**IMPLEMENTASI SENTIMEN MASYARAKAT BERDASARKAN *TWEET* TERKAIT KEBIJAKAN KEMENDIKBUD DI MASA PANDEMI COVID-19**

**SKRIPSI**

Program Studi Sarjana Informatika

Jurusan Informatika

Oleh:  
**CHAVIN LORENTO**NIM D1041161029

****

FAKULTAS TEKNIK

UNIVERSITAS TANJUNGPURA   
PONTIANAK  
2022

**HALAMAN PERNYATAAN**

Yang bertanda tangan dibawah ini:

Nama : Chavin Lorento

NIM : D1041161029

menyatakan bahwa dalam skripsi yang berjudul “IMPLEMENTASI SENTIMEN MASYARAKAT BERDASARKAN *TWEET* TERKAIT KEBIJAKAN KEMENDIKBUD DI MASA PANDEMI COVID-19” tidak terdapat karya yang pernah diajukan untuk memperoleh gelar sarjana di suatu perguruan tinggi manapun. Sepanjang pengetahuan Saya, tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam Daftar Pustaka.

Demikian pernyataan ini dibuat dengan sebenar-benarnya. Saya sanggup menerima konsekuensi akademis dan hukum di kemudian hari apabila pernyataan yang dibuat ini tidak benar.

****Pontianak, 12 Mei 2022

Chavin Lorento

NIM D1041161029



KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,  
RISET, DAN TEKNOLOGI

UNIVERSITAS TANJUNGPURA

**FAKULTAS TEKNIK**

Jalan Prof. Dr. H. Hadari Nawawi Pontianak 78124 Telp. (0561) 740186   
 Kotak Pos 1049 Email : [ft@untan.ac.id](mailto:ft@untan.ac.id) Website : http://teknik.untan.ac.id

**HALAMAN PENGESAHAN**

IMPLEMENTASI SENTIMEN MASYARAKAT BERDASARKAN *TWEET* TERKAIT KEBIJAKAN KEMENDIKBUD DI MASA PANDEMI COVID-19

Program Studi Sarjana Informatika  
Jurusan Informatika

Oleh :   
  
Chavin Lorento  
NIM. D1041161029

Telah dipertahankan didepan Penguji Skripsi pada tanggal 12 Mei 2022 dalam sidang secara daring (*online*) dan diterima sebagai salah satu persyaratan untuk memperoleh gelar sarjana.

|  |  |  |
| --- | --- | --- |
| Dosen Pembimbing Utama | : Dr. Arif Bijaksana P.N., S.T., M.T. | (NIP. 197208081998021002) |
| Dosen Pembimbing Kedua | : Rudy Dwi Nyoto, S.T., M.Eng. | (NIP. 197803302005011002) |
| Dosen Penguji Utama | : Helfi Nasution, S.Kom., M.Cs. | (NIP. 197104291998021002) |
| Dosen Penguji Kedua | : Enda Esyudha P., S.T., M.Eng. | (NIP. 197206222000122001) |

Susunan Penguji Skripsi :

|  |  |
| --- | --- |
| Pontianak, 12 Mei 2022  Dekan  Dr.rer.nat. Ir. R. M. Rustamaji, M.T., IPU  NIP. 196801161994031003 | Wakil Dekan Bidang Akademik  Dr.-Ing. Ir. Slamet Widodo, M.T., IPM  NIP. 196712231992031002 |

**HALAMAN PERSEMBAHAN**

Skripsi ini didedikasikan kepada orang tua, saudara, keluarga, teman, dan semua pihak yang telah senantiasa memberi masukan, semangat, dan bantuan apapun dalam menyelesaikan skripsi sebagai syarat untuk menyelesaikan kuliah.

**KATA PENGANTAR**

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa, karena dengan rahmat dan karunia-Nya, penulis dapat menyelesaikan skripsi yang berjudul ”IMPLEMENTASI SENTIMEN MASYARAKAT BERDASARKAN *TWEET* TERKAIT KEBIJAKAN KEMENDIKBUD DI MASA PANDEMI COVID-19” sebagai syarat guna memperoleh gelar Sarjana Komputer (S.Kom.) pada Program Studi Informatika, Jurusan Informatika, Fakultas Teknik, Universitas Tanjungpura Pontianak.

Secara khusus penulis mengucapkan terima kasih sebesar-besarnya kepada Bapak Dr. Arif Bijaksana Putra Negara, S.T., M.T. selaku dosen pembimbing yang telah membimbing, memberi masukan/saran, serta penulisan skripsi ini, kemudian Bapak Rudy Dwi Nyoto, S.T., M.Eng. selaku dosen pembimbing skripsi sekaligus pembimbing akademik yang telah membimbing dan memberi masukan/saran dalam penulisan skripsi ini maupun selama proses perkuliahan, serta kepada Bapak Helfi Nasution, S.Kom., M.Cs., dan Bapak Enda Esyudha Pratama, S.T., M.Eng. selaku dosen penguji yang telah memberi masukan/saran untuk perbaikan dalam skripsi ini.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna karena adanya kerterbatasan ilmu dan pengalaman yang dimiliki. Oleh karena itu, semua kritik dan saran yang bersifat membangun akan penulis terima dengan senang hati. Penulis berharap skripsi ini dapat bermanfaat untuk pengembangan wawasan dan peningkatan ilmu pengetahuan bagi semua pihak yang memerlukan.

Pontianak, 12 Mei 2022  
Penulis,

Chavin Lorento

**ABSTRAK**

Twitter selalu menyajikan informasi *trending* sehingga pengguna selalu *update* terkait segala sesuatu yang sedang hangat diperbincangkan (*tweet*), contohnya sentimen masyarakat terkait kebijakan Kemendikbud di masa pandemi Covid-19. Identifikasi t*weets* tersebut dapat dilakukan dengan klasifikasi. Sentimen *tweets* diklasifikasikan ke dalam tiga kelas, yaitu negatif, netral, atau positif. Tujuan penelitian ini adalah menghasilkan model klasifikasi yang memiliki performa terbaik dalam implementasi permasalahan tersebut. Selain itu, penelitian juga melihat pengaruh *tuning hyperparameter* dan pendekatan *multi class* SVM. Metode klasifikasi yang digunakan adalah *multi class* SVM (*Support Vector Machine*) dengan pendekatan OVO (*One Versus One*) dan OVA (*One Versus All*), serta PSO (*Particle Swarm Optimization*) digunakan sebagai *tuning hyperparameter*. Pembagian dataset dilakukan menjadi data *training* sebanyak 5939 data dan data *testing* sebanyak 660 data dengan rasio 90%:10%. Evaluasi dilakukan menggunakan 10-*fold* CV terhadap data *training* dan confusion matrix terhadap data *testing*. Berdasarkan evaluasi terhadap empat model skenario yang dibangun berupa nilai akurasi *training* dan *testing*, model OVO tanpa PSO sebesar 76,86% dan 79,70%, model OVO dengan PSO sebesar 77,94% dan 83,03%, model OVA tanpa PSOsebesar 76,66% dan 79,70%, dan model OVA dengan PSO sebesar 78,62% dan 83,18%. Kemudian, PSO dapat meningkatkan nilai akurasi *training* dan *testing*, yaitu OVO sebesar 1,08% dan 3,33%, serta OVA sebesar 1,96% dan 3,48%. Setelah itu, OVA memiliki akurasi *training* dan *testing* lebih tinggi dibandingOVO hanya jika menggunakan PSO, yaitu 0,68% dan 0,15%. Sedangkan, OVO memiliki nilai akurasi *training* lebih tinggi dibandingOVA hanya jika tidak menggunakan PSO, yaitu 0,2% dan keduanya memiliki nilai akurasi *testing* yang sama. Setelah hasil evaluasi tersebut, model OVA dengan PSOmemberikan performa terbaik dibanding model lainnya dan digunakan sebagai model klasifikasi (*machine learning*) pada aplikasi generik yang dibangun dalam penelitian ini.

