

**FINAL PROJECT**  
**PENGANTAR PEMROSESAN DATA MULTIMEDIA**



**Kelas B**  
**Anggota Kelompok:**

**Gede krisnawa sandhya wandhana (2108561017)**  
**I Kadek Widiarthawan (2108561092)**  
**Anak Agung Ngurah Frady Cakranegara (2108561097)**

**Dosen Pengampu:**  
**Dr. Anak Agung Istri Ngurah Eka Karyawati, S.Si., M.Eng.**

**PROGRAM STUDI INFORMATIKA**  
**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM UNIVERSITAS**  
**UDAYANA**  
**JIMBARAN**  
**2023**

# **BAB I**

## **PENDAHULUN**

### **1.1 Latar Belakang**

Dalam era globalisasi saat ini, perkembangan teknologi telah mencapai tingkat yang sangat cepat. Kemajuan ini didorong oleh penemuan beberapa teknologi baru oleh para ahli. Salah satu kemajuan teknologi yang signifikan adalah Text Preprocessing. Teknik ini memungkinkan kita untuk menganalisis sentimen pada suatu kalimat, yaitu mengkategorikan apakah kalimat tersebut bersifat positif atau negatif.

Text Preprocessing umumnya digunakan dalam beberapa tugas pada Text Mining, Natural Language Processing (NLP), dan Information Retrieval (IR). Dalam bidang Text Mining, preprocessing digunakan untuk mengambil informasi yang penting dan menarik dari kumpulan data teks yang tidak terstruktur. Sedangkan dalam Information Retrieval (IR), preprocessing digunakan untuk memutuskan dokumen mana yang harus dipilih untuk memenuhi kebutuhan informasi pengguna. Keputusan ini dibuat dengan membandingkan istilah dalam dokumen dengan istilah indeks yang muncul. Tahapan preprocessing terdiri dari Case Folding, Tokenizing, Filtering, dan Stemming.

Namun, melakukan preprocessing secara manual membutuhkan waktu yang lama dan berulang, karena harus memeriksa setiap kata dalam kalimat yang akan diproses. Selain itu, penulisan ulang contoh kalimat pada setiap tahapan preprocessing juga dapat memakan waktu dan membingungkan saat menganalisis teks.

Untuk mengatasi kendala tersebut, dalam Final Project ini dibuat program menggunakan bahasa pemrograman Python untuk membantu melakukan text preprocessing. Kalimat akan melewati beberapa tahapan seperti Case Folding, Tokenizing, Filtering, dan Stemming. Output yang diharapkan adalah klasifikasi kalimat sebagai positif atau negatif.

Dengan menggunakan program ini, diharapkan proses preprocessing menjadi lebih efisien dan akurat. Selain itu, tidak perlu melakukan penulisan ulang pada setiap tahapan preprocessing, sehingga memudahkan analisis teks.

### **1.2 Rumusan Masalah**

Adapun rumusan masalah dari tugas ini adalah sebagai berikut:

1. Bagaimana langkah-langkah untuk melakukan text preprocessing dengan menggunakan metode pembobotan TF-IDF dan metode Support Vector Machine (SVM)?

2. Apa faktor-faktor yang mempengaruhi tingkat akurasi pada Chi-Square Feature Selection, dan bagaimana menentukan tingkat akurasi yang optimal?

### **1.3 Tujuan**

Adapun tujuan dari tugas ini adalah sebagai berikut :

1. Bagaimana langkah-langkah yang dapat digunakan untuk mengkategorikan suatu kalimat apakah bersifat positif atau negatif?
2. Bagaimana cara mengimplementasikan metode Support Vector Machine (SVM) dalam proses Text Preprocessing untuk analisis sentimen?

### **1.4 Manfaat**

Adapun manfaat dari tugas ini adalah sebagai berikut :

1. Bagaimana text preprocessing dapat meningkatkan pemahaman pembaca dan penulis terhadap data teks yang tidak terstruktur?
2. Apa yang perlu dipahami tentang pembobotan TF-IDF dan bagaimana metode Support Vector Machine (SVM) dapat digunakan dalam kombinasi dengan TF-IDF untuk analisis teks yang lebih efektif?

