berbeda. Dengan hal itu, SMOTE secara efektif mengurangi masalah *overfitting* yang umumnya terjadi pada metode *random oversampling*[11].

2.5. Logistic Regression

Salah satu algoritma yang digunakan dalam pengklasifikasian data adalah *Logistic Regression*, khususnya digunakan untuk memprediksi probabilitas atau kemungkinan dari suatu termasuk dalam satu atau dua kelas [12]. Algoritma ini memodelkan hubungan variabel independen dengan *output* klasifikasi biner, dengan memanfaatkan pendekatan probabilistik untuk memprediksi nilai dari variabel dependen[13]. Persamaan *Logistic Regression* ditunjukkan pada persamaan 4 [13].

$$\sigma(t) = \frac{1}{1+\exp(-t)} \quad (4)$$

2.6. Confusion Matrix

Evaluasi kinerja model atau algoritma dilakukan dengan menggunakan sejumlah metrik, yaitu *accuracy*, *precision*, *recall*, dan *F*1-*score* dalam bentuk *confusion matrix. Confusion matrix* merupakan sebuah tabel yang digunakan untuk melihat perbandingan antara jumlah prediksi yang tepat dan keliru yang terdapat pada tiap-tiap kelas. Berikut adalah persamaan untuk mengukur kinerja algoritma dengan *confusion matrix* [14]:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (5)$$

$$Precision = \frac{TP}{TP+FP} \quad (6)$$

$$Recall = \frac{TP}{TP+FN} \quad (7)$$

$$F1 - score = \frac{2(recall \times precision)}{(recall + precision)} \quad (8)$$

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

3. Hasil dan Pembahasan

3.1. Perolehan Data

Perolehan data dilakukan dengan cara mengambil data sekunder terkait ulasan aplikasi Wondr by BNI dari *Kaggle*. Karena dataset ini belum berisi label sentimen, maka data ini diberi label berdasarkan skor bintang dari tiap ulasan, dimana bintang 1, 2, dan 3 dilabeli 'Negatif', dan bintang 4 dan 5 dilabeli 'Positif'.

Tabel 2. *Dataset*

userName	score	at	content	Sentiment
Pengguna Google	1	2024-09-08 00:45:30	OTP EMAIL SELALU SALAH, PERBAIKI DULU BARU LAUNCHING APLIKASINYA	Negatif
Pengguna Google	5	2024-09-07 08:21:45	Dengan wondr menjadikan transaksi lebih mudah dan lebih cepat.	Positif
Pengguna Google	3	2024-09-07 03:32:30	Kok tidak ada mobile tunai. Pakai yang lama aja dulu.	Negatif
Pengguna Google	2	2024-09-06 07:49:36	Apk makin lama makin lemot	Negatif
Pengguna Google	4	2024-09-05 15:16:49	Lebih ringkas penggunaan nya, walau menu tidak selengkap mbanking nya	Positif

3.2. Data *Preprocessing*

Setelah pengumpulan data, tahap berikutnya adalah pengolahan data yang dilakukan dengan beberapa tahapan dengan, yaitu *cleaning* data untuk menghilangkan noise, *case folding* untuk membuat teks menjadi huruf kecil, *stopword removal* untuk memilah kata yang relevan, *tokenization* untuk memcah kalimat menjadi kata, dan *stemming* untuk menghapus imbuhan.

Tabel 3. Contoh *Preprocessing* Data

Tahapan	Hasil Tahapan Preprocessing	Sentiment
Data Awal	2 aplikasi dari BNI memang ok, Wonder dan BNI Mobile Banking, kami pake keduanya untuk traksaksi keuangan, mantap 👍Terimakasih🙏	Positif
Cleaning	aplikasi dari BNI memang ok Wonder dan BNI Mobile Banking kami pake keduanya untuk traksaksi keuangan mantap Terimakasih	Positif
Case Folding	aplikasi dari bni memang ok wonder dan bni mobile banking kami pake keduanya untuk traksaksi keuangan mantap terimakasih	Positif
Stopword Removal	aplikasi bni memang wonder bni mobile banking pake keduanya traksaksi keuangan mantap terimakasih	Positif
Tokenization	[aplikasi, bni, memang ,wonder, bni, mobile, banking, pake, keduanya, traksaksi, keuangan, mantap, terimakasih]	Positif
Stemming	[aplikasi, bni, memang ,wonder, bni, mobile, banking, pake, dua, traksaksi, keuangan, mantap, terimakasih]	Positif