Kata kunci: *Twitter*, *klasifikasi, machine learning, PSO, multi class SVM*

**DAFTAR ISI**

[**HALAMAN PERNYATAAN** ii](#_Toc103869745)

[**HALAMAN PENGESAHAN** iii](#_Toc103869746)

[**HALAMAN PERSEMBAHAN** iv](#_Toc103869747)

[**KATA PENGANTAR** v](#_Toc103869748)

[**ABSTRAK** vi](#_Toc103869749)

[**DAFTAR ISI** vii](#_Toc103869750)

[**DAFTAR GAMBAR** x](#_Toc103869751)

[**DAFTAR TABEL** xi](#_Toc103869752)

[**DAFTAR KODE PROGRAM** xii](#_Toc103869753)

[**DAFTAR LAMPIRAN** xiii](#_Toc103869754)

[BAB I PENDAHULUAN 1](#_Toc103869755)

[1.1 Latar Belakang 1](#_Toc103869756)

[1.2 Rumusan Masalah 3](#_Toc103869757)

[1.3 Tujuan Penelitian 3](#_Toc103869758)

[1.4 Batasan Masalah 4](#_Toc103869759)

[1.5 Sistematika Penulisan Skripsi 5](#_Toc103869760)

[BAB II TINJAUAN PUSTAKA 6](#_Toc103869761)

[2.1 Kajian Terkait 6](#_Toc103869762)

[2.2 Twitter 8](#_Toc103869763)

[2.3 Kebijakan Kemendikbud 8](#_Toc103869764)

[2.4 *Scraping* 9](#_Toc103869765)

[2.5 *Text Preprocessing* 9](#_Toc103869766)

[2.6 Pelabelan Data 12](#_Toc103869767)

[2.7 *Imbalanced Data* 13](#_Toc103869768)

[2.8 Pembagian Dataset 14](#_Toc103869769)

[2.9 *Overfitting* 14](#_Toc103869770)

[2.10 *Term Frequency–Inverse Document Frequency* (TF-IDF) 15](#_Toc103869771)

[2.11 *Particle Swarm Optimization* (PSO) 16](#_Toc103869772)

[2.12 *Multi Class* SVM (*Support Vector Machine*) 18](#_Toc103869773)

[2.13 10-*Fold Cross Validation* 22](#_Toc103869774)

[2.14 *Confusion Matrix* 23](#_Toc103869775)

[BAB III METODOLOGI PENELITIAN 25](#_Toc103869776)

[3.1 Alat Penelitian 25](#_Toc103869777)

[3.1.1 Perangkat Keras 25](#_Toc103869778)

[3.1.2 Perangkat Lunak 25](#_Toc103869779)

[3.2 Data Penelitian 25](#_Toc103869780)

[3.3 Langkah Penelitian 25](#_Toc103869781)

[3.3.1 Pengumpulan Data 27](#_Toc103869782)

[3.3.2 *Text Preprocessing* 27](#_Toc103869783)

[3.3.3 Pelabelan Data 27](#_Toc103869784)

[3.3.4 Vektorisasi 27](#_Toc103869785)

[3.3.5 Pembagian Dataset 27](#_Toc103869786)

[3.3.6 *Tuning Hyperparameter* 27](#_Toc103869787)

[3.3.7 *Modelling* 28](#_Toc103869788)

[3.3.8 Skenario Pengujian 28](#_Toc103869789)

[3.3.9 Evaluasi 28](#_Toc103869790)

[BAB IV IMPLEMENTASI DAN HASIL 29](#_Toc103869791)

[4.1 Implementasi 29](#_Toc103869792)

[4.1.1 Pengumpulan Data 29](#_Toc103869793)

[4.1.2 *Text Preprocessing* 30](#_Toc103869794)

[4.1.3 Pelabelan Data 33](#_Toc103869795)

[4.1.4 Vektorisasi 34](#_Toc103869796)

[4.1.5 Pembagian Dataset 34](#_Toc103869797)

[4.1.6 *Tuning Hyperparameter* 35](#_Toc103869798)

[4.1.7 *Modelling* 37](#_Toc103869799)

[4.1.8 Evaluasi 39](#_Toc103869800)

[4.1.9 Arsitektur Sistem 40](#_Toc103869801)

[4.2 Hasil 41](#_Toc103869802)

[4.2.1 Pengumpulan Data 41](#_Toc103869803)

[4.2.2 *Text Preprocessing* 42](#_Toc103869804)

[4.2.3 Pelabelan Data 43](#_Toc103869805)

[4.2.4 Vektorisasi 44](#_Toc103869806)

[4.2.5 Pembagian Dataset 45](#_Toc103869807)

[4.2.6 *Tuning Hyperparameter* 45](#_Toc103869808)

[4.2.7 Evaluasi 46](#_Toc103869809)

[4.2.8 Perbandingan Evaluasi *Training* dan *Testing* 48](#_Toc103869810)

[4.2.9 Aplikasi Generik 50](#_Toc103869811)

[BAB V PENUTUP 52](#_Toc103869812)

[5.1 Kesimpulan 52](#_Toc103869813)

[5.2 Saran 52](#_Toc103869814)

[**DAFTAR PUSTAKA** 53](#_Toc103869815)

**LAMPIRAN**

**DAFTAR GAMBAR**

[**Gambar II.1** Ilustrasi case folding 9](#_Toc103869912)

[**Gambar II.2** Ilustrasi cleaning 10](#_Toc103869913)

[**Gambar II.3** Ilustrasi normalisasi 10](#_Toc103869914)

[**Gambar II.4** Ilustrasi convert negation 11](#_Toc103869915)

[**Gambar II.5** Ilustrasi tokenisasi 11](#_Toc103869916)

[**Gambar II.6** Ilustrasi stopword removal 11](#_Toc103869917)

[**Gambar II.7** Ilustrasi stemming 12](#_Toc103869918)

[**Gambar II.8** Ilustrasi rejoin teks 12](#_Toc103869919)

[**Gambar II.9** Ilustrasi pelabelan VADER 13](#_Toc103869920)

[**Gambar II.10** Imbalanced data 13](#_Toc103869921)

[**Gambar II.11** Ilustrasi vektorisasi TF-IDF 16](#_Toc103869922)

[**Gambar II.12** Flowchart PSO 17](#_Toc103869923)

[**Gambar II.13** Support Vector Machine (SVM) 18](#_Toc103869924)

[**Gambar II.14** Pendekatan multi class 20](#_Toc103869925)

[**Gambar II.15** 10-fold cross validation 22](#_Toc103869926)

[**Gambar II.16** Confusion matrix 23](#_Toc103869927)

[**Gambar III.1** Metodologi penelitian 26](#_Toc103869928)

[**Gambar IV.1** Terjemahan teks rejoin 34](#_Toc103869929)

[**Gambar IV.2** Model skenario pertama 37](#_Toc103869930)

[**Gambar IV.3** Skenario model kedua 38](#_Toc103869931)

[**Gambar IV.4** Skenario model ketiga 38](#_Toc103869932)

[**Gambar IV.5** Skenario model keempat 39](#_Toc103869933)

[**Gambar IV.6** Arsitektur sistem 40](#_Toc103869934)

[**Gambar IV.7** Hasil scraping 42](#_Toc103869935)

[**Gambar IV.8** Hasil remove duplicate 43](#_Toc103869936)

[**Gambar IV.9** Hasil pelabelan data 44](#_Toc103869937)

[**Gambar IV.10** TF-IDF 44](#_Toc103869938)

[**Gambar IV.11** Confusion matrix data testing tiap skenario 46](#_Toc103869939)

[**Gambar IV.12** Perbandingan training dan testing tiap skenario 48](#_Toc103869940)

[**Gambar IV.13** Aplikasi generik 50](#_Toc103869941)

**DAFTAR TABEL**

[**Tabel II.1** Rangkuman Kajian Terkait 7](#_Toc103869942)

[**Tabel III.1** Skenario Pengujian 28](#_Toc103869943)

[**Tabel IV.1** Text Preprocessing 42](#_Toc103869944)

[**Tabel IV.2** Hasil Pembagian Dataset 45](#_Toc103869945)

[**Tabel IV.3** Hasil Tuning Hyperparameter PSO 45](#_Toc103869946)

[**Tabel IV.4** Hasil Evaluasi Tiap Skenario 46](#_Toc103869947)

[**Tabel IV.5** Hasil Klasifikasi Teks Aplikasi Generik 50](#_Toc103869948)

**DAFTAR KODE PROGRAM**

[**Kode Program IV.1** Twitter API 29](#_Toc103869953)

[**Kode Program IV.2** Scraping 29](#_Toc103869954)

[**Kode Program IV.3** Case Folding 30](#_Toc103869955)

[**Kode Program IV.4** Cleaning 30](#_Toc103869956)

[**Kode Program IV.5** Kamus Normalisasi Kata 30](#_Toc103869957)

[**Kode Program IV.6** Normalisasi 31](#_Toc103869958)

[**Kode Program IV.7** Convert Negation 31](#_Toc103869959)

[**Kode Program IV.8** Tokenisasi 31](#_Toc103869960)

[**Kode Program IV.9** Inisialisasi Stoplist 32](#_Toc103869961)

[**Kode Program IV.10** Stopword Removal 32](#_Toc103869962)

[**Kode Program IV.11** Inisialisasi Stemmer 32](#_Toc103869963)

[**Kode Program IV.12** Stemming 32](#_Toc103869964)

[**Kode Program IV.13** Rejoin 33](#_Toc103869965)

[**Kode Program IV.14** Remove Duplicate 33](#_Toc103869966)

[**Kode Program IV.15** VADER dan Teks Terjemahan 33](#_Toc103869967)

[**Kode Program IV.16** Skor dan Label Sentimen 34](#_Toc103869968)

[**Kode Program IV.17** Vektorisasi TF-IDF 34](#_Toc103869969)

[**Kode Program IV.18** Pembagian Dataset 34](#_Toc103869970)

[**Kode Program IV.19** Inisialisasi Parameter PSO 35](#_Toc103869971)

[**Kode Program IV.20** Inisalisasi Partikel PSO 35](#_Toc103869972)

[**Kode Program IV.21** Fitness Function OVO 36](#_Toc103869973)

[**Kode Program IV.22** Fitness Function OVA 36](#_Toc103869974)

[**Kode Program IV.23** Algoritma PSO 36](#_Toc103869975)

[**Kode Program IV.24** Menampilkan Gbest 37](#_Toc103869976)

[**Kode Program IV.25** Model Skenario Pertama 37](#_Toc103869977)