## BAB II

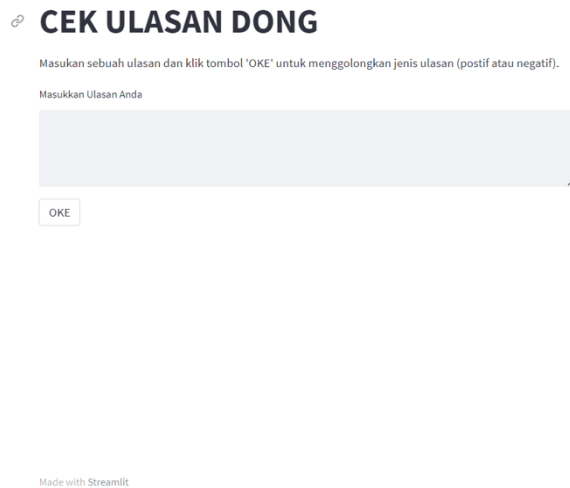
### ISI

#### 2.1. Manual Aplikasi

##### 2.1.1 Fitur Sistem

User bisa memasukkan/mengetikan/menuliskan kalimat review kemudian mengklik tombol Classify untuk melihat review tersebut termasuk positif atau negative.

##### 2.2.2 Antar Muka



[CEK ULASAN DONG](#)

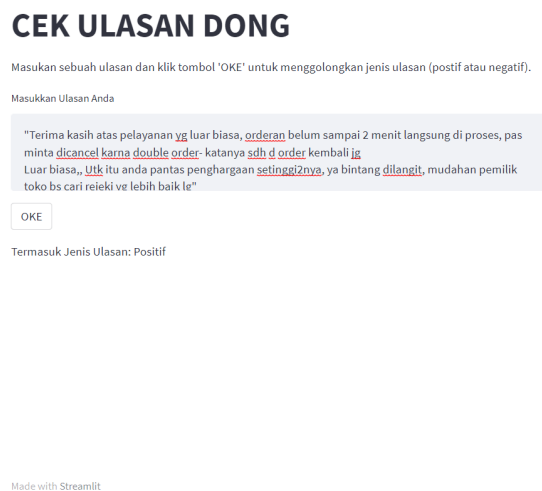
Masukan sebuah ulasan dan klik tombol 'OKE' untuk menggolongkan jenis ulasan (positif atau negatif).

Masukkan Ulasan Anda

OKE

Made with Streamlit

Hasil prediksi klasifikasi atau sentimen review akan ditampilkan di bawah tombol OKE



**CEK ULASAN DONG**

Masukan sebuah ulasan dan klik tombol 'OKE' untuk menggolongkan jenis ulasan (positif atau negatif).

Masukkan Ulasan Anda

"Terima kasih atas pelayanan yg luar biasa, orderan belum sampai 2 menit langsung di proses, pas minta dicancel karna double order- katanya sdh d order kembali lg. Luar biasa,, Utk itu anda pantas penghargaan setinggi2nya, ya bintang dilangit, mudahan pemilik toko bs cari reieki ve lebih baik le"

OKE

Termasuk Jenis Ulasan: Positif

Made with Streamlit

## CEK ULASAN DONG

Masukan sebuah ulasan dan klik tombol 'OKE' untuk menggolongkan jenis ulasan (postif atau negatif).

Masukkan Ulasan Anda

"setiap belanja selalu ada saj belanja selalu ada saja barang yg tidak dikirim  
tidak jujur  
Kadangana sempat vidio unboxing karna toko lain jujur semua saya pikir toko ini jujur juga ternyata  
tidak iuiur sama sekali

OKE

Termasuk Jenis Ulasan: Negatif

Made with Streamlit

## CEK ULASAN DONG

Masukan sebuah ulasan dan klik tombol 'OKE' untuk menggolongkan jenis ulasan (postif atau negatif).

Masukkan Ulasan Anda

OKE

Silakan masukan sebuah ulasan.

Made with Streamlit

## 2.2. Source Code

Implementasi kode menggunakan bahasa python. Terdapat dua file, yaitu model.ipynb dan app.py, yang kegunaan dan penjelasannya dijelaskan di bawah.

### 2.2.1 Model.ipynb

File kode untuk membuat dan mencari model machine learning SVM terbaik

- Mounting Google Drive

Mounting google drive agar bisa mengakses file yang ada di dalamnya dan juga bisa menyimpan ke dalam google drive

```
[47] from google.colab import drive
drive.mount('/content/drive')
!cd /content/drive/MyDrive/Colab-Notebooks/PPDM/

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
/content/drive/MyDrive/Colab-Notebooks/PPDM
```

- Library

Berikut beberapa library dan/atau modul yang digunakan untuk membuat model machine learning SVM

```

Library

from flask import Flask, render_template, url_for
import numpy as np
import pandas as pd
import csv
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import svm
from sklearn.metrics import accuracy_score

# Packages for visuals
import matplotlib.pyplot as plt
import seaborn as sns; sns.set(font_scale=1.2)

import re
!pip install Sastrawi
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
from nltk.tokenize import word_tokenize
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

Requirement already satisfied: Sastrawi in /usr/local/lib/python3.10/dist-packages (1.0.1)

```

- Import Data

Data berupa file csv disimpan ke dalam variabel “review” dan ditampilkan. Data berupa kumpulan ulasan yang berisi label 1=positif atau 0= negative

Import Data

```
reviews = pd.read_csv("reviews.csv")
```

reviews

No	Reviews	Label
0	1 keraja nya bagusss bgittttt🤔🤔🤔aaaa mauuu ringiss...	1
1	2 Jahitannya sih rapi,cuman ada benang yang ikut...	0
2	3 Sesuai harga. Agak tipis tapi masih oke kok. W...	0
3	4 Wah gila siihhh sebgus itu, se worth it, se l...	1
4	5 Kain nya bagus halus \nTapi kok di bukak koto...	0
...	...	...
826	827 Terima kasih barang sudah sampai sesuai ukuran...	1
827	828 Mantapp realpittt bangittt tapi pengemasan nya...	1
828	829 Suka bgt sama tasnya, ga kayak tas local. Kere...	1
829	830 kualitas produk sangat baik. produk original. ...	1
830	831 Barang udah sampai dg selamat, mantul banget d...	1

831 rows x 3 columns

- Preprocessing

Preprocessing untuk menyiapkan data sebelum diolah, yang meliputi tahap tokenization, lower case converting, stop word removal, dan stemming. Lalu data hasil preprocessing ditampilkan sehingga terlihat perbedaannya dari sebelumnya.

```
[50] factory = StemmerFactory()
      stemmer = factory.create_stemmer()

      factory = StopWordRemoverFactory()
      stopword = factory.create_stop_word_remover()

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

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: x.lower())

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: x.replace('\n', ''))

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: re.compile('[a-zA-Z]').sub(" ", x))

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: x.split())

      def removeAkhir(s):
          cadangan = s[-1]
          s = s.rstrip(s[-1])

          return s + cadangan

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: (removeAkhir(i) for i in x))

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: [i for i in x if len(i) > 2])

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: " ".join(x))

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: stopword.remove(x))

      reviews['Reviews'] = reviews['Reviews'].apply(lambda x: stemmer.stem(x))

reviews

[52] # reviews.to_csv(r"/content/drive/MyDrive/Colab-Notebooks/final-project/preprocessed-data.csv")
      # reviews = pd.read_csv('preprocessed-data.csv')
```

- Membagi Data Menjadi Train dan Test
- Membagi data menjadi data test sebanyak 20% dan train menjadi 80%

Split the data into training and test sets

```

[53] train_X, test_X, train_Y, test_Y = model_selection.train_test_split(reviews['Reviews'], reviews['Label'], test_size = 0.2, random_state = 0)

[54] df_train80 = pd.DataFrame()
df_train80['Reviews'] = train_X
df_train80['Label'] = train_Y

df_test20 = pd.DataFrame()
df_test20['Reviews'] = test_X
df_test20['Label'] = test_Y

```

df\_train80

	Reviews	Label
759	bahan adembagus pake ukur xxi alhamdulillah pas	1
752	barang sdh darat kualitasbagus kirim sesuai es...	1
249	bagus bangggeeemtharga murahhcuma setting dul...	1
363	kecewa banget	0
451	kualitas produk bagus sih saya sedikit kecewa ...	0
...	...	...
763	kualitas produk baik harga sangat baik cepat k...	1
192	bagus cuma kirim lama bagus	1
629	!!!!!! jelek bgt tipis sesuai sama gambar b...	0
559	warna beda jauh digmbr jatuh nya wmanya gelap	0
684	alhamdulillah udah sampesuka bahan adem jatuh ...	1

664 rows x 2 columns

df\_test20

	Reviews	Label
320	jelek banget padahal gambar bagus	0
471	selalu langgan banget sini gapernah kecewa sam...	1
235	syg banget maroon tombol patah tau patah kmnay...	0
419	mantap gak pernah kecewa poka rekomendasi poko...	1
8	kecewa sih karna pesan warna fuschia malah dat...	0
...	...	...
499	mewah lihat kayak tas mahal bahan kokoh gak ka...	1
778	alhamdulillah pesan barang bagus spt fre kaos ...	1
767	sepatu pas bagus suka kirim cepat	1
575	bagus cuma pengemasan lama pake banget sampe...	0
764	kualitas produk sangat baik produk original ha...	1

167 rows x 2 columns

- Feature Extraction

## Pembobotan TF-IDF

Feature Extraction

TF-IDF

```

[11] from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vect_8020 = TfidfVectorizer(max_features = 5000)
tfidf_vect_8020.fit(reviews['Reviews'])
train_X_tfidf_8020 = tfidf_vect_8020.transform(df_train80['Reviews'])
test_X_tfidf_8020 = tfidf_vect_8020.transform(df_test20['Reviews'])

```

- Feature Selection

Feature Selection dengan menggunakan formula Chi-Square dengan menggunakan beberapa variasi jumlah fitur yang dipertahankan (10% atau 20% atau 30%, dsb). Pada model ini Chi-Square yang digunakan adalah 10% atau 0.1

Feature Selection

Chi-Square = [10%, 20%, 30%]

```

from sklearn.feature_selection import SelectKBest, chi2

percent_kept = 0.1 # Ubah value ini sesuai keinginan
num_features = int(percent_kept * train_X_tfidf_8020.shape[1])
selector = SelectKBest(chi2, k=num_features)
train_X_tfidf_8020_selected = selector.fit_transform(train_X_tfidf_8020, train_Y)
test_X_tfidf_8020_selected = selector.transform(test_X_tfidf_8020)

```



- Proses Pelatihan

Dilakukan proses pelatihan (training) dengan Chi Square = 0.1, parameter c = 10, kernel rbf, gamma = 0.1. Pada gambar di bawah ditampilkan beberapa kombinasi chi square dan parameter terbaik dari data yang ada.

```

- Proses Pelatihan

Chi-square = 0.1, 0.2, atau 0.3 C= 0.1, 1, 10, atau 100, nilai gamma = 0.0001, 0.001, 0.1, atau 1, dan fungsi kernel: rbf atau polynomial

Chi Square = 0.1 Parameter: c=10, kernel=rbf, gamma=0.1 Akurasi: 93.4
Chi Square = 0.1 Parameter: c=100, kernel=rbf, gamma=0.1 Akurasi: 93.4
Chi Square = 0.2 Parameter: c=100, kernel=rbf, gamma=0.1 Akurasi: 93.4
Chi Square = 0.3 Parameter: c=10, kernel=rbf, gamma=0.1 Akurasi: 93.4
Chi Square = 0.3 Parameter: c=100, kernel=rbf, gamma=0.1 Akurasi: 93.4

[54] from sklearn.svm import SVC

model = SVC(C=10, kernel='rbf', gamma=0.1)
model.fit(train_X_tfidf_8020, train_Y)