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

3.3. Perhitungan Ekstraksi Fitur TD-IDF

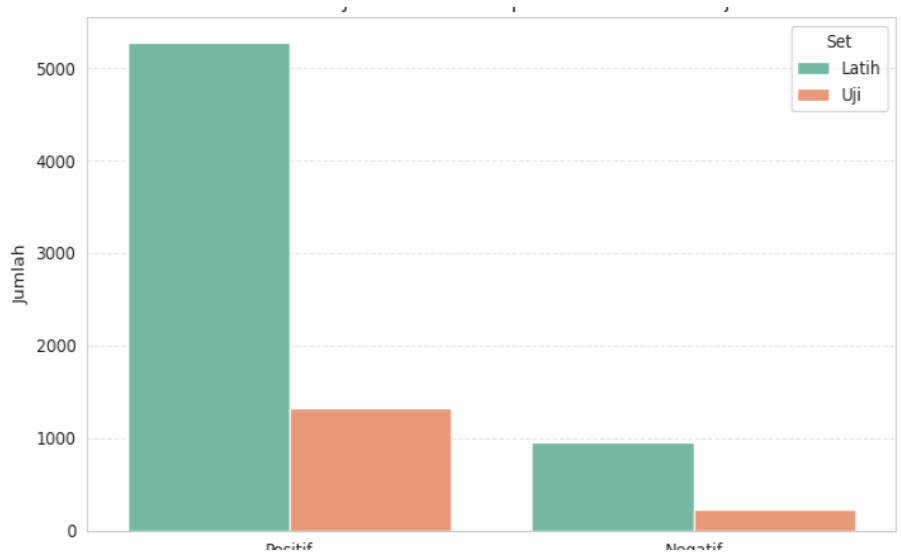
Setelah tahap *preprocessing* data, tahap selanjutnya adalah perhitungan ekstraksi fitur TF-IDF dengan mengonversi data tekstual ke bentuk numerik sehingga dapat digunakan sebagai *input* pada model *Logistic Regression*. Ekstraksi fitur ini akan dilakukan menggunakan *library sklearn.feature_extraxtion.text* untuk membantu mengonversi teks menjadi bentuk numerik. Pada tahap ini juga dilakukan pembagian data untuk data latih dan data uji dengan skala perbandingan 70:30. Berikut adalah potongan program pembobotan TF-IDF:

```
[81] from sklearn.model_selection import train_test_split

     vectorizer = TfidfVectorizer(max_features=5000, )
     X = df_model['stemmed_data']
     y = df_model['Sentiment']

     #split data
     x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)

     #visualisasi distribusi data
     visualize_sentiment_distribution(y_train, y_test)

     #vectorisasi TF-IDF
     X_train_vec = vectorizer.fit_transform(x_train)
     X_test_vec = vectorizer.transform(x_test)
```

Gambar 3. Implementasi Ekstraksi Fitur TF-IDF

3.4. SMOTE *Oversampling*

Dari hasil pembagian data yang dilakukan pada tahap perhitungan ekstraksi fitur TF-IDF, ditemukan bahwa terdapat ketidakseimbangan kelas pada data, karena jumlah kelas positif lebih banyak daripada kelas negatif yang divisualisasikan pada Gambar 4.



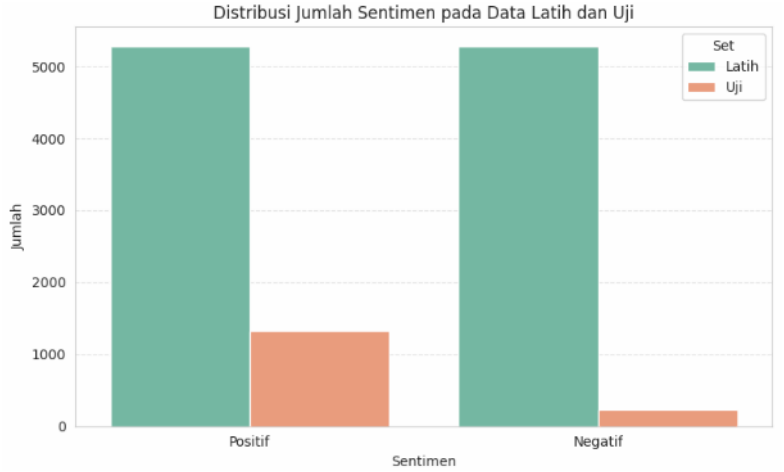
Gambar 4. Distribusi Kelas Sebelum SMOTE *Oversampling*

Oleh karena itu, metode SMOTE akan diterapkan guna menyeimbangkan data. Implementasi SMOTE pada penelitian ini menggunakan *library imblearn.over_sampling*. *Oversampling* akan dilakukan pada data latih yang selanjutnya akan digunakan untuk melatih model. Potongan program *oversampling* data dengan SMOTE dapat dilihat pada Gambar 5. Distribusi kelas pada data latih berhasil diseimbangkan setelah dilakukan *oversampling*, yang divisualisasikan dalam Gambar 6.

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

```
from imblearn.over_sampling import SMOTE

#SMOTE Oversampling
smote = SMOTE(sampling_strategy='auto', random_state=42)
x_resampled, y_resampled = smote.fit_resample(X_train_vec, y_train)

#visualisasi data setelah SMOTE
visualize_sentiment_distribution(y_resampled, y_test)
```

Gambar 5. Implementasi SMOTE *Oversampling*



Gambar 6. Distribusi Kelas SMOTE *Oversampling*

3.5. Implementasi Model *Logistic Regression*

Implementasi model *Logistic Regression* dalam penelitian ini dilakukan dengan menggunakan *library scikit-learn*, serta menggunakan teknik *grid search* untuk menemukan kombinasi parameter terbaik. Proses pencarian parameter terbaik dilakukan memvariasikan nilai pada parameter C dengan rentang nilai [0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1], parameter *penalty* dengan pilihan ['l1', 'l2'], dan parameter *solver* dengan pilihan ['lbfgs','newton-cg','liblinear','sag','saga'] menggunakan *GridSearchCV* dengan *cross-validation 5-fold*. Hasil pencarian menunjukkan bahwa kombinasi parameter terbaik adalah C=1, *penalty*='l2', dan *solver*='newton-cg' dengan *max_iter*=1500. Parameter C berfungsi mengatur kekuatan regularisasi, dimana semakin kecil nilainya maka regularisasi semakin kuat. Sementara itu, *penalty* 'l2' menerapkan regularisasi *ridge,* dan parameter *solver* 'newton-cg' dipilih karena efisien dalam menangani regularisasi L2 pada dataset berukuran sedang hingga besar, dan mampu menangani optimisasi dengan cukup stabil. Model kemudian dilatih menggunakan data latih yang telah melalui teknik SMOTE dan transformasi fitur menggunakan TF-IDF.

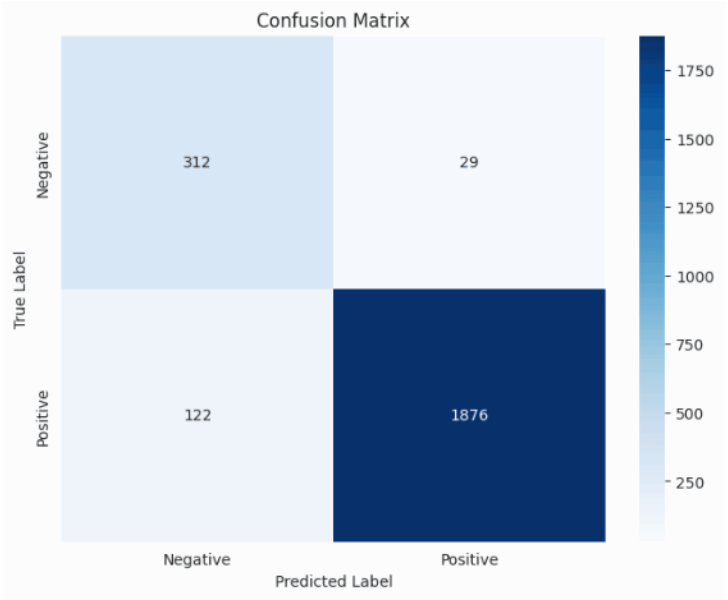
```
from sklearn.model_selection import GridSearchCV

model = LogisticRegression(class_weight='balanced')

param_grid_logrec = {'C': [0.01, 0.05, 0.25, 0.5, 0.75, 1],
                     'penalty': ['l1', 'l2'],
                     'solver': ['lbfgs','newton-cg','liblinear','sag','saga']}
grid_search_logrec = GridSearchCV(LogisticRegression(max_iter=5000), param_grid_logrec, cv=5, scoring='accuracy')
grid_search_logrec.fit(x_resampled, y_resampled)

best_log = grid_search_logrec.best_estimator_
y_pred_logmodel = best_log.predict(X_test_vec)
print("Parameter Logistic Regression: ", grid_search_logrec.best_params_)
print("Classification Report: \n", classification_report(y_test, y_pred_logmodel, digits=4))
```

Gambar 7. Implementasi Model *Logistic Regression*

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

3.6. Evaluasi

Setelah model *Logistic Regression* dilatih, Langkah selanjutnya adalah mengevaluasi performa dari model dengan menggunakan data uji.



Gambar 8. *Confusion Matrix Logistic Regression*

Berdasarkan Gambar 8, model berhasil memprediksi 312 data sentimen negatif, meskipun terdapat 28 sentimen negatif yang keliru diprediksi sebagai positif. Sementara itu, model berhasil memprediksi 1876 data sentimen positif, namun masih terdapat 122 sentimen positif yang keliru diprediksi sebagai negatif. Hasil evaluasi menunjukkan bahwa model *Logistic Regression* menghasilkan nilai *accuracy*, *precision*, *recall*, dan *F*1-*score,* yang dapat dilihat pada Gambar 9.

	precision	recall	f1-score	support
Negatif	0.7189	0.9150	0.8052	341
Positif	0.9848	0.9389	0.9613	1998
accuracy			0.9354	2339
macro avg	0.8518	0.9269	0.8832	2339
weighted avg	0.9460	0.9354	0.9385	2339

Gambar 9. Hasil Evaluasi

Dari Gambar 9 dapat dilihat hasil evaluasi untuk sentimen negatif dengan nilai *precision* 71.89%, *recall* 91.5%, dan *F*1-*score* 80.52%. Sementara itu, untuk sentimen positif menghasilkan nilai *precision* 98.48%, *recall* 93.89%, dan *F*1-*score* 96.13%. Sehingga secara keseluruhan, akurasi model mencapai 93.54%. Selain itu, dilakukan juga pengujian model dalam melakukan analisis sentimen dengan memasukkan teks uji secara manual seperti yang ditunjukkan pada Gambar 10. Lebih lanjut, visualisasi kata-kata yang paling dominan dalam sentimen positif maupun sentimen negatif ditampilkan melalui visualisasi *wordcloud* pada Gambar 11.

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

```
===== HASIL PREDIKSI SENTIMEN =====
Teks: 'keren banget apliasinya puas sama tampilannya'
=> Prediksi Final: POSITIF
=> Skor Probabilitas:
   - Negatif: 4.46%
   - Positif: 95.54%
------------------------------
Teks: 'tampilan sudah baik, namun cardless tidak ada dan sering ada masalah yang parah dan saya membencinya'
=> Prediksi Final: POSITIF
=> Skor Probabilitas:
   - Negatif: 45.82%
   - Positif: 54.18%
------------------------------
Teks: 'wondr bagus tapi sayangnya sering gagal verifikasi'
=> Prediksi Final: NEGATIF
=> Skor Probabilitas:
   - Negatif: 73.21%
   - Positif: 26.79%
------------------------------
Teks: 'Aplikasinya keren dan simpel, tapi loadingnya lama'
=> Prediksi Final: POSITIF
=> Skor Probabilitas:
   - Negatif: 15.78%
   - Positif: 84.22%
```

Gambar 10. Pengujian Manual

Gambar 11. *Wordcloud* Positif dan Negatif

4. Kesimpulan

Dari rangkaian analisis yang telah dilakukan, dapat disimpulkan bahwa penerapan algoritma *Logistic Regression* menghasilkan akurasi yang baik dalam analisis sentimen ulasan pengguna aplikasi Wondr by BNI dari *Google Play Store* dan ditunjang dengan pembobotan fitur dengan TF-IDF yang dilakukan untuk mengubah data tekstual menjadi representasi numerik yang bisa diolah oleh model. Untuk mengatasi ketidakseimbangan kelas dalam dataset, diterapkan metode *oversampling* SMOTE pada data latih. Data kemudian diuji dengan metode algoritma *Logistic Regression* yang dioptimalkan dengan *grid search* dan menghasilkan kombinasi parameter terbaik C=1, *penalty*='l2', dan *solver*='newton-cg'. Evaluasi performa model menggunakan *confusion matrix* menunjukkan akurasi keseluruhan sebesar 93.54%. Secara lebih rinci, untuk sentimen negatif, model mencapai *precision* 71.89%, *recall* 91.5%, dan *F*1-*score* 80.52%, sementara untuk sentimen positif, model menghasilkan *precision* 98.48%, *recall* 93.89%, dan *F*1-*score* 96.13%. Disarankan untuk penelitian selanjutnya agar mempertimbangkan penggunaan algoritma lain atau metode *deep learning* serta melakukan eksplorasi teknik *oversampling* sehingga memperoleh model dengan performa terbaik.

Daftar Pustaka

[1] I. Bazar, F. Wajidi, and A. A. A. Cirua, “Analisis Sentimen Ulasan Aplikasi Wondr By Bni Menggunakan Algoritma Svm Dengan Optimasi Kernel Trick,” *STORAGE: Jurnal Ilmiah Teknik dan Ilmu Komputer*, vol. 4, no. 2, pp. 69–81, May 2025, doi: 10.55123/storage.v4i2.5178.

[2] I. A. Aldrin, Z. Zahara, and J. Sudiman, “Analisis Tingkat Minat Masyarakat Pekanbaru Terhadap Mobile Banking BNI Menggunakan Pendekatan TAM,” *Jurnal Akuntansi, Bisnis dan Ekonomi Indonesia (JABEI)*, vol. 2, no. 2, pp. 118–126, Aug. 2023, doi: 10.30630/jabei.v2i2.152.

[3] D. Alita and A. R. Isnain, “Pendeteksian Sarkasme pada Proses Analisis Sentimen Menggunakan Random Forest Classifier,” *jurnal komputasi*, vol. 8, no. 2, Oct. 2020, doi: 10.23960/komputasi.v8i2.2615.

[4] A. G. Budianto, R. Rusilawati, A. T. E. Suryo, G. R. Cahyono, A. F. Zulkarnain, and M. Martunus, “Perbandingan Performa Algoritma Support Vector Machine (SVM) dan Logistic Regression untuk Analisis Sentimen Pengguna Aplikasi Retail di Android,” *Jurnal Sains dan Informatika*, vol. 10, no. 2, Nov. 2024, doi: 10.34128/jsi.v10i2.911.

[5] B. Ramadhani and R. R. Suryono, “Komparasi Algoritma Naïve Bayes dan Logistic Regression Untuk Analisis Sentimen Metaverse,” *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 8, no. 2, pp. 714–725, Apr. 2024, doi: 10.30865/mib.v8i2.7458.

[6] A. D. A. Putra and S. Juanita, “Analisis Sentimen pada Ulasan pengguna Aplikasi Bibit Dan Bareksa dengan Algoritma KNN,” *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, vol. 8, no. 2, pp. 636–646, Jun. 2021, doi: 10.35957/jatisi.v8i2.962.

[7] C. H. Yutika, A. Adiwijaya, and S. Al Faraby, “Analisis Sentimen Berbasis Aspek pada Review Female Daily Menggunakan TF-IDF dan Naïve Bayes,” *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 5, no. 2, pp. 422–430, Apr. 2021, doi: 10.30865/mib.v5i2.2845.

[8] R. Ahuja, A. Chug, S. Kohli, S. Gupta, and P. Ahuja, “The Impact of Features Extraction on the Sentiment Analysis,” *Procedia Comput Sci*, vol. 152, pp. 341–348, 2019, doi: 10.1016/j.procs.2019.05.008.

[9] 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,” *Journal of Intelligent System and Computation*, vol. 1, no. 1, pp. 43–49, Aug. 2019, doi: 10.52985/insyst.v1i1.36.

[10] S. Matharaarachchi, M. Domaratzki, and S. Muthukumarana, “Enhancing SMOTE for imbalanced data with abnormal minority instances,” *Machine Learning with Applications*, vol. 18, Dec. 2024, doi: 10.1016/j.mlwa.2024.100597.

[11] R. Asif, D. Upadhyay, M. Zaman, and S. Sampalli, “Enhancing Diabetes Risk Prediction: A Comparative Evaluation of Bagging, Boosting, and Ensemble Classifiers with SMOTE Oversampling,” *Inform Med Unlocked*, Jun. 2025, doi: 10.1016/j.imu.2025.101661.

[12] H. Junianto, R. E. Saputro, B. A. Kusuma, and D. I. S. Saputra, “COMPARISON OF LOGISTIC REGRESSION AND RANDOM FOREST IN SENTIMENT ANALYSIS OF DISDUKCAPIL APPLICATION REVIEWS,” *Jurnal Teknik Informatika (Jutif)*, vol. 5, no. 6, pp. 1539–1547, Feb. 2024, doi: 10.52436/1.jutif.2024.5.6.1802.

[13] M. S. T. Putra and Y. Azhar, “Perbandingan Model Logistic Regression dan Artificial Neural Network pada Prediksi Pembatalan Hotel,” *JISKA (Jurnal Informatika Sunan Kalijaga)*, vol. 6, no. 1, pp. 29–37, Jan. 2021, doi: 10.14421/jiska.2021.61-04.

[14] Erlin, Y. N. Marlim, Junadhi, L. Suryati, and N. Agustina, “Deteksi Dini Penyakit Diabetes Menggunakan Machine Learning dengan Algoritma Logistic Regression,” *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, vol. 11, no. 2, pp. 88–96, May 2022, doi: 10.22146/jnteti.v11i2.3586.