[**Kode Program IV.26** Model Skenario Kedua 38](#_Toc103869978)

[**Kode Program IV.27** Model Skenario Ketiga 38](#_Toc103869979)

[**Kode Program IV.28** Model Skenario Keempat 39](#_Toc103869980)

[**Kode Program IV.29** Evaluasi Data Training Tiap Skenario 39](#_Toc103869981)

[**Kode Program IV.30** Evaluasi Data Testing Tiap Skenario 40](#_Toc103869982)

**DAFTAR LAMPIRAN**

[**LAMPIRAN A KAMUS NORMALISASI DAN STOPLIST** A-1](#_Toc103869983)

[**LAMPIRAN B DATASET PENELITIAN** B-1](#_Toc103869984)

[**LAMPIRAN C FITNESS CANDIDATE DAN GBEST (OVA)** C-1](#_Toc103869985)

[**LAMPIRAN D FITNESS CANDIDATE DAN GBEST (OVO)** D-1](#_Toc103869986)

[**LAMPIRAN E CLASSIFICATION REPORT** E-1](#_Toc103869987)

# PENDAHULUAN

## Latar Belakang

Di Indonesia, penggunaan media sosial memudahkan masyarakat untuk berteman dan berbagi informasi. Segala sesuatu yang diunggah ke dalam media sosial dapat dilihat dengan mudah dan bisa diikuti oleh orang lain, terutama sesuatu yang diunggah oleh figur publik dan menjadi populer, contohnya di Twitter. Twitter selalu menyajikan informasi *trending* sehingga pengguna akan selalu *update* terkait segala sesuatu yang sedang hangat diperbincangkan. Indonesia memperoleh peringkat ke-3 di dunia dengan jumlah pengguna Twitter mencapai 140 juta(Viva.co.id, 2020).

Twitter menjadi media sosial yang sering digunakan oleh masyarakat Indonesia untuk menyampaikan sentimen terhadap topik yang ada, contohnya terkait kebijakan Kemendikbud di masa pandemi Covid-19 ini. Contoh kebijakan yang telah dilakukan oleh Kemendikbud, yaitu kuliah daring, sekolah daring, bantuan UKT (Uang Kuliah Tunggal), pelaksanaan KIP (Kartu Indonesia Pintar), dana BOS (Bantuan Operasional Sekolah), dan bantuan kuota belajar(Teknologi et al., 2021). Kebijakan Kemendikbud tersebut tentunya akan menjadi sebuah topik yang menuai berbagai sentimen dari masyarakat. Sentimen tersebut berupa kumpulan *tweet* yang dapat bersifat negatif, netral, ataupun positif.

Identifikasi terhadap kumpulan *tweet* tersebut dapat dilakukan dengan teknik klasifikasi walaupun dalam jumlah data yang besar agar dapat mengelompokkan *tweets* ke dalam sentimen positif, negatif, ataupun netral. Klasifikasi adalah suatu pengelompokan data dimana data yang digunakan tersebut mempunyai kelas label atau target. Dalam melakukan klasifikasi terdapat beberapa metode yang dapat dimanfaatkan, antara lain *Support Vector Machine* (SVM) (Luqyana et al., 2018)(Alita et al., 2020)(Alhaq et al., 2021)(Buntoro, 2017), *Naive Bayes* (NB)(Hakim et al., 2020)(Hayuningtyas & Sari, 2019), *Artificial Neural Network* (ANN)(Sigala et al., 2019)(Ulfa et al., 2019), *K-Nearest Neighbors* (KNN)(Tempola et al., 2018)(Nasution & Hayaty, 2019), *Convolutional Neural Network*(Peryanto et al., 2020), *Logistic Regression*(Negara et al., 2021)dan lainnya.

Terdapat empat penelitian terkait telah dilakukan sebelumnya, pertama (Hakim et al., 2020) melakukan penelitian membandingkan metode klasifikasi SVM, NB dan KNN. Hasilnya metode SVM memperoleh akurasi sebesar 80,05%, dibanding NB dan KNN masing-masing hanya 66,81% dan 51,45%. Kedua, (Putra et al., 2020) melakukan penelitian membandingkan kernel RBF, Polynomial dan Linear. Hasilnya RBF memiliki akurasi sebesar 100%, dibanding dengan Polynomial dan Linear masing- masing hanya 83,117% dan 74,026%. Ketiga, (Hayuningtyas & Sari, 2019) melakukan penelitian membandingkan metode klasifikasi NB terhadap NB berbasis *Particle Swarm Optimization* (PSO). Hasilnya NB berbasis PSO memiliki akurasi sebesar 94,02%, dibanding NB hanya 70%. Keempat, (Alita et al., 2020) melakukan penelitian membandingkan *multi class* SVM pendekatan *One Versus One* (OVO) dan *One Versus All* (OVA). Hasilnya pendekatanOVA sedikit lebih baik dibanding pendekatan OVO dengan akurasi masing-masing 92% dan 91,94%.

Oleh karena itu, metode klasifikasi yang akan digunakan adalah *multi class* SVM. Metode SVM memungkinkan perhitungan untuk masalah linier dengan menerapkan transformasi matematis untuk ruang belajar menggunakan fungsi kernel, contohnya kernel RBF. SVM mencari hyperplane (sebuah fungsi yang dapat digunakan untuk pemisah antar kelas) terbaik dengan memaksimalkan jarak antar kelas(Tineges et al., 2020). *Multi class* SVM ini sendiri adalah pengembangan dari SVM agar dapat melakukan klasifikasi *non linier* (lebih dari dua kelas). *Multi class* SVM memiliki dua pendekatan(Alhaq et al., 2021), yaitu *One Versus One* (OVO) berupa strategi yang terdiri dari satu *classifier* per kelas, yang dilatih untuk membedakan sampel satu kelas dari sampel kelas lainnya dan *One Versus All* (OVA) berupa strategi yang terdiri dari satu *classifier* per kelas, yang dilatih untuk membedakan sampel satu kelas dari sampel semua kelas yang tersisa.

Penelitian ini juga akan menggunakan PSO sebagai *tuning hyperparameter*. PSO merupakan teknik optimasi yang sangat sederhana untuk menerapkan dan memodifikasi beberapa parameter. PSO juga terbukti sebagai algoritma yang baik dan efektif untuk permasalahan optimisasi karena kemudahannya dalam penerapan kode dan performanya yang konsisten(Hakim et al., 2020). Sedangkan, *tuning hyperparameter* merupakan pemilihan parameter yang optimal pada algoritma pembelajaran. *Tuning hyperparameter* juga salah satu cara untuk mengatasi *overfitting* yang merupakan penyebab terbesar buruknya performa model *machine learning*(Negara et al., 2021).

Berdasarkan uraian diatas, penelitian ini akan menggunakan dua pendekatan *multi class* SVM, yaitu OVO dan OVA dengan kernel RBF. Kemudian, diterapkan tuning hyperparameter PSO untuk melihat pengaruhnya terhadap setiap pendekatan *multi class* SVM tersebut. Setiap model klasifikasi yang dibangun akan saling dibandingkan untuk mencari model klasifikasi dengan performa terbaik. Model klasifikasi dengan performa terbaik digunakan untuk menentukan klasifikasi sentimen pada aplikasi generik yang dibangun.

## Rumusan Masalah

Berdasarkan latar belakang tersebut, maka rumusan masalahnya yaitu:

1. Skenario model manakah yang memiliki performa terbaik dalam implementasi sentimen masyarakat berdasarkan *tweet* terkait kebijakan Kemendikbud di masa pandemi Covid-19.
2. Bagaimana pengaruh *tuning hyperparameter* terhadap model klasifikasi yang dibangun.
3. Bagaimana pengaruh pendekatan *multi class* terhadap model klasifikasi yang dibangun.

## Tujuan Penelitian

Berdasarkan rumusan masalah tersebut, maka tujuan penelitian ini dilakukan yaitu:

1. Menghasilkan model klasifikasi yang memiliki performa terbaik dalam implementasi sentimen masyarakat berdasarkan *tweet* terkait kebijakan Kemendikbud di masa pandemi Covid-19.
2. Dapat mengetahui pengaruh *tuning hyperparameter* terhadap model klasifikasi yang dibangun.
3. Dapat mengetahui pengaruh pendekatan *multi class* terhadap model klasifikasi yang dibangun.

## Batasan Masalah

Peneliti memberikan batasan masalah diberikan untuk menghindari adanya penyimpangan maupun pelebaran inti masalah dalam penyusunan penelitian ini, yaitu:

1. Klasifikasi dan pembuatan aplikasi generik dilakukan dengan bahasa pemrograman Python.
2. Data yang digunakan adalah *tweets* berupa *plaintext* berbahasa Indonesia yang diperoleh dengan *scraping* dimulai dari tanggal 03 April 2021 hingga 20 Mei 2021. Scraping yang dilakukan memanfaatkan *library* Python, yaitu Tweepy.
3. Pelabelan data dilakukan ke dalam tiga kelas, yaitu negatif, netral, dan positif. Pelabelan dilakukan setelah *text preprocessing* dan secara otomatis dengan memanfaatkan *library* Python, yaitu Vader.
4. Pembagian keseluruhan dataset yang dilakukan dalam penelitian ini, yaitu 90% untuk *training dataset* dan 10% untuk *testing dataset*. Pembagian dataset dilakukan secara otomatis dengan memanfaatkan *library* Python, yaitu Scikit-Learn.
5. Metode klasifikasi yang digunakan adalah SVM dengan dua pendekatan *multi class*, yaitu *One Versus One* (OVO) dan *One Versus All* (OVA). Fungsi kernel SVM yang digunakan adalah RBF/*gaussian*.
6. *Particle Swarm Optimization* (PSO) digunakan sebagai *tuning hyperparameter* dalam penelitian ini. *Tuning hyperparameter* dilakukan terhadap dua *hyperparameter* didalam *multi class* SVM kernel RBF, yaitu *C* dan *gamma*.