```

- Proses Pengujian

Setelah proses training dilakukan, selanjutnya proses testing terhadap model untuk melihat berapa akurasinya

```

- Proses Pengujian

[55] from sklearn.metrics import accuracy_score

predictions_SVM_8020 = model.predict(test_X_tfidf_8020)
test_prediction_8020 = pd.DataFrame()
test_prediction_8020['Reviews'] = test_X
test_prediction_8020['Label'] = predictions_SVM_8020
SVM_accuracy_8020 = accuracy_score(predictions_SVM_8020, test_Y)*100
SVM_accuracy_8020 = round(SVM_accuracy_8020,1)

[56] #test_prediction_8020

[57] #test_prediction_8020.to_csv("test_prediction_8020.csv")

[63] SVM_accuracy_8020

93.4

```

- Accuracy, Precision, Recall, F-1 Score

Ditampilkan nilai accuracy, precision, recall, f-1 score dari model machine learning

```

- Accuracy, Precision, Recall, f1-score

[58] from sklearn.metrics import classification_report

print ("\nHere is the classification report:")
print (classification_report(test_Y, predictions_SVM_8020, zero_division='warn'))

Here is the classification report:

```

	precision	recall	f1-score	support
0	0.94	0.93	0.94	91
1	0.92	0.93	0.93	76
accuracy			0.93	167
macro avg	0.93	0.93	0.93	167
weighted avg	0.93	0.93	0.93	167

- Menyimpan Model dan Vectorizer yang sudah dilatih  
Model yang sudah dibuat dan di-training kemudian disimpan dalam bentuk .pkl untuk digunakan dalam pembuatan web, setiap kali web digunakan tidak perlu melakukan tahap training.

#### ▾ Save the trained model and vectorizer

```
[61] import joblib
      joblib.dump(model, "model.pkl")
      joblib.dump(tfidf_vect_8020, "vectorizer.pkl")

['vectorizer.pkl']
```

- Mencari Kombinasi Parameter terbaik

Pada proses ini merupakan iterasi proses seleksi fitur, training, dan testing untuk mencari kombinasi chi square dan parameter (c, kernel, gamma) terbaik. Kemudian menghasilkan output kombinasi dan akurasi.

#### ▾ Mencari Parameter Terbaik

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.feature_selection import SelectKBest, chi2

def mulai(h, i, j, k):

    percent_kept = h # Change this value to the desired percentage of features to keep
    num_features = int(percent_kept * train_X_tfidf_8020.shape[1])
    selector = SelectKBest(chi2, k=num_features)
    train_X_tfidf_8020_selected = selector.fit_transform(train_X_tfidf_8020, train_Y)
    test_X_tfidf_8020_selected = selector.transform(test_X_tfidf_8020)

    print(f"Chi Square = {h}\tParameter: c={i}, kernel={j}, gamma={k}")
    model = SVC(C=i, kernel=j, gamma=k)
    model.fit(train_X_tfidf_8020_selected, train_Y)

    predictions_SVM_8020 = model.predict(test_X_tfidf_8020_selected)
    test_prediction_8020 = pd.DataFrame()
    test_prediction_8020['Reviews'] = test_X_tfidf_8020_selected['Reviews']
    test_prediction_8020['Label'] = predictions_SVM_8020
    SVM_accuracy_8020 = accuracy_score(predictions_SVM_8020, test_Y)*100
    SVM_accuracy_8020 = round(SVM_accuracy_8020,1)

    print(f"Akurasi: {SVM_accuracy_8020}")

    print("Here is the classification report:")
    print(classification_report(test_Y, predictions_SVM_8020, zero_division=0.0), end="\n")
    print("-----")

    chi = [0.1, 0.2, 0.3]
    c = [0.1, 1, 10, 100]
    kernel = ['rbf', 'poly']
    gamma = [0.0001, 0.001, 0.1, 1, 'scale', 'auto']

    for h in chi:
        for i in c:
            for j in kernel:
                for k in gamma:
                    mulai(h, i, j, k)
```

```

Chi Square = 0.1      Parameter: c=0.1, kernel=rbf, gamma=0.0001
Akurasi: 54.5
Here is the classification report:
      precision    recall  f1-score   support

     0       0.54       1.00       0.71        91
     1       0.00       0.00       0.00        76

 accuracy          0.54        167
 macro avg          0.27        167
 weighted avg       0.30        167

=====
Chi Square = 0.1      Parameter: c=0.1, kernel=rbf, gamma=0.001
Akurasi: 54.5
- - - - -

Chi Square = 0.2      Parameter: c=10, kernel=rbf, gamma=0.001
Akurasi: 54.5
Here is the classification report:
      precision    recall  f1-score   support

     0       0.54       1.00       0.71        91
     1       0.00       0.00       0.00        76

 accuracy          0.54        167
 macro avg          0.27        167
 weighted avg       0.30        167

=====
Chi Square = 0.2      Parameter: c=10, kernel=rbf, gamma=0.1
Akurasi: 93.4

```

```

All Combinations:
Chi Square    C Kernel  Gamma  Accuracy
0      0.2    0.1   rbf    0.0001   54.5
1      0.2    0.1   rbf    0.001    54.5
2      0.2    0.1   rbf    0.1      54.5
3      0.2    0.1   rbf    1        68.3
4      0.2    0.1   rbf   scale   61.1
5      0.2    0.1   rbf   auto    54.5
6      0.2    0.1   poly  0.0001   54.5
7      0.2    0.1   poly  0.001    54.5
8      0.2    0.1   poly   0.1     54.5
9      0.2    0.1   poly   1       54.5
10     0.2    0.1   poly  scale   56.3
11     0.2    0.1   poly   auto    54.5
12     0.2    1.0   rbf   0.0001   54.5
13     0.2    1.0   rbf   0.001    54.5
14     0.2    1.0   rbf   0.1     90.4
15     0.2    1.0   rbf    1      94.6
16     0.2    1.0   rbf  scale   94.6
17     0.2    1.0   rbf   auto    54.5
18     0.2    1.0   poly  0.0001   54.5
19     0.2    1.0   poly  0.001    54.5
20     0.2    1.0   poly   0.1     54.5
21     0.2    1.0   poly    1     57.5
22     0.2    1.0   poly  scale   70.7
23     0.2    1.0   poly   auto    54.5
24     0.2   10.0   rbf   0.0001   54.5
25     0.2   10.0   rbf   0.001    54.5
26     0.2   10.0   rbf    0.1    93.4
27     0.2   10.0   rbf    1     89.8
28     0.2   10.0   rbf  scale   91.0
29     0.2   10.0   rbf   auto    55.1
30     0.2   10.0   poly  0.0001   54.5
31     0.2   10.0   poly  0.001    54.5
32     0.2   10.0   poly   0.1    54.5
33     0.2   10.0   poly    1    73.7
34     0.2   10.0   poly  scale   83.2
35     0.2   10.0   poly   auto    54.5
36     0.2  100.0   rbf   0.0001   54.5
37     0.2  100.0   rbf   0.001    92.2
38     0.2  100.0   rbf    0.1    90.4
39     0.2  100.0   rbf    1     88.0
40     0.2  100.0   rbf  scale   89.2
41     0.2  100.0   rbf   auto    94.6
42     0.2  100.0   poly  0.0001   54.5

```

Accuracy Counts:		
	Accuracy	Count
1	94.6	3
9	93.4	1
4	92.2	1
5	91.0	1
2	90.4	2
3	89.8	2
6	89.2	1
7	88.0	1
8	85.0	1
17	83.2	1
10	73.7	1
11	70.7	1
12	68.3	1
13	61.1	1
14	57.5	1
15	56.3	1
16	55.1	1
0	54.5	27

```

Worst Combination:
Chi Square      C Kernel Gamma  Accuracy
47      0.2  100.0   poly  auto    54.5

Moderate Combination:
Chi Square      0.2
C              10.0
Kernel         poly
Gamma          0.1
Accuracy       54.5
Name: 32, dtype: object

Best Combination:
Chi Square      C Kernel Gamma  Accuracy
15      0.2    1.0   rbf     1    94.6

```

Berikut tabel akurasi dari beberapa kombinasi parameter:

1. Akurasi model SVM dengan **Chi-Square 10%**:

	Salah satu Kombinasi (C, kernel, gamma)	Akurasi	Jumlah kombinasi
Terjelek	C = 100.0, kernel = poly, gamma = auto	54.5%	27
Moderate	C= 10.0, kernel= poly, gamma= 0.1	54.5%	27
Terbaik	C=1.0, kernel= rbf, gamma= 1	92.8%	1

2. Akurasi model SVM dengan **Chi-Square 20%**:

	Salah satu Kombinasi (C, kernel, gamma)	Akurasi	Jumlah kombinasi
Terjelek	C = 100.0, kernel = poly, gamma = auto	54.5%	27
Moderate	C = 10.0, kernel = poly, gamma = 0.1	54.5%	27
Terbaik	C =1.0, kernel = rbf, gamma = 1	94.6%	3

### 3. Akurasi model SVM dengan **Chi-Square 30%**:

	Salah satu Kombinasi (C, kernel, gamma)	Akurasi	Jumlah kombinasi
Terjelek	C = 100.0, kernel = poly, gamma = auto	54.5%	28
Moderate	C = 10.0, kernel = poly, gamma = 0.1	54.5%	28
Terbaik	C = 10.0, kernel = rbf, gamma = 0.1	94.6%	1

Berdasarkan tiga tabel di atas, dapat dilihat bahwa kombinasi *feature selection* dan parameter yang menghasilkan akurasi terbaik, yaitu 94,6%, adalah Chi-Square 20% dan 30%, C = 1.0, kernel = rbf, dan gamma = 1 dan C = 10.0, kernel = rbf, gamma = 0.1.

#### 2.2.2 App.py

Source Code	Penjelasan
<pre>import streamlit as st import joblib from sklearn.feature_extraction.text import TfidfVectorizer import nltk from nltk.corpus import stopwords from Sastrawi.Stemmer.StemmerFactory import StemmerFactory  nltk.download('stopwords')</pre>	File app.py adalah file web dari model yang dibuat. Library yang digunakan adalah streamlit
<pre># Preprocessing def preprocess_text(text):     # Tokenisasi     tokens = nltk.word_tokenize(text)      # Konversi ke huruf kecil     tokens = [token.lower() for token in tokens]      # Penghapusan kata-kata stopwords     stop_words = set(stopwords.words('indonesian'))     tokens = [token for token in tokens if token not in stop_words]      # Stemming     factory = StemmerFactory()     stemmer = factory.create_stemmer()     tokens = [stemmer.stem(token) for token in tokens]      return tokens</pre>	Ada tahap preprocessing untuk kalimat yang diinput user
<pre># Load the trained model and vectorizer model = joblib.load("model.pkl") tfidf_vect_8020 = joblib.load("vectorizer.pkl")  def classify_review(review):     # Preprocess the input text     review = " ".join(preprocess_text(review))     review = tfidf_vect_8020.transform([review])     # Make predictions</pre>	Model dan vectorizer yang telah dibuat pada file model.py diload ke dalam file app.py untuk kegunaan klasifikasi

<pre> prediction = model.predict(review) return prediction[0] </pre>	
<pre> import streamlit as st import joblib from sklearn.feature_extraction.text import TfidfVectorizer import nltk from nltk.corpus import stopwords from Sastrawi.Stemmer.StemmerFactory import StemmerFactory  nltk.download('stopwords') # Preprocessing def preprocess_text(text):     # Tokenisasi     tokens = nltk.word_tokenize(text)      # Konversi ke huruf kecil     tokens = [token.lower() for token in tokens]      # Penghapusan kata-kata stopwords     stop_words = set(stopwords.words('indonesian'))     tokens = [token for token in tokens if token not in stop_words]      # Stemming     factory = StemmerFactory()     stemmer = factory.create_stemmer()     tokens = [stemmer.stem(token) for token in tokens]      return tokens  # Load the trained model and vectorizer model = joblib.load("model.pkl") tfidf_vect_8020 = joblib.load("vectorizer.pkl")  def classify_review(review):     # Preprocess the input text     review = " ".join(preprocess_text(review))     review = tfidf_vect_8020.transform([review])     # Make predictions     prediction = model.predict(review)     return prediction[0]  # Create a Streamlit web app def main():     st.title("CEK ULASAN DONG")     st.write("Masukan sebuah ulasan dan klik tombol 'OKE' untuk menggolongkan jenis ulasan (postif atau negatif).")      # Input text box     review = st.text_area("Masukkan Ulasan Anda", "")      # Classify button     if st.button("OKE"):         if review:             # Perform classification             prediction = "Positif" if classify_review(review) == 1         else "Negatif" </pre>	<p>Pada bagian main() berisi kode tampilan dari web. User bisa menginputkan kalimat review dan melihat prediksi emosi atau label dari kalimat review tersebut</p>

```
        st.write("Termasuk Jenis Ulasan:", prediction)
    else:
        st.write("Silakan masukkan sebuah ulasan.")

if __name__ == "__main__":
    main()
```

### **BAB III**

### **PENUTUP**

Dalam sistem ini, digunakan teknik pembobotan TF-IDF dapat digunakan dalam kombinasi dengan metode Support Vector Machine (SVM) mengidentifikasi sentimen/emosi dari sebuah/beberapa ulasan. Terdapat dua sentimen yaitu sentimen positif dan sentimen negative. Sistem ini mencapai tingkat akurasi sebesar 94,6% dalam memprediksi sentimen dari teks dengan (Chi Square = 0.2, Parameter: C = 10.0, kernel = rbf, gamma = 0.1). Akurasi tersebut mengindikasikan kemampuan model dalam mengklasifikasikan ulasan sebagai sentimen positif atau negatif.

Pada website menampilkan pilihan user untuk submit teks yang akan di prosessing, setelahnya sistem menampilkan hasil prosessing teks, dan hasil prediksi teks tergolong sentimen positif atau negative. Sistem ini memberikan pengalaman interaktif kepada pengguna dalam melihat hasil analisis dari teks yang diunggah.



## DAFTAR PUSTAKA

- Hermawan, L., & Ismiati, M. B. (2020). Pembelajaran text preprocessing berbasis simulator untuk mata kuliah information retrieval. *Jurnal Transformatika*, 17(2), 188-199.
- Isnain, A. R., Sakti, A. I., Alita, D., & Marga, N. S. (2021). Sentimen Analisis Publik Terhadap Kebijakan Lockdown Pemerintah Jakarta Menggunakan Algoritma Svm. *Jurnal Data Mining Dan Sistem Informasi*, 2(1), 31-37.
- Khomsah, S., & Aribowo, A. S. (2020). Text-preprocessing model youtube comments in indonesian. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, 4(4), 648-654
- Khairunnisa, S., Adiwijaya, A., & Al Faraby, S. (2021). Pengaruh Text Preprocessing terhadap Analisis Sentimen Komentar Masyarakat pada Media Sosial Twitter (Studi Kasus Pandemi COVID-19). *Jurnal Media Informatika Budidarma*, 5(2), 406-414.
- Maarif, A. A. (2015). Penerapan Algoritma TF-IDF untuk Pencarian Karya Ilmiah. Dok. Karya Ilm.| Tugas Akhir| Progr. Stud. Tek. Inform.-S1| Fak. Ilmu Komput.| Univ. Dian Nuswantoro Semarang, (5), 4.