e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

Analisis Sentimen Menggunakan Metode Klasifikasi *Support Vector Machine* (SVM) dan Seleksi Fitur *Chi-Square*

Ewen Hokijuliandy, Herlina Napitupulu, Firdaniza Firdaniza
Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Padjadjaran
Email: ewen19001@mail.unpad.ac.id, herlina@unpad.ac.id, firdaniza@unpad.ac.id

Abstrak

Analisis sentimen adalah teknik komputasi untuk mengidentifikasi opini, sikap, emosi, dan maksud seseorang terhadap suatu subjek melalui ulasan yang diberikan. Studi sebelumnya menunjukkan teknik analisis sentimen menggunakan *machine learning*, seperti metode klasifikasi *Support Vector Machine* (SVM) telah terbukti efektif dalam mengklasifikasi opini. Penerapan metode seleksi fitur dapat meningkatkan performa model dan efisiensi model. Salah satu metode yang sering digunakan untuk seleksi fitur adalah metode *Chi-Square*. Penelitian ini bertujuan untuk memperoleh model SVM dari data teks yang telah melewati tahap seleksi fitur *Chi-Square*. Analisis sentimen dilakukan dengan kerangka kerja yang terdiri dari *text preprocessing*, representasi kata *Term Frequency Inverse Document Frequency* (TF-IDF), seleksi fitur *Chi-Square*, klasifikasi menggunakan metode SVM, evaluasi performa model, dan *hyperparameter tuning*.

Kata Kunci: Analisis sentiment, *machine learning*, SVM, *Chi-Square*, *hyperparameter tuning*.

Abstract

Sentiment analysis is a computational technique to identify opinions, attitudes, emotions, and intentions of an individual towards a subject through the provided reviews. Previous studies have shown that sentiment analysis techniques using machine learning, such as the Support Vector Machine (SVM) classification method, have proven effective in classifying opinions. The application of feature selection methods can improve the performance and efficiency of the model. One commonly used method for feature selection is the Chi-Square method. This research aims to obtain an SVM model from text data that has undergone the Chi-Square feature selection stage. Sentiment analysis is conducted within a framework consisting of text preprocessing, Term Frequency Inverse Document Frequency (TF-IDF) word representation, Chi-Square feature selection, classification using the SVM method, model performance evaluation, and hyperparameter tuning.

Keywords: *Sentiment analysis, machine learning, SVM, Chi-Square, hyperparameter tuning.*

1 PENDAHULUAN

Analisis sentimen adalah sebuah teknik untuk mendeteksi pendapat yang menguntungkan dan tidak menguntungkan terhadap subjek tertentu (seperti organisasi dan produk mereka) yang dapat digunakan untuk berbagai tujuan (Nasukawa dan Yi, 2003). Menurut Medhat *et al*. (2014), analisis sentimen atau *opinion mining* adalah studi komputasi opini, sikap, dan emosi orang terhadap suatu entitas. Sedangkan Shaik *et al*. (2022) menyatakan analisis sentimen adalah salah satu aplikasi *Natural Language*

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

Processing (NLP) yang paling banyak digunakan untuk mengidentifikasi maksud seseorang dari ulasan mereka. Analisis sentimen dilakukan menggunakan *machine learning* untuk mengklasifikasikan teks ulasan sebagai positif, netral, atau negatif.

Menurut Uysal dan Gunal (2014), tahapan kerangka kerja untuk klasifikasi teks terdiri dari *preprocessing*, representasi kata, seleksi fitur, klasifikasi, dan evaluasi performa model. Selanjutnya, performa model dapat ditingkatkan dengan *hyperparameter tuning*. *Hyperparameter tuning* merupakan proses mencari *hyperparameter* model yang lebih optimal. Setiap tahap kerangka kerja memengaruhi hasil performa model klasifikasi yang dibuat. Evaluasi hasil performa model dapat dilakukan dengan mengukur metrik *Accuracy, Precision, Recall,* dan *F1-Score*. Pemilihan metrik evaluasi yang digunakan bergantung pada kompleksitas dan distribusi data yang dimiliki. Şahin dan Klç (2019) menggunakan metrik *F1-Score* untuk mengevaluasi model klasifikasi untuk *imbalanced dataset* pada *Reuters*-21578 *dataset*. Pada kasus *imbalanced dataset, F1-Score* umum digunakan karena menggabungkan *precision* dan *recall* secara merata untuk kelas mayoritas dan minoritas. Metrik performa model ini menjadi acuan dalam melakukan *hyperparameter tuning*.

Pada tahap klasifikasi dengan *machine learning*, Mantovani *et al*. (2019) menyatakan bahwa kebanyakan algoritma *machine learning* sensitif terhadap nilai *hyperparameter* yang langsung berpengaruh terhadap performa model yang dimiliki. Salah satu algoritma *machine learning* yang sering digunakan untuk berbagai masalah adalah *Support Vector Machine* (SVM).

Penelitian analisis sentimen sudah dilakukan sebelumnya pada layanan perbankan oleh Sari dan Irhamah (2020). Penelitian tersebut mengklasifikasi data *Twitter* menjadi sentimen positif dan negatif menggunakan representasi kata *Term Frequency Inverse Document Frequency* (TF-IDF) sebagai input algoritma *Naïve Bayes Classifier* (NBC) dan *Support Vector Machine* (SVM) dengan *SMOTE*. Dalam penelitiannya, Mahendrajaya (2019) melakukan analisis sentimen terhadap data *tweet* opini pengguna layanan *Gopay*. Penelitian tersebut menggunakan metode *lexicon based* untuk memberikan label sentimen positif dan negatif pada data. Representasi kata yang digunakan adalah TF-IDF sebagai input algoritma SVM dengan *kernel linear* dan *polinomial* untuk klasifikasinya.

Penerapan metode seleksi fitur dapat dilakukan pada metode klasifikasi untuk meningkatkan performa model dengan mengurangi jumlah fitur yang digunakan. Cahyono (2017) menyatakan bahwa seleksi fitur digunakan untuk mengurangi *set* fitur yang besar menjadi *subset* fitur yang lebih kecil. Seleksi fitur akan mengurangi waktu komputasi dan meningkatkan efisiensi model karena hanya menggunakan fitur yang dianggap relevan atau paling berdampak pada model. Penelitian analisis sentimen menggunakan *Naïve Bayes Classifier* dengan seleksi fitur *Chi-Square* dan *Particle Swarm Optimization* untuk klasifikasi sentimen menjadi positif, netral, dan negatif telah dilakukan oleh Septiana *et al*. (2021) dan memperoleh hasil seleksi fitur *Chi-Square* sebagai metode seleksi fitur terbaik dengan meningkatkan akurasi model dari 63,69% menjadi 69,13%. Kemudian Luthfiana *et al*. (2020) melakukan analisis sentimen untuk tiga kelas yaitu sentimen positif, netral, dan negatif menggunakan metode SVM dan seleksi fitur *Chi-Square*. Penelitian tersebut memperoleh hasil performa model tanpa seleksi fitur dengan akurasi sebesar 69%, *precision* sebesar 48%, *recall* sebesar 53%, dan *F1-Score* sebesar 50%. Setelah diterapkan seleksi fitur, performa model meningkat dengan akurasi sebesar 77%, *precision* sebesar 50%, *recall* sebesar 55%, dan *F1-Score* sebesar 73%.

2 KAJIAN PUSTAKA

2.1 Analisis Sentimen

Menurut Rosdiana *et al*. (2019) analisis sentimen merupakan proses memahami, mengekstrak, dan mengolah data tekstual secara otomatis untuk mendapatkan informasi

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

sentimen yang terkandung dalam suatu kalimat opini. Kemudian, Aditama *et al*. (2022) menyatakan informasi sentimen yang terkandung dalam suatu kalimat opini dapat diklasifikasi menjadi sentimen positif, negatif, dan netral berdasarkan polaritas sebuah teks. Banyak penelitian membatasi pengklasifikasian kalimat opini menjadi sentimen positif dan negatif karena sulit menentukan polaritas teks untuk sentimen netral. Analisis sentimen dapat digunakan untuk melakukan analisis pasar oleh perusahaan, mengetahui opini masyarakat terhadap isu politik, dan perbaikan suatu layanan publik berdasarkan ulasan masyarakat. Analisis sentimen memudahkan klasifikasi sentimen data teks dalam jumlah besar dengan menerapkan metode klasifikasi secara otomatis.

Menurut Ligthart *et al*. (2021), klasifikasi teks untuk analisis sentimen dapat dilakukan dengan pendekatan *machine learning*, *deep learning*, *lexicon-based*, dan *hybrid*. Analisis sentimen dengan pendekatan *machine learning* membutuhkan data latih yang sudah diberi label untuk melatih model.

2.2 *Machine Learning*

Machine learning adalah kategori *Artificial Intelligence* (AI) yang memungkinkan komputer berpikir dan belajar sendiri (Alzubi *et al*., 2018). *Machine learning* bertujuan untuk menangani data lebih efisien. Sering kali, orang dihadapkan pada data dengan jumlah yang besar dan kesulitan menginterpretasi hasil ekstrak dari data yang dimiliki. Oleh karena itu, menggunakan *machine learning* adalah solusi yang tepat untuk permasalahan tersebut. *Machine learning* akan belajar dari data untuk mendapatkan performa model yang maksimal tanpa diprogram secara manual.

Machine learning dapat diklasifikasikan menjadi *supervised learning* dan *unsupervised learning* (Khan *et al*., 2018). *Supervised learning* adalah jenis *machine learning* dimana komputer dilatih menggunakan dataset dengan *input* dan *output* yang telah diketahui, sedangkan *unsupervised learning* adalah jenis *machine learning* dimana komputer dilatih menggunakan *dataset* dengan *output* yang tidak diketahui. Pada penelitian analisis sentimen, jenis *machine learning* yang digunakan adalah *supervised learning* karena *output* dari dataset sudah diketahui berupa label sentimen positif atau negatif. Khan *et al*. (2018) menyatakan bahwa metode *supervised learning* yang paling efisien dan sering digunakan adalah *K-Nearest Neighbors* (KNN), *Support Vector Machine* (SVM), *Large Margin Nearest Neighbor* (LMNN), dan *Extended Nearest Neighbor* (ENN). Mantovani *et al*. (2019) menyatakan bahwa tahap analisis sentimen dengan *machine learning* terdiri dari *text preprocessing*, representasi kata, seleksi fitur, dan tahap klasifikasi.

2.3 *Text Preprocessing*

Text Preprocessing dilakukan dengan menghilangkan *noise* pada teks untuk meningkatkan akurasi dari model *machine learning* (Alam dan Yao, 2019). Penerapan *text preprocessing* yang baik dapat meningkatkan performa model. Putra *et al*. (2020) menyatakan secara umum *text preprocessing* terdiri dari empat tahap berikut:

1. *Case folding* merupakan proses untuk mengubah semua huruf menjadi huruf kecil.
2. *Stopword filtering* merupakan proses untuk menghilangkan kata tanpa makna. Misalnya kata “malah”, “adalah”, “di”, “ke”, dan “yang” akan dihilangkan pada tahap ini.
3. *Tokenizing* merupakan proses untuk memisahkan kalimat menjadi beberapa kata. Biasanya setiap kata akan dipisahkan oleh spasi sehingga pada kasus ini akan digunakan *space delimiter*.
4. *Stemming* merupakan proses untuk mengekstrak atau mengurangi imbuhan dengan tujuan mendapatkan bentuk kata dasar.

HaCohen-Kerner *et al*. (2020) menyatakan bahwa *text preprocessing* yang lebih mutakhir dapat dilakukan dengan mengubah kata singkatan atau kata gaul menjadi kata baku yang memiliki makna yang sama. Tahap *text preprocessing* berkaitan erat

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

dengan seleksi fitur. Tahap ini memastikan data latih ulasan yang dimiliki hanya berisi fitur atau kata yang relevan sebelum masuk ke tahap seleksi fitur.

2.4 Seleksi Fitur *Chi-Square*

Seleksi fitur digunakan untuk meningkatkan performa model dengan mengurangi jumlah fitur yang digunakan agar komputasi yang dilakukan lebih efisien. Salah satu metode seleksi fitur yang sering digunakan adalah metode *Chi-Square*. Seleksi fitur *Chi-Square* bertujuan memilih fitur menggunakan nilai statistika *Chi-Square* untuk mengukur ketergantungan term dengan kelasnya (Thaseen *et al*., 2019). Nilai uji *Chi-Square* setiap *term* akan diurutkan dari yang tertinggi untuk menentukan *term* atau kata yang akan digunakan sebagai fitur. Fungsi dari uji *Chi-Square* sebuah kata terhadap suatu kategori diperoleh dari Persamaan (1) (Suharno *et al*., 2017)

$$\chi^2(t) = \frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)} \quad (1)$$

dengan t adalah kata, c adalah kelas atau kategori, N adalah banyaknya dokumen latih, A adalah banyaknya dokumen pada kelas positif yang memuat t, B adalah banyaknya dokumen bukan kelas positif yang memuat t, C adalah banyaknya dokumen pada kelas positif yang tidak memuat t, dan D adalah banyaknya dokumen bukan kelas positif yang tidak memuat t.

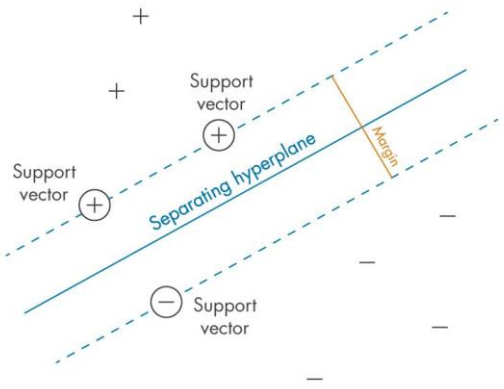
Pemilihan fitur dilakukan dengan mengurutkan nilai statistik $\chi^2(t)$ dan mengambil fitur dengan nilai yang tertinggi. Seleksi fitur *Chi-Square* akan menentukan fitur atau kata yang paling signifikan sebelum tahap representasi kata.

2.5 Representasi Kata

Representasi kata adalah salah satu faktor penting dalam klasifikasi teks. Proses ini dilakukan dengan mentransformasikan data teks menjadi vektor agar dapat diproses oleh mesin. Ada beberapa teknik yang dapat digunakan untuk melakukan proses representasi kata seperti *Bag-of-Words* (BoW), *Term Frequency Inverse Document Frequency* (TF-IDF), dan *N-Grams* (Putra *et al*., 2020). Pada penelitian ini, TF-IDF digunakan untuk representasi kata. Arifin *et al*. (2021) menyatakan bahwa TF-IDF adalah metode yang umumnya dipakai untuk menentukan hubungan kata terhadap dokumen atau kalimat dengan memberikan bobot atau nilai pada masing-masing kata. Vektor TF-IDF yang diperoleh menjadi *input* pada metode klasifikasi SVM.

2.6 *Support Vector Machine* (SVM)

Saraswati (2011) menyatakan bahwa *Support Vector Machine* (SVM) adalah seperangkat metode *supervised learning* yang menganalisis data dan mengenali pola, digunakan untuk klasifikasi dan analisis regresi. Klasifikasi SVM dilakukan dengan mencari *hyperplane* atau garis pembatas (*decision boundary*) yang memisahkan antara satu kelas dengan kelas yang lain. *Hyperplane* terbaik dapat ditentukan dengan mengukur margin *hyperplane* dan mencari titik maksimum (Ariyanto dan Chamidah, 2021). Margin adalah jarak antara *hyperplane* dengan pola terdekat (*support vector*) dari masing-masing kelas. Komponen SVM terdiri atas *optimal hyperplane*, *hyperplane* kelas positif, *hyperplane* kelas negatif, dan *margin*. Ilustrasi SVM dapat dilihat pada Gambar 1.



Sumber: *towardsdatascience.com*
Gambar 1 Ilustrasi SVM

Penerapan metode SVM menghasilkan nilai $\mathbf{w}$ dan b dan dapat dibentuk fungsi $f(x)$

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

dan $sign(f(x))$ untuk menentukan kelas data *input* dengan Persamaan (2) dan (3).

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b, \quad (2)$$

$$sign(f(\mathbf{x})) = \begin{cases} +1, & f(\mathbf{x}) \geq 0 \\ -1, & f(\mathbf{x}) < 0 \end{cases}. \quad (3)$$

2.7 *Confusion Matrix*

Evaluasi dari suatu model klasifikasi diperoleh dari tingkat kebenaran dengan menghitung ukuran statistik yaitu *True Positives* (TP), *True Negatives* (TN), *False Positives* (FP), dan *False Negatives* (FN). Komponen ini akan membentuk suatu *Confusion Matrix* pada Tabel (2.1) (Sari dan Irhamah, 2020). *Confusion Matrix* dapat dibentuk untuk model klasifikasi data biner dengan tujuan menggambarkan performa model.

Tabel 1. *Confusion Matrix.*

	Prediksi: Positif	Prediksi: Negatif
Aktual: Positif	TP	FN
Aktual: Negatif	FP	TN

Empat istilah pada Tabel 1 dijelaskan sebagai berikut:

1. *True Positives* (TP) adalah jumlah data kelas positif yang tepat diprediksi sebagai kelas positif.
2. *True Negatives* (TN) adalah jumlah data kelas negatif yang tepat diprediksi sebagai kelas negatif.
3. *False Positives* (FP) adalah jumlah data kelas negatif yang salah diprediksi menjadi kelas positif.
4. *False Negatives* (FN) adalah jumlah data kelas positif yang salah diprediksi menjadi kelas negatif.

2.8 *Performance Metrics*

Setelah menyelesaikan suatu model klasifikasi, perlu dilakukan pengujian dan evaluasi untuk mengetahui performa model klasifikasi. Hasil evaluasi akan menentukan pengembangan model selanjutnya untuk meningkatkan performa model. Arifin *et al*. (2021) menyatakan bahwa metode pengujian *data mining* yang paling banyak digunakan adalah mencari nilai *precision*, *recall*, *F1-Score* dan *accuracy*. Penjelasan dari masing-masing metrik tersebut sebagai berikut:

1. *Precision*
   Precision adalah rasio perbandingan jumlah data prediksi benar positif dengan keseluruhan data prediksi positif atau dapat dituliskan pada Persamaan (4).

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

2. *Recall*
   Recall adalah rasio perbandingan jumlah data prediksi benar positif dengan jumlah data benar positif dan data salah negatif atau dapat dituliskan pada Persamaan (5).

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

3. *F1-Score*
   F1-Score adalah parameter ukuran keberhasilan *retrieval* yang menggabungkan *precision* dan *recall*. Perhitungan metrik *F1-Score* melibatkan informasi FP dan FN sehingga membuat metrik ini cocok digunakan untuk kasus *imbalanced data*. Nilai dari *F1-Score* diperoleh dari Persamaan (6).

   $$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (6)$$

4. *Accuracy*
   Accuracy adalah rasio prediksi benar dengan keseluruhan data. Perhitungan accuracy diperoleh dari Persamaan (7).

   $$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (7)$$

2.9 *Hyperparameter Tuning*

Elgeldawi *et al.* (2021) menyatakan bahwa model *machine learning* akan otomatis mempelajari dan menyesuaikan parameter internalnya berdasarkan data latih. Parameter ini disebut "*model parameters*" atau dapat disebut juga secara singkat sebagai "*parameters*". Namun, ada parameter lain yang harus dikonfigurasi terlebih dahulu

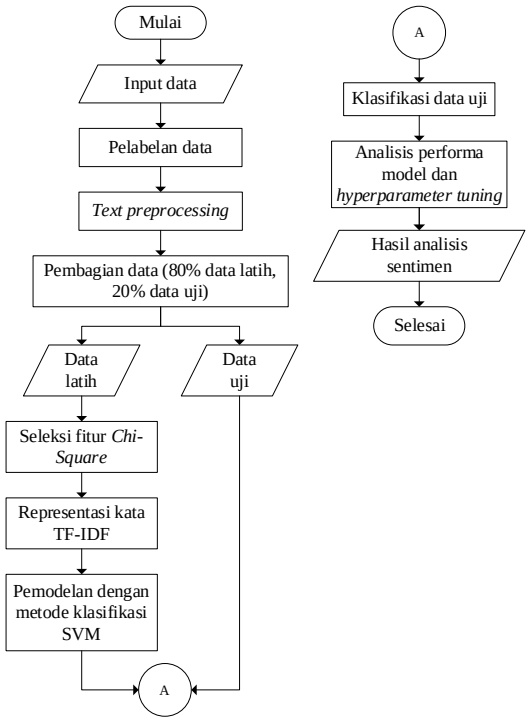
e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

sebelum proses pembelajaran model dimulai dan tidak berubah selama proses pembelajaran model. Parameter ini disebut sebagai "*hyperparameters*". *Model parameters* menunjukkan bagaimana data input diubah menjadi output yang diinginkan, sedangkan *hyperparameters* menunjukkan bagaimana model disusun. Semua *hyperparameter* model klasifikasi akan memengaruhi hasil performa model.

Hyperparameter tuning adalah proses penyesuaian *hyperparameters* dari sebuah model *machine learning* untuk menghasilkan model dengan performa yang lebih baik. Proses ini akan mengambil hasil performa model pada saat ini dengan nilai *hyperparameter* yang sudah diubah dan membandingkannya dengan hasil performa model sebelumnya.

3 METODE PENELITIAN

Metode yang digunakan dalam penelitian ini adalah metode klasifikasi SVM dan seleksi fitur *Chi-Square* dengan diagram alir penelitian ditunjukkan oleh Gambar 2.



Gambar 2 Diagram alir penelitian

4 HASIL DAN PEMBAHASAN

4.1 *Import Library*

Proses penerapan metode SVM dan seleksi fitur *Chi-Square* dengan *Python* memerlukan beberapa *library*. Berikut adalah *script library* yang digunakan.

```
import pandas as pd
import numpy as np
import nltk
import re
import json
from Sastrawi.Stemmer.StemmerFactory
import StemmerFactory
from sklearn.preprocessing import
LabelEncoder
from sklearn.feature_extraction.text
import CountVectorizer, TfidfVectorizer
from sklearn import model_selection, svm
from sklearn.metrics import
confusion_matrix, accuracy_score,
precision_score, recall_score, f1_score
import random
random.seed(42)
```

4.2 *Text Preprocessing*

Data yang telah memiliki label dimasukkan ke *interpreter Python* dengan *script* sebagai berikut.

```
df = pd.read_excel('Data.xlsx')
```

Text preprocessing dilakukan pada data yang telah diberi label. Proses ini melibatkan beberapa langkah, antara lain *case folding* untuk mengubah semua huruf menjadi huruf kecil, *stopword filtering* untuk menghapus kata-kata tidak bermakna, *tokenizing* untuk memisahkan kalimat menjadi kata-kata individual, dan *stemming* untuk mengambil kata dasar dari kata-kata yang memiliki imbuhan. Selain itu, dilakukan juga penggantian singkatan menjadi kepanjangan kata yang relevan serta penghapusan karakter spesial seperti tanda baca atau emoji. Tahap ini juga mencakup penghapusan angka yang terletak di belakang kata, seperti contohnya kata 'masing2' menjadi 'masing'.

Proses penggantian singkatan menjadi kepanjangan kata yang relevan memerlukan *set* kamus kata singkatan yang dapat diperoleh dari internet. Diperukan *user defined function* sebagai berikut untuk proses tersebut.

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

```
def load_slang_words(file_path):
    with open(file_path, 'r') as file:
        slang_data = json.load(file)
    return slang_data

def replace_slang(text, slang_words):
    words = text.split()
    for i in range(len(words)):
        if words[i] in slang_words:
            words[i] =
slang_words[words[i]]
    cleaned_text = ' '.join(words)
    return cleaned_text
```

Setelah itu *text preprocessing* dilakukan dengan *script* sebagai berikut.

```
final data = []
tokenize_text = []

factory = StemmerFactory()
stemmer = factory.create_stemmer()

with open('stop_words.txt', 'r') as file:
    stop_words = [line.strip() for line in
file]

slang_file_path = 'slang_words.json'
slang_words =
load_slang_words(slang_file_path)

for text in df['content']:
    text = re.sub("[^a-zA-Z]"," ",text)
    text = replace_slang(text.lower(),
slang_words)
    text =
nltk.word_tokenize(text.lower())
    text = [stemmer.stem(word) for word in
text]
    text = [word for word in text if word
not in stop_words]
    text = [re.sub(r'(?<=[a-z])\d+\b', '',
word) for word in text]
    tokenize_text.append(str(text))
    text = " ".join(text)
    final_data.append(text)

df_cleaned =
pd.DataFrame(list(zip(df['content'],final_
data,tokenize_text,df['Sentiment'])),colum
ns=['original_content','content','final_te
xt','label'])
```

Data yang telah melewati tahap *text preprocessing* akan dibagi menjadi data latih dan data uji serta dilakukan *label encoding* dengan *script* sebagai berikut.

```
Train_X, Test_X, Train_Y, Test_Y =
model_selection.train_test_split(df_cleane
d['final_text'],df_cleaned['label'],
random_state=42,test_size=0.2)

Encoder = LabelEncoder()
Train_Y = Encoder.fit_transform(Train_Y)
Test_Y = Encoder.fit_transform(Test_Y)
```

4.3 Seleksi Fitur *Chi-Square*

Proses seleksi fitur *Chi-Square* dilakukan dengan menghitung nilai *Chi-Square* semua kata unik pada data latih dan dipilih sebanyak *k* kata dengan nilai *Chi-Square* tertinggi. Pada penelitian ini dipilih *k* sebanyak 1000 kata. Fitur atau kata yang terpilih diaplikasikan juga pada data uji. Proses ini dilakukan dengan *script* berikut.

```
def calculate_chi2(word, Train_X, Train_Y,
count_vect):
    X_train_counts =
count_vect.transform(Train_X)
    word_index =
count_vect.vocabulary_.get(word)
    b = np.sum((X_train_counts[Train_Y ==
1][:, word_index] > 0), dtype=np.float64)
    a = np.sum((X_train_counts[Train_Y ==
0][:, word_index] > 0), dtype=np.float64)
    d = np.sum((X_train_counts[Train_Y ==
1][:, word_index] == 0), dtype=np.float64)
    c = np.sum((X_train_counts[Train_Y ==
0][:, word_index] == 0), dtype=np.float64)
    n = len(Train_Y)
    chi2 = (n * (a * d - b * c)**2) / ((a
+ b) * (c + d) * (a + c) * (b + d))
    return a, b, c, d, chi2

count_vect = CountVectorizer()
X_train_counts =
count_vect.fit_transform(Train_X)
word_list = count_vect.get_feature_names()

results = []
for word in word_list:
    result = calculate_chi2(word, Train_X,
Train_Y, count_vect)
    results.append(result)

columns = ["A", "B", "C", "D", "Chi2"]
df_chi2 = pd.DataFrame(results,
columns=columns)
df_chi2.insert(0, "word", word_list)
df_chi2 = df_chi2.sort_values(by='Chi2',
ascending=False)

k = 1000
top_k_words = df_chi2['word'][:k].tolist()
vocabulary = {}
for i, word in enumerate(top_k_words):
    vocabulary[word] = i

count_vect_selected =
CountVectorizer(vocabulary=top_k_words)
X_train_selected =
count_vect_selected.fit_transform(Train_X)
df_train_selected =
pd.DataFrame(X_train_selected.toarray(),
columns=top_k_words)
X_test_selected =
count_vect_selected.transform(Test_X)
df_test_selected =
pd.DataFrame(X_test_selected.toarray(),
columns=top_k_words)
```

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

4.4 Representasi Kata

Fitur atau kata yang telah dipilih dari tahap seleksi fitur digunakan sebagai parameter representasi kata TF-IDF. Elemen vektor TF-IDF yang dihasilkan sama dengan banyaknya fitur yang dipilih atau sebanyak *k* elemen. Penerapan representasi kata TF-IDF dilakukan dengan *script* berikut.

```
vectorizer =
TfidfVectorizer(vocabulary=vocabulary)
X_train_tfidf =
vectorizer.fit_transform(Train_X)
X test tfidf =
vectorizer.transform(Test_X)
```

4.5 Klasifikasi dengan SVM

Vektor TF-IDF yang diperoleh menjadi vektor input **x** pada metode klasifikasi SVM. Pemodelan dengan metode SVM dilakukan dengan *script* berikut.

```
SVM = svm.SVC(kernel='linear')
SVM.fit(X_train_tfidf, Train_Y)
```

Parameter pada fungsi svm.SVC() disebut dengan *hyperparameter*. Jika *hyperparameter* tidak dideklarasikan, maka nilai yang digunakan adalah nilai *default*. Setelah memperoleh model SVM, dapat dilakukan klasifikasi data uji dengan *script* berikut.

```
SVM = svm.SVC(kernel='linear')
SVM.fit(X_train_tfidf, Train_Y)

predictions_SVM =
SVM.predict(X_test_tfidf)
result_predict = pd.DataFrame(Test_X)
result_predict['actual_label'] = Test_Y
result_predict['predict_label'] =
predictions_SVM
```

4.6 Evaluasi dan *Hyperparameter Tuning*

Performa model dapat ditingkatkan dengan melakukan *hyperparameter tuning*. Sebelum melakukan *hyperparameter tuning*, perlu dilakukan pemilihan *performance metric* yang dijadikan acuan. Sebagai ilustrasi, jika dataset yang digunakan *imbalance* maka *performance metric* yang cocok digunakan adalah *F1-Score*. Pencarian semua nilai *performance metrics* tetap perlu dilakukan untuk mendapatkan gambaran kinerja model. Proses *hyperparameter tuning* dilakukan dengan *script* berikut.

```
results_df = pd.DataFrame(columns=['C',
'kernel', 'TP', 'TN', 'FP', 'FN',
'Accuracy', 'Precision', 'Recall', 'F1-
Score'])

C_values = [0.1, 1, 10, 100]

for C in C_values:
    SVM = svm.SVC(C=C, kernel='linear')
    SVM.fit(X_train_tfidf, Train_Y)
    predictions =
SVM.predict(X_test_tfidf)

    tn, fp, fn, tp =
confusion_matrix(Test_Y,
predictions).ravel()
    accuracy = accuracy_score(Test_Y,
predictions)
    precision = precision_score(Test_Y,
predictions)
    recall = recall_score(Test_Y,
predictions)
    f1 = f1_score(Test_Y, predictions)

    results_df = results_df.append({
        'C': C,
        'kernel': 'linear',
        'TP': tp,
        'TN': tn,
        'FP': fp,
        'FN': fn,
        'Accuracy': accuracy,
        'Precision': precision,
        'Recall': recall,
        'F1-Score': f1
    }, ignore_index=True)
```

Script diatas menghasilkan tabel yang berisi nilai *hyperparameter* yang diubah dan *performance metrics* yang diperoleh. Pemilihan model terbaik dilakukan dengan melihat *hyperparameter* acuan tertinggi.

5 SIMPULAN

Berdasarkan penelitian yang dilakukan, dapat disimpulkan bahwa analisis sentimen dapat dilakukan melalui klasifikasi teks menggunakan metode *machine learning* yaitu *supervised learning*, khususnya SVM, yang dipadukan dengan seleksi fitur *Chi-Square* dengan menggunakan bahasa pemrograman *Python*. Penelitian selanjutnya disarankan untuk melakukan lebih banyak kombinasi *hyperparameter tuning*.

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

DAFTAR PUSTAKA

Aditama, M. I., Pratama, R. I., Wiwaha, K. H. U., & Rakhmawati, N. A. (2022). Analisis Klasifikasi Sentimen Pengguna Media Sosial Twitter Terhadap Pengadaan Vaksin COVID-19. *Journal Information Engineering and Educational Technology) ISSN*, *2549*, 869X.

Alam, S., & Yao, N. (2019). The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis. *Computational and Mathematical Organization Theory*, *25*, 319–335.

Alzubi, J., Nayyar, A., & Kumar, A. (2018). Machine learning from theory to algorithms: an overview. *Journal of Physics: Conference Series*, *1142*(1), 12012.

Arifin, N., Enri, U., & Sulistiyowati, N. (2021). Penerapan Algoritma *Support Vector Machine* (SVM) dengan TF-IDF *N-Gram* untuk *Text Classification*. *STRING (Satuan Tulisan Riset Dan Inovasi Teknologi)*, *6*(2), 129–136.

Ariyanto, R. A., & Chamidah, N. (2021). Sentiment Analysis for Zoning System Admission Policy Using Support Vector Machine and Naive Bayes Methods. *Journal of Physics: Conference Series*, *1776*(1), 12058.

Cahyono, Y., & UNPAM, T. I. (2017). Analisis Sentiment pada Sosial Media *Twitter* Menggunakan *Naïve Bayes Classifier* dengan *Feature Selection Particle Swarm Optimization* dan *Term Frequency*. *METODE*, *81*, 67.

Elgeldawi, E., Sayed, A., Galal, A. R., & Zaki, A. M. (2021). Hyperparameter tuning for machine learning algorithms used for arabic sentiment analysis. *Informatics*, *8*(4), 79.

HaCohen-Kerner, Y., Miller, D., & Yigal, Y. (2020). The influence of preprocessing on text classification using a bag-of-words representation. *PloS One*, *15*(5), e0232525.

Khan, M. M. R., Arif, R. B., Siddique, M. A. B., & Oishe, M. R. (2018). Study and observation of the variation of accuracies of KNN, SVM, LMNN, ENN algorithms on eleven different datasets from UCI machine learning repository. *2018 4th International Conference on Electrical Engineering and Information & Communication Technology (ICEEiCT)*, 124–129.

Ligthart, A., Catal, C., & Tekinerdogan, B. (2021). Systematic reviews in sentiment analysis: a tertiary study. *Artificial Intelligence Review*, 1–57.

Luthfiana, L., Young, J. C., & Rusli, A. (2020). Implementasi Algoritma *Support Vector Machine* dan *Chi Squar*e untuk Analisis Sentimen *User Feedback* Aplikasi. *Ultimatics: Jurnal Teknik Informatika*, *12*(2), 125–126.

Mahendrajaya, R., Buntoro, G. A., & Setyawan, M. B. (2019). Analisis Sentimen Pengguna Gopay Menggunakan Metode *Lexicon Based* Dan *Support Vector Machine*. *KOMPUTEK*, *3*(2), 52–63.

Mantovani, R. G., Rossi, A. L. D., Alcobaça, E., Vanschoren, J., & de Carvalho, A. C. (2019). A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves SVM classifiers. *Information Sciences*, *501*, 193–221.

Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, *5*(4), 1093–1113.

Nasukawa, T., & Yi, J. (2003). Sentiment analysis: Capturing favorability using natural language processing. *Proceedings of the 2nd International Conference on Knowledge Capture*, 70–77.

Putra, O. V., Wasmanson, F. M., Harmini, T., & Utama, S. N. (2020). Sundanese twitter dataset for emotion classification. *2020 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*, 391–395.

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

Rosdiana, R., Eddy, T., Zawiyah, S., & Muhammad, N. Y. U. (2019). Analisis Sentimen pada Twitter terhadap Pelayanan Pemerintah Kota Makassar. *Seminar Nasional Teknik Elektro Dan Informatika*, 87–93.

Şahin, D. Ö., & Klç, E. (2019). Two new feature selection metrics for text classification. *Automatika: Časopis Za Automatiku, Mjerenje, Elektroniku, Računarstvo i Komunikacije*, *60*(2), 162–171.

Saraswati, N. W. S. (2011). Text mining dengan metode na\"\ive bayes classifier dan support vector machines untuk sentiment analysis. *Universitas UDAYANA, Teknik Elektro. Denpasar: Universitas UDAYANA*.

Sari, E. D. N., & Irhamah, I. (2020). Analisis Sentimen Nasabah Pada Layanan Perbankan Menggunakan Metode Regresi Logistik Biner, Na\"\ive Bayes Classifier (NBC), dan Support Vector Machine (SVM). *Jurnal Sains Dan Seni ITS*, *8*(2), D177–D184.

Septiana, R. D., Susanto, A. B., & Tukiyat, T. (2021). Analisis Sentimen Vaksinasi Covid-19 Pada Twitter Menggunakan Naive Bayes Classifier Dengan Feature Selection Chi-Squared Statistic dan Particle Swarm Optimization. *Jurnal SISKOM-KB (Sistem Komputer Dan Kecerdasan Buatan)*, *5*(1), 49–56.

Shaik, T., Tao, X., Dann, C., Xie, H., Li, Y., & Galligan, L. (2022). Sentiment analysis and opinion mining on educational data: A survey. *Natural Language Processing Journal*, 100003.

Suharno, C. F., Fauzi, M. A., & Perdana, R. S. (2017). Klasifikasi Teks Bahasa Indonesia Pada Dokumen Pengaduan Sambat Online Menggunakan Metode K-Nearest Neighbors Dan Chi-square. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer E-ISSN*, *2548*, 964X.

Thaseen, I. S., Kumar, C. A., & Ahmad, A. (2019). Integrated intrusion detection model using chi-square feature selection and ensemble of classifiers. *Arabian Journal for Science and Engineering*, *44*, 3357–3368.

Uysal, A. K., & Gunal, S. (2014). The impact of preprocessing on text classification. *Information Processing & Management*, *50*(1), 104–112.