## Sistematika Penulisan Skripsi

Adapun sistematika dari penulisan tugas akhir ini disusun dalam lima bab yang terdiri dari:

1. Bab I Pendahuluan adalah bab yang berisi tentang latar belakang, perumusan masalah, tujuan penelitian, pembatasan masalah, dan sistematika penulisan skripsi.
2. Bab II Tinjauan Pustaka adalah bab yang berisi tentang landasan teori berkaitan dengan penelitian yang akan dilakukan.
3. Bab III Metodologi Penelitian adalah bab yang berisi tentang gambaran langkah-langkah penelitian yang akan dikerjakan. Langkah-langkah tersebut berupa pengumpulan data, *text preprocessing*, pelabelan data, vektorisasi, pembagian dataset, *tuning hyperparameter*, *modelling*, dan evaluasi.
4. Bab IV Implementasi dan Hasil adalah bab yang berisi tentang pemaparan penelitian yang telah dilakukan sesuai metodologi penelitian yang dibangun. Setiap langkah penelitian yang telah dikerjakan akan dipaparkan prosesnya, yaitu berupa narasi yang disertai gambar/tabel/grafik untuk mendukung narasi tersebut.
5. Bab V Penutup adalah bab yang berisi kesimpulan dari penelitian yang telah dilakukan dan saran/rekomendasi untuk perbaikan, pengembangan atau kesempurnaan/kelengkapan penelitian yang telah dilakukan.

# TINJAUAN PUSTAKA

## Kajian Terkait

Pertama, (Hakim et al., 2020) melakukan penelitian tentang “Sentimen Analisis Stay Home menggunakan Naive Bayes, SVM, dan KNN”. Penelitian tersebut mengklasifikasikan sentimen berupa negatif dan positif dari *tweets* masyarakat. Metode klasifikasi yang digunakan adalah *Support Vector Machine* (SVM), *Naive Bayes* (NB), dan *K-Nearest Neighbor* (KNN). Setiap metode akan dilakukan perbandingan dan jumlah dataset yang digunakan, yaitu 1652 *tweet* yang berasal dari hasil *scraping* Twitter. Dari hasil pengujian terhadap 1178 *tweet*, metode SVM memiliki akurasi lebih tinggi dibanding dengan NB dan KNN. Urutan nilai akurasinya adalah SVM (80,05%), NB (66,81%), dan KNN (51,45%).

Kedua, (Putra et al., 2020) melakukan penelitian tentang “Tingkat Akurasi Jarak Kelahiran Di Kampung Keluarga Berencana (KB) Dengan Metode SVM”. Penelitian tersebut mengklasifikasikan data jarak antar kelahiran bayi ke dalam dua kelas, yaitu ‘ideal’ dan ‘tidak ideal’. Metode klasifikasi yang digunakan adalah *Support Vector Machine* (SVM) dengan menerapkan tiga jenis kernel, yaitu RBF, Polynomial, dan Linear. Setiap kernel yang digunakan akan dibandingkan dan jumlah dataset yang digunakan adalah 96 data yang berasal dari responden. Dari hasil pengujian terhadap 19 data, kernel RBF memiliki akurasi lebih tinggi dibanding Polynomial dan Linear. Urutan nilai akurasinya adalah RBF (100%), Polynomial (83,117%), dan Linear (74,026%).

Ketiga, (Hayuningtyas & Sari, 2019) melakukan penelitian tentang “Analisis Sentimen Opini Publik Bahasa Indonesia Terhadap Wisata TMII Menggunakan Naive Bayes dan PSO”. Penelitian tersebut mengklasifikasikan sentimen berupa negatif dan positif dari ulasan wisatawan. Metode klasifikasi yang digunakan adalah *Naive Bayes* (NB) dengan *Particle Swarm Optimization* (PSO) sebagai fitur seleksi. Penelitian ini membandingkan pengaruh PSO terhadap kinerja NB dan jumlah dataset yang digunakan adalah 100 data. Dari hasil pengujian terhadap 100 data dari dataset yang sama, NB berbasis PSO memiliki akurasi lebih tinggi dibanding dengan NB tanpa PSO dengan perbedaan sebesar 24,02%. Urutan nilai akurasinya adalah NB-PSO (94,02%) dan NB (70%).

Keempat, (Alita et al., 2020) melakukan penelitian tentang ”Implementasi Algoritma Multi Class SVM pada Opini Publik Berbahasa Indonesia di Twitter”. Penelitian tersebut mengklasifikasikan sentimen berupa negatif, netral, maupun positif dari *tweets* publik. Metode klasifikasi yang digunakan adalah *multi class* SVM dengan dua pendekatan, yaitu *One Versus All* (OVA) dan *One Versus One* (OVO). Penelitian ini membandingkan kedua pendekatan *multi class* SVM tersebut dan jumlah dataset yang digunakan adalah 2000. Dari hasil pengujian terhadap 2000 data dari dataset yang sama, pendekatan OVA sedikit lebih unggul dibanding dengan OVO dengan akurasi masing-masing 92% dan 91,94%.

Berikut ditampilkan rangkuman kajian terkait dalam penelitian ini pada Tabel II.1 dibawah ini.

**Tabel II.1** Rangkuman Kajian Terkait

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Peneliti | Topik | Kelas Klasifikasi | Metode Klasifikasi | Dataset | Hasil (Akurasi) |
| Hakim et al (2020) | Sentimen tweets masyarakat | Positif dan negatif | SVM, NB, dan KNN | Training = 474 data Testing = 1178 data | SVM = 80,05% NB = 66,81% KNN = 51,45% |
| Putra et al (2020) | Klasifikasi jarak anatar kelahiran | Ideal dan tidak ideal | SVM dengan kernel RBF, Polynomial, dan Linear | Training = 77 data  Testing = 19 data | RBF = 100% Polynomial = 83,117% Linear = 74,026% |
| Hayuningtyas, Ratih Yulia & Retno Sari (2019) | Klasifikasi ulasan wisatawan | Positif dan negatif | NB-PSO dan NB | Training dan testing dataset yang sama = 100 data | NB-PSO = 94,02% NB = 70% |
| Alita et al (2020) | Sentimen tweets publik | Negatif, netral, dan positif | Multi class SVM: OVA dan OVO | Training dan testing dataset yang sama = 2000 data | OVA = 92% OVO = 91,94% |

## Twitter

Twitter merupakan layanan social media *micro blogging* yang memberi keleluasaan bagi penggunanya dalam menyampaikan pesan berupa opini dan penilaian atas suatu isu atau persoalan. Pesan yang disampaikan umumnya disebut dengan istilah *tweet* atau kicauan yang memiliki kapasitas maksimal 280 karakter teks (Husada & Paramita, 2021). Pengguna bisa membatasi pengiriman atau pesan yang hanya dapat dilihat oleh teman-teman atau pengikut (*followers*). Pengguna dapat memeriksa orang-orang yang membatalkan pertemanan (*unfollowing*) melalui layanan yang ada di dalam Twitter. Pengguna dapat menulis pesan dan mengelompokannya berdasarkan topic atau jenis pesan dengan menggunakan tagar “#” (hashtag). Pengguna juga dapat menghubungkan pesan *tweet* mereka dengan orang lain menggunakan tanda “@” yang diikuti dengan nama pengguna yang akan dihubungkan untuk mengirim atau membalas pengguna lain (Tane et al., 2019).

Dalam penelitian ini, Twitter API (Application Programming Interface) akan menjadi media untuk pengumpulan dataset yang dibutuhkan dalam klasifikasi. Diperlukan *consumer keys* dan *access tokens* dari Twitter API agar dapat dilakukan *scraping*.

## Kebijakan Kemendikbud

Kemendikbud adalah kementerian dalam pemerintah Indonesia (sekarang Kemendikbudristek) yang memiliki tugas **Berdasarkan Peraturan Presiden Republik Indonesia Nomor 62 tahun 2021 tentang Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi (Kemendikbudristek), yaitu** menyelenggarakan urusan pemerintahan di bidang pendidikan, kebudayaan, ilmu pengetahuan, dan teknologi untuk membantu Presiden dalam menyelenggarakan pemerintahan negara**.** Pada tahun 2021, Kementerian Riset dan Teknologi digabungkan ke Kementerian Pendidikan dan Kebudayaan (Kemendikbud) menjadi Kementerian Pendidikan, Kebudayaan, Riset dan Teknologi (Kemendikbudristek). Kementerian ini dipimpin oleh Nadiem Anwar Makarim yang dilantik oleh Presiden Joko Widodo pada 28 April 2021 (Teknologi et al., 2021).

**Dalam masa pandemi Covid-19 sejak Maret 2020 lalu, ada berbagai kebijakan Kemendikbud yang berjalan antara lain** bantuan kuota belajar, dana BOS (Bantuan Operasional Sekolah), KIP (Kartu Indonesia Pintar), bantuan UKT (Uang Kuliah Tunggal), serta pembelajaran secara daring (*online*), baik di persekolahan maupun di perkuliahan, dan lainnya(Teknologi et al., 2021).

Dalam penelitian ini, digunakan enam buah kata kunci dalam melakukan *scraping* terhadap Twitter, yaitu ’kuota belajar’, ’dana BOS’, ’KIP’, ’UKT’, ’kuliah daring’, dan ’sekolah daring’.

## *Scraping*

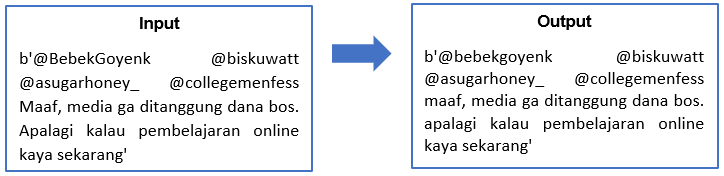
*Scraping* adalah teknik pengambilan atau esktraksi data/informasi dari sebuah *website*/situs secara otomatis dengan mengurai *tag hypertext* dan mengambil informasi *plaintext* yang terkandung didalamnya (Hernandez-Suarez et al., 2018). Dalam penelitian ini, scraping *tweets* dilakukan dengan bahasa pemrograman Python dan memanfaatkan *library*, yaitu Tweepy untuk akses ke Twitter API.

## *Text Preprocessing*

*Text preprocessing* merupakan tahap mempersiapkan teks yang tidak terstruktur agar menjadi data yang baik dan siap untuk diolah. Proses ini dilakukan untuk menggali, mengolah dan mengatur infomasi dan untuk menganalisis hubungan tekstual dari data terstruktur dan data tidak testruktur(Luqyana et al., 2018).

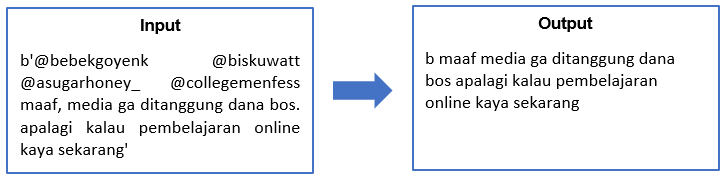
Dalam penelitian ini, tahapan yang dilakukan dalam *text preprocessing* antara lain:

1. *Case folding*, yaitu mengubah setiap bentuk kata dalam teks/dokumen menjadi menjadi *lowercase* atau huruf kecil secara keseluruhan(Luqyana et al., 2018). Berikut ditampilkan ilustrasi *case folding* pada Gambar II.1 dibawah ini.



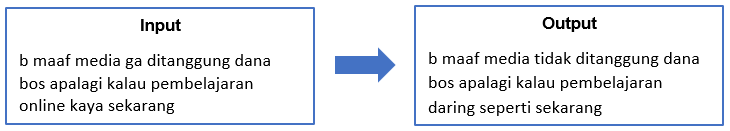
**Gambar II.1** Ilustrasi case folding

1. *Cleaning, yaitu* proses pembersihan teks/dokumen yang bertujuan untuk mengurangi *noise*(Luqyana et al., 2018). Pembersihan teks yang dilakukan adalah tanda baca/delimeter (koma, titik, kurung, dan sejenisnya), *enter*, emoji, *username*, *hashtag*, *URL/link*, *mention*, dan angka. Berikut ditampilkan ilustrasi *cleaning* pada Gambar II.2 dibawah ini.



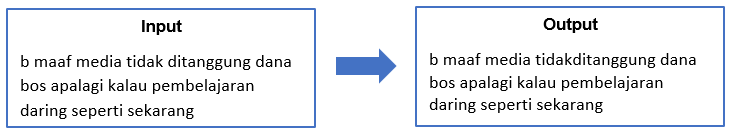
**Gambar II.2** Ilustrasi cleaning

1. Normalisasi, yaitu proses normalisasi bahasa terhadap kata tidak baku. Tahapan ini bertujuan untuk mengembalikan bentuk penulisan dari masing-masing kata yang sesuai dengan Kamus Besar Bahasa Indonesia (KBBI) (Luqyana et al., 2018). Normalisasi juga dilakukan terhadap *slang word* maupun kata bernuansa bahasa Inggris. Berikut ditampilkan ilustrasi normalisasi pada Gambar II.3 dibawah ini.



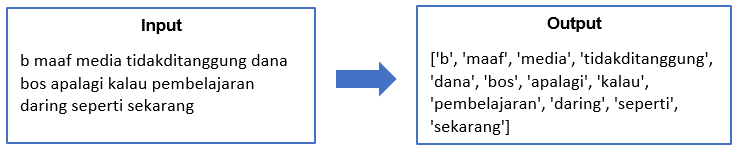
**Gambar II.3** Ilustrasi normalisasi

1. *Convert negation*, yaitu proses konversi kata-kata negasi yang terdapat pada suatu kalimat, karena kata negasi mempunyai pengaruh dalam merubah nilai emosi/sentimen pada sebuah kalimat(Salekhah, 2016). Kata-kata negasi dalam bahasa Indonesia antara lain ‘tidak’, ‘bukan’, ‘jangan’, dan ‘belum’(Syafar, 2016). Kata negasi tersebut akan digabungkan dengan kata sesudahnya. Berikut ditampilkan ilustrasi *convert negation* pada Gambar II.4 dibawah ini.



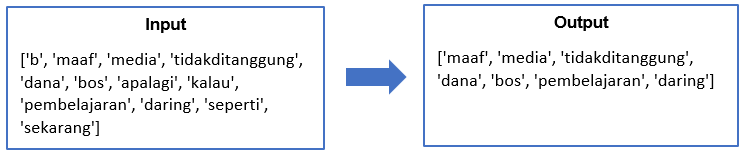
**Gambar II.4** Ilustrasi convert negation

1. Tokenisasi, yaitu proses untuk memotong dokumen/teks menjadi pecahan kecil yang dapat berupa bab, *sub-bab*, paragraf, kalimat, dan kata (token) (Luqyana et al., 2018). Dalam tokenisasi ini, teks dipecah menjadi token-token dan *white space* akan dihilangkan. Berikut ditampilkan ilustrasi tokenisasi pada Gambar II.5 dibawah ini.



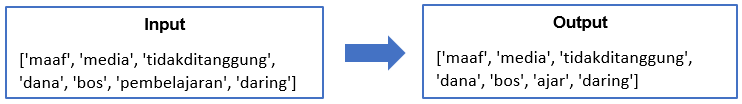
**Gambar II.5** Ilustrasi tokenisasi

1. *Stopword Removal*, yaitu proses menghapus kata umum yang tidak memiliki arti penting dan tidak digunakan. Tujuannya untuk mengurangi jumlah kata yang disimpan oleh sistem(Luqyana et al., 2018). Acuan terhadap kata-kata tersebut (*stopword*) ada di dalam daftar berupa stoplist yang digunakan. Berikut ditampilkan ilustrasi *stopword removal* pada Gambar II.6 dibawah ini.



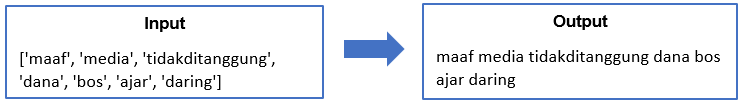
**Gambar II.6** Ilustrasi stopword removal

1. *Stemming*, yaitu proses untuk mencari *stem* (kata dasar) dari kata hasil *stopword removal* (*filtering*). Terdapat dua aturan dalam melakukan *stemming* yaitu dengan pendekatan kamus dan pendekatan aturan (Luqyana et al., 2018). Berikut ditampilkan ilustrasi *stemming* pada Gambar II.7 dibawah ini.



**Gambar II.7** Ilustrasi stemming

1. *Rejoin*, yaitu penggabungan kembali token-token dalam teks/dokumen sebelumnya menjadi kalimat-kalimat yang utuh. Berikut ditampilkan ilustrasi *rejoin* pada Gambar II.8 dibawah ini.



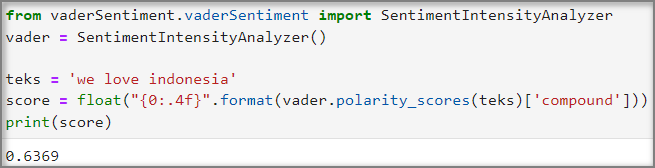
**Gambar II.8** Ilustrasi rejoin teks

Tahapan-tahapan *preprocessing* tersebut dilakukan dengan menggunakan bahasa pemrograman Python yang memanfaatkan *library* seperti Natural Language Toolkit (NLTK), Sastrawi, Pandas, dan lainnya.

## Pelabelan Data

Pelabelan ditujukan untuk membagi data ke dalam beberapa kelas/kategori sentimen yang akan digunakan dalam penelitian. Jumlah kelas sentimen yang digunakan adalah tiga kelas, yaitu negatif, netral, dan positif. Pelabelan yang digunakan bersifat otomatis pada dataset yang belum memiliki label kelas agar data dapat digunakan untuk membangun model klasifikasi. Metode ini digunakan untuk menghindari pemberian label kelas untuk dataset yang sangat besar(Ayunin, 2019). Contoh pelabelan otomatis dalam kelas sentimen dapat menggunakan *library* Python, yaitu Textblob dan Vader(Ghudafa et al., 2021).

Dalam penelitian ini, pelabelan otomatis menggunakan Vader dikarenakan Vader memiliki performa lebih baik dibanding Textblob(Ghudafa et al., 2021). VADER (Valence Aware Dictionary and Sentiment Reasoner) merupakan metode analisis *lexicon based*, yaitu menganalisis teks berdasarkan *lexicon* (*a library*) yang menghasilkan *class sentiment* berupa positif, negatif dan netral dengan tambahan skor total atau disebut *compound score*(Sumitro et al., 2021). Nilai *compound* ini dapat digunakan sebagai satuan standar untuk mengklasifikasikan kalimat dengan ketentuan positif untuk *compound* lebih dari 0, negatif untuk *compound* kurang dari 0, dan netral untuk *compound* sama dengan 0(Ghudafa et al., 2021). Untuk ilustrasi pelabelan Vader dapat dilihat pada Gambar II.9 dibawah ini.



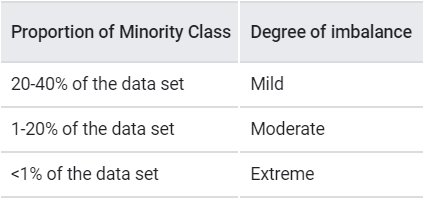
**Gambar II.9** Ilustrasi pelabelan VADER

## *Imbalanced Data*

Data tidak seimbang (*imbalanced data*) mengacu pada kumpulan data di mana kelas target memiliki distribusi data yang tidak merata (*imbalanced class*), yaitu kelas yang membentuk proporsi besar dari kumpulan data (kelas mayor) dan kelas yang membentuk proporsi yang lebih kecil (kelas minoritas).

Dalam klasifikasi, *imbalanced class* tidak mengakibatkan hilangnya performa yang signifikan apabila rasio *imbalanced* class tergolong rendah. Namun, klasifikasi akan mengalami peningkatan kehilangan performa secara cepat apabila rasio *imbalanced class* tergolong tinggi(Thabtah et al., 2020).

Berdasarkan Gambar II.10 (Google, 2019), *imbalanced data* dikategorikan rendah (*mild*) apabila kelas minoritas memiliki proporsi 20 hingga 40% dari kumpulan data, dikategorikan sedang (*moderate*) apabila kelas minoritas memiliki proporsi 1 hingga 20% dari kumpulan data, dan dikategorikan tinggi (*extreme*) apabila kelas minoritas kurang dari 1% dari kumpulan data.



**Gambar II.10** Imbalanced data

Dalam penelitian ini, jika perbedaan label kelas tidak tergolong *moderate*/*extreme* untuk kelas minoritas, maka tidak dilakukan *balancing data* didalamnya.

## Pembagian Dataset

Pembagian dataset menjadi dua bagian, yaitu *training dataset* untuk melatih model klasifikasi yang dibangun dan *testing dataset* untuk menguji model klasifikasi yang telah dibangun. Kemudian, ada *validation dataset* yang merupakan sebagian kecil yang terpisah dari *training dataset*. *Training* dan *validation* dataset digunakan untuk membangun dan menyempurnakan *classifier* dengan fungsi *cross validation*. Performa *classifier* yang sudah dilatih kemudian diperiksa dan dievaluasi menggunakan *testing dataset*(Pawluszek-Filipiak & Borkowski, 2020).

Dalam penelitian ini, pembagian dataset dilakukan menggunakan bahasa pemrograman Python yang memanfaatkan *library*, yaitu *Scikit-Learn.* Rasio *training dataset* dan *testing dataset* yang digunakan adalah 90%:10% dikarenakan memiliki nilai *error rate* lebih kecil dibanding rasio 50%:50%, 60%:40%, 70%:30%, dan 80%:20%(Thabtah et al., 2020). Kemudian, *training dataset* tersebut akan dipecah menjadi *training set* dan *validation set* dengan menggunakan 10-*Fold Cross Validation* (CV).

## *Overfitting*

*Overfitting* adalah masalah mendasar dalam *machine learning* yang mencegah model dalam menggeneralisasi dengan baik data yang diamati (data pelatihan) serta data yang tidak terlihat (data uji)(Ying, 2019). Model dilatih pada data *training*, tetapi kemudian diterapkan untuk membuat prediksi pada data *testing*. Tujuannya adalah untuk memaksimalkan akurasi prediksi pada data *testing*. Apabila model yang dibuat terlalu fokus pada data *training*, ada risiko menangkap adanya data *noise* yang seharusnya diabaikan sehingga tidak bisa melakukan prediksi dengan baik jika diberikan dataset lain (*testing*). Fenomena ini biasanya disebut *overfitting* (Dietterich, n.d.).

Dalam penelitian ini, dilakukan pengecekan *overfitting* berdasarkan nilai evaluasi *training* dan *testing* tiap model skenario yang dibangun.

## *Term Frequency–Inverse Document Frequency* (TF-IDF)

Vektorisasi bertujuan untuk memberikan bobot pada fitur kata berdasarkan frekuensi kemunculan kata. Fitur kata yang telah diberi bobot dapat digunakan untuk proses klasifikasi(Rofiqoh et al., 2017). Vektorisasi yang digunakan adalah *Term Frequency-Inverse Document Frequency* (TF-IDF).

Model pembobotan TF-IDF merupakan metode yang mengintegrasikan model *Term Frequency* (TF) dan *Inverse Document Frequency* (IDF). TF merupakan jumlah kemunculan atau frekuensi kata pada suatu dokumen dan IDF bobot kebalikan dari bobot *Document Frequency* (DF). Kata yang jarang muncul di banyak dokumen mempunyai bobot IDF yang tinggi. Maka, pembobotan TF-IDF ini adalah hasil perkalian dari pembobotan TF dan IDF dari suatu term(Rofiqoh et al., 2017).

Berikut rumus dalam pembobotan TF-IDF (Rofiqoh et al., 2017):

|  |  |  |
| --- | --- | --- |
|  |  | (II.1) |
|  |  | (II.2) |
|  |  | (II.3) |

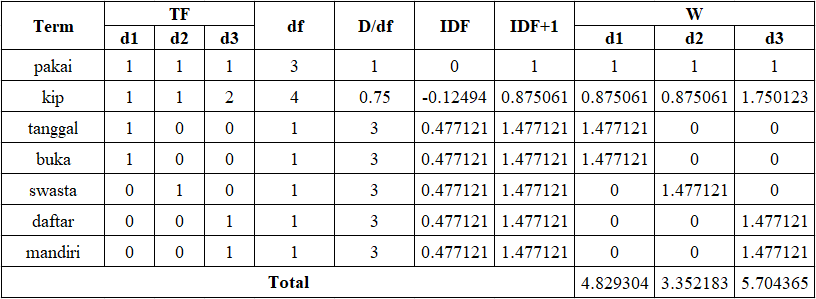
**Keterangan :**

|  |  |
| --- | --- |
|  | = frekuensi term |
|  | = bobot frekuensi term |
|  | = jumlah frekuensi dokumen yang mengandung term |
|  | = jumlah total dokumen |
|  | = bobot TF-IDF |

Berikut akan diberikan ilustrasi pembobotan TF-IDF:

|  |  |
| --- | --- |
| Dokumen 1 (**d1**) : | b'@fy\_zoe @sbmptnfess Yang pake kip sampai tanggal berapa bukanya?' |
| Dokumen 2 (**d2**) : | b'@utbkfess Mau nanya swasta udah bisa pakai kip belum ya?' |
| Dokumen 3 (**d3**) : | b' kalo gini mau daftar mandiri udah bisa pake kip? soalnya bru buat kip https://t.co/IYCFne1S0C' |

Apabila sudah melalui *text preprocessing*, maka kata yang tersisa antara lain ’pakai’, ’kip’, ’tanggal’, dan ’buka’ pada d1, ’swasta’, ’pakai’, dan ’kip’ pada d2, dan ’daftar’, ’mandiri’, ’pakai’, ’kip’, dan ’kip’ pada d3. Kata-kata tersebut akan dihitung bobotnya dan dapat dilihat pada Gambar II.11 dibawah ini.



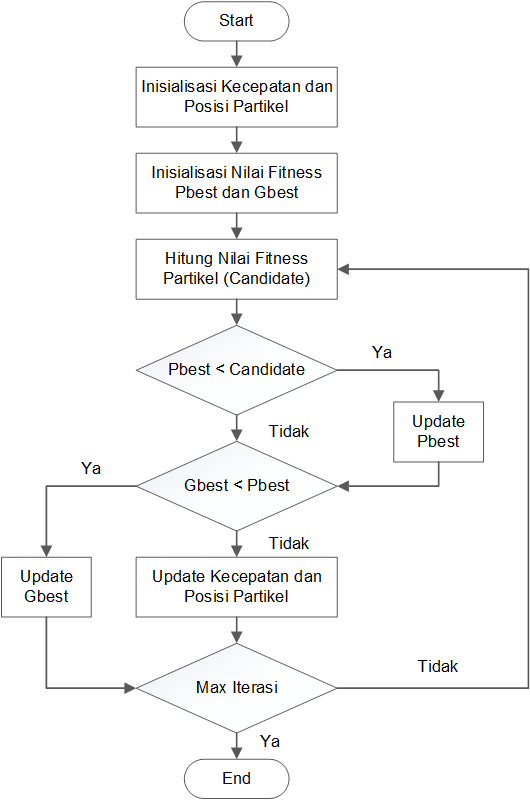
**Gambar II.11** Ilustrasi vektorisasi TF-IDF

Dalam penelitian ini, vektorisasi TF-IDF dilakukan menggunakan bahasa pemrograman Python yang memanfaatkan *library*, yaitu *Scikit-Learn.*

## *Particle Swarm Optimization* (PSO)

*Particle Swarm Optimization* (PSO) merupakan teknik optimasi yang sangat sederhana untuk menerapkan dan memodifikasi beberapa parameter. Dalam PSO terdapat beberapa teknik untuk melakukan pengoptimasian, diantaranya meningkatkan bobot atribut (*attribute weight*) terhadap semua atribut atau variabel yang dipakai, menyeleksi atribut (*attribute selection*), dan *feature selection* (Aaputra, 2019).

PSO ditemukan oleh James Kennedy dan Russ Eberhart pada tahun 1995. PSO terinspirasi oleh tingkah laku sosial kawanan burung yang terbang berduyun-duyung (*bird flocking*) atau gerombolan ikan yang berenang berkelompok (*fish schooling*). Ratusan burung mampu terbang tanpa bertabrakan dan ratusan ikan berenang sanggup bergerak cepat tanpa saling bertabrakan. Padahal jarak mereka begitu dekat(Aaputra, 2019).



**Gambar II.12** Flowchart PSO

Pada Gambar II.12, ditampilkan flowchart PSO yang dibangun berdasarkan (Istikomah et al., 2017) dengan prosedural antara lain:

1. Inisialisasi kecepatan dan posisi partikel awal
2. Inisialisasi nilai *fitness* *Pbest* (Personal Best) dan Gbest (Global Best)
3. Menghitung nilai *fitness* partikel (*candidate*)
4. Membandingkan nilai *fitness candidate* terhadap nilai *fitness Pbest*. Apabila nilai *fitness Pbest* lebih kecil dibanding nilai *fitness candidate*, maka nilai *fitness candidate* menjadi nilai *fitness Pbest* yang baru.
5. Membandingkan nilai *fitness Pbest* terhadap nilai *fitness Gbest*. Apabila nilai *fitness Gbest* lebih kecil dibanding nilai *fitness Pbest*, maka nilai *fitness Pbest* menjadi nilai *fitness Gbest* yang baru.
6. *Update* kecepatan dan posisi partikel untuk iterasi selanjutnya.
7. Lakukan kembali langkah ketiga hingga mencapai iterasi maksimum.

Berikut rumus dalam meng-*update* kecepatan dan posisi partikel:

|  |  |  |
| --- | --- | --- |
|  |  | (II.4) |
|  |  | (II.5) |

**Keterangan :**

|  |  |
| --- | --- |
|  | = kecepatan partikel saat ini |
|  | = posisi partikel saat ini |
|  | = berat inertia |
|  | = koefisien akselerasi |
|  | = bilangan acak dengan rentang 0 hingga 1 |
|  | = posisi personal best partikel saat ini |
|  | = posisi global best partikel saat ini |

Dalam penelitian ini, PSO akan digunakan sebagai *tuning hyperparameter* terhadap hyperparameter gamma dan C pada *multi class* SVM dengan kernel RBF(Prangga, 2017). *Tuning hyperparameter* PSO akan dibangun dengan bahasa pemrograman Python dan memanfaatkan *library*, yaitu *random*, Numpy, *Scikit-Learn*, dan lainnya. Berdasarkan (Okwu & Tartibu, 2021), parameter *w*, *c1*, dan *c2* yang optimal adalah bernilai 0.9, 0.5, dan 0.3. Berdasarkan (Piotrowski et al., 2020), jumlah populasi/partikel yang optimal bernilai 100, sedangkan iterasi dilakukan sebanyak 3 kali sebagai kondisi tercapainya algoritma PSO oleh penulis.

## *Multi Class* SVM (*Support Vector Machine*)

*Support Vector Machine* (SVM) adalah suatu teknik yang relatif baru untuk melakukan prediksi, baik dalam kasus klasifikasi maupun regresi. Metode SVM asli diciptakan oleh Vladimir Vapnik dan turunan standar saat ini (margin lunak) diusulkan oleh Corinna Cortes dan Vapnik Vladimir(Luqyana et al., 2018).



**Gambar II.13** Support Vector Machine (SVM)

Pada Gambar II.13(Rofiqoh et al., 2017), metode SVM memungkinkan perhitungan untuk masalah linear dengan menerapkan transformasi matematis untuk ruang belajar menggunakan fungsi kernel. Metode SVM memiliki konsep sentral dalam mengklasifikasikan data, yaitu menentukan *hyperplane* terbaik untuk memberi jarak atau pemisah antara dua kelas yang telah ditentukan. Cara kerja SVM yaitu dengan dengan menemukan *hyperplane* optimal yang memberi jarak atau pemisah antar dua kelas, yaitu *hyperplane* yang mempunyai *margin* maksimum. Jarak antar titik data terdekat dengan *hyperplane* diklaim sebagai *margin*. Adapun vektor pendukung sebagai titik terdekat dengan *hyperplane*(Tineges et al., 2020).

Tahapan dalam metode *Support Vector Machine* adalah sebagai berikut(Tineges et al., 2020):

1. Menentukan kata yang sering muncul dari tiap dokumen atau teks yang digunakan.
2. Menentukan inisialisasi awal untuk nilai , , , , dan .
3. Menghitung matriks dengan rumus:

|  |  |
| --- | --- |
|  |  |

**Keterangan :**

|  |  |
| --- | --- |
|  | = elemen matriks data ke-ij |
|  | = kelas atau label data ke-i |
|  | = kelas atau label data ke-j |
|  | = turunan batas teoritis |
|  | = fungsi kernel |
|  |  |

1. Untuk data ke – gunakan persamaan berikut ini :

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 𝛿𝛼𝑖 | = min{𝑚𝑎𝑥[𝛾(1−𝐸𝑖),−𝛼𝑖],𝐶−𝛼𝑖} |  |
| 𝛼𝑖 | = 𝛼𝑖+𝛿𝛼𝑖 |  |

**Keterangan :**

|  |  |
| --- | --- |
|  | **=** nilai error data ke - i |
|  | = tingkat pembelajaran |
|  | **=** nilai maksimum diagonal matriks hessian |
|  |  |

1. Mencari nilai bias (b) dengan menggunakan persamaan :

|  |  |
| --- | --- |
| 𝑏=[ w . + w . ] |  |

1. Pengujian pada dokumen yang diuji
2. Perhitungan keputusan sebagai berikut:

|  |  |
| --- | --- |
|  |  |
|  |  |

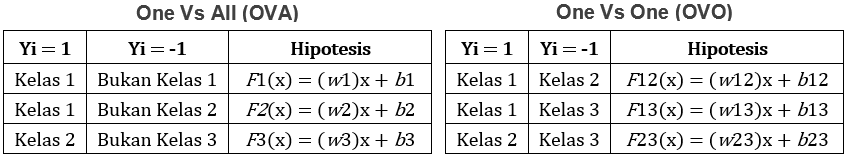
1. Jika hasil perhitungan keputusan lebih dari sama dengan 0 maka nilai adalah +1, maka termasuk kelas positif dan jika hasil perhitungan keputusan kurang dari 0 maka nilai nilai -1, maka termasuk kelas negatif. Perhitungan keputusan dengan menggunakan persamaan :

|  |  |
| --- | --- |
| ℎ(𝑥) = 𝑤 . 𝑥 + 𝑏 |  |
| atau |
|  |

Fungsi kernel umum yang digunakan dalam SVM (Prangga, 2017)antara lain:

|  |  |  |
| --- | --- | --- |
| Linear : |  |  |
| RBF/Gaussian : |  |  |
| Polynomial : |  |  |

Dalam proses klasifikasi, jarang ditemui kasus klasifikasi linear pada kehidupan nyata, melainkan lebih ke *non* linear (lebih dari dua kelas). Oleh karena itu, SVM dikembangkan agar dapat melakukan klasifikasi lebih dari dua kelas yang dinamakan *multi class* SVM.



**Gambar II.14** Pendekatan multi class

Pada Gambar II.14, terdapat dua pendekatan dalam *multi class*(Alita et al., 2020), yaitu:

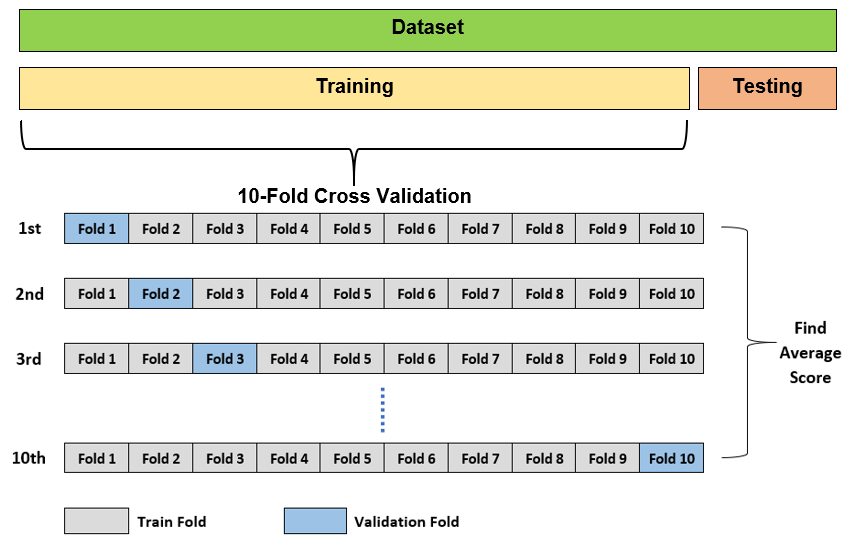
1. *One Versus One* (OVO), yaitu strategi yang terdiri dari satu *classifier* per kelas, yang dilatih untuk membedakan sampel satu kelas dari sampel kelas lainnya. Prinsip dari strategi ini, yaitu dibangun (*k*−1)2 buah model klasifikasi biner (*k* adalah jumlah kelas). Setiap model klasifikasi dilatih pada data dari dua kelas. Selanjutnya, dilakukan kernelisasi pada set data berdimensi (*N*x1) dari fitur dimensi lama sehingga mendapatkan didapatkan dimensi baru (*NxN*), dimana *N* adalah banyaknya data.
2. *One Versus All* (OVA), yaitu strategi yang terdiri dari satu *classifier* per kelas, yang dilatih untuk membedakan sampel satu kelas dari sampel semua kelas yang tersisa. Prinsip dari strategi ini adalah dibangun *k* buah model klasifikasi biner (*k* adalah jumlah kelas). Setiap model klasifikasi ke-*i* dilatih dengan menggunakan keseluruhan data, untuk mencari solusi permasalahan. Selanjutnya, dilakukan kernelisasi pada set data berdimensi (*N*x1) dari fitur dimensi lama sehingga mendapatkan didapatkan dimensi baru (*NxN*), dimana *N* adalah banyaknya data.

Dalam penelitian ini, *multi class* SVM akan digunakan sebagai metode klasifikasi untuk permasalahan sentimen tiga kelas, yaitu negatif, netral, dan positif. Pendekatan *multi class* yang digunakan adalah OVO dan OVA yang akan dibandingkan hasilnya. Untuk fungsi kernel yang digunakan, kernel RBF memiliki kelebihan, yaitu secara otomatis menentukan nilai, lokasi dari *center* serta nilai pembobot dan bisa mencakup nilai rentang tak terhingga. *Hyperparameter* yang digunakan dalam kernel RBF adalah *C* dan *gamma* (Ningrum, 2018). Kernel Gaussian/RBF juga efektif menghindari *overfitting* dengan memilih nilai yang tepat untuk parameter *C* dan *gamma* serta RBF baik digunakan ketika tidak ada pengetahuan terdahulu(Prangga, 2017). Kernel RBF juga memiliki performa lebih baik dibanding kernel Linear dan Polynomial(Putra et al., 2020). *Multi class* SVM dibangun dengan bahasa pemrograman Python dan memanfaatkan *library*, yaitu *Pandas*, *Scikit-Learn*, dan lainnya.

## 10-*Fold Cross Validation*

*K-fold cross validation* (CV) adalah salah satu metode yang dapat memeriksa *overfitting* pada suatu model(Nasution & Hayaty, 2019). Pada *k-fold* CV, model yang telah dibuat dibagi menjadi *k* bagian yang sama atau mendekati ukurannya (*fold*). Performa model akan diuji menggunakan data uji pada setiap fold yang sudah dilatih sebelumnya dengan data latih, dan berlanjut ke *fold* selanjutnya sampai selesai(Nasution & Hayaty, 2019). Hasil pengukuran dari masing-masing model *k* kemudian ditotalkan dan dibagi sebanyak jumlah *k* agar didapatkan hasil rata-rata (mean) (Muhidin & Burhan, 2018). *K-fold* CV yang sering digunakan adalah 10-*fold* CV.

Pada *10-fold* CV, *training* dataset dibagi menjadi 10 *fold*. Pada iterasi pertama, *fold* pertama sebagai *validation fold* dan sisanya berfungsi sebagai *train fold*. Pada iterasi kedua, *fold* kedua sebagai *validation fold* dan sisanya berfungsi sebagai *train fold*. Proses ini diulangi sampai setiap *fold* dari 10 *fold* telah digunakan sebagai *validation fold*. Untuk ilustrasi *10-fold* CV dapat dilihat pada Gambar II.15 dibawah ini.



**Gambar II.15** 10-fold cross validation

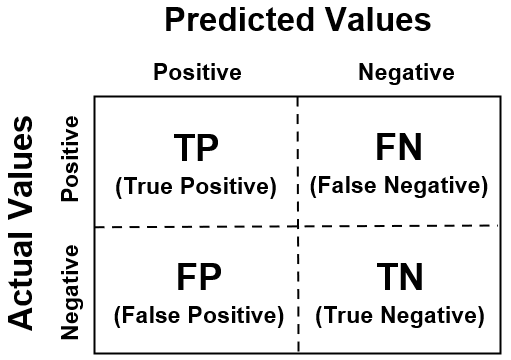
Dalam penelitian ini, *10-fold* CV digunakan sebagai evaluasi terhadap data *training* pada model yang sedang dibangun, serta evaluasi terhadap hasil *tuning hyperparameter* yang dilakukan. *10-fold* CV dibangun dengan bahasa pemrograman Python dan memanfaatkan *library*, yaitu *Pandas*, *Scikit-Learn*, dan lainnya.

## *Confusion Matrix*

*Confusion matrix* merupakan suatu instrumen yang digunakan untuk mengevaluasi performa dari model klasifikasi yang telah dihasilkan. Pada *confusion matrix*, hasil kelas klasifikasi akan dibandingkan dengan kelas data yang sebenarnya. Hasil tersebut kemudian akan digunakan untuk menghitung nilai *accuracy*, *precision*, *recall*, dan *f1-score*(Husada & Paramita, 2021).

Terdapat empat istilah representasi dari hasil pengukuran pada confusion matrix dan dapat dilihat pada Gambar II.16 dibawah ini, antara lain (Buntoro, 2017):

1. *True Positive* (TP), yaitu data positif yang diklasifikasikan benar.
2. *True Negative* (TN), yaitu data negatif yang diklasifikasikan benar.
3. *False Positive* (FP), yaitu data negatif yang diklasifikasikan salah.
4. *False Negative* (FN), yaitu data positif yang diklasifikasikan salah.



**Gambar II.16** Confusion matrix

Untuk mengukur performa *confusion matrix*, terdapat empat parameter yang dapat digunakan antara lain:

1. *Accuracy*, yaitu penggambaran seberapa akurat model dalam mengklasifikasikan dengan benar(Putra Nuansa, 2017).
2. *Precision*, yaitu penggambaran akurasi antara data yang diminta dengan dengan hasil klasifikasi yang diberikan oleh model klasifikasi(Putra Nuansa, 2017).
3. *Recall*, yaitu penggambaran keberhasilan model klasifikasi dalam menemukan kembali sebuah informasi(Putra Nuansa, 2017).
4. *F1-score*, yaitu *harmonic mean* dari nilai *recall* dan *precision* untuk mengetahui seberapa presisi dan handalnya performa model dalam mengklasifikasikan kelas (Alita et al., 2020).

Berikut adalah rumus dalam mencari empat paramater pengukur performa *confusion matrix*(Nasution & Hayaty, 2019):

|  |  |  |
| --- | --- | --- |
|  |  | (15) |
|  |  | (16) |
|  |  | (17) |
|  |  | (18) |

Dalam penelitian ini, *confusion matrix* akan dijadikan evaluasi terhadap data *testing* pada model yang telah dibangun/dilatih. Kriteria evaluasi yang dipertimbangkan adalah *accuracy*, diikuti dengan *precision*, *recall*, dan *f1-score*. *Confusion matrix* dibangun dengan bahasa pemrograman Python dan memanfaatkan *library*, yaitu *Pandas*, *Scikit-Learn*, dan lainnya.

# METODOLOGI PENELITIAN

## Alat Penelitian

Alat untuk mendukung pengerjaan dalam menyusun sebuah laporan penelitian. Berikut adalah alat yang digunakan dalam penelitian.

### Perangkat Keras

Perangkat keras yang digunakan dalam penelitian ini adalah Laptop ASUS ROG dengan spesifikasi:

1. *Processor* : Intel® Core™ i7-8750H Processor 2.2 GHz (9M Cache, up to 4.1 GHz)
2. *Memory* : 16 GB DDR4, 2666Mhz
3. *Graphics* : NVIDIA® GeForce® GTX1050Ti 4GB GDDR5 VRAM

### Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini antara lain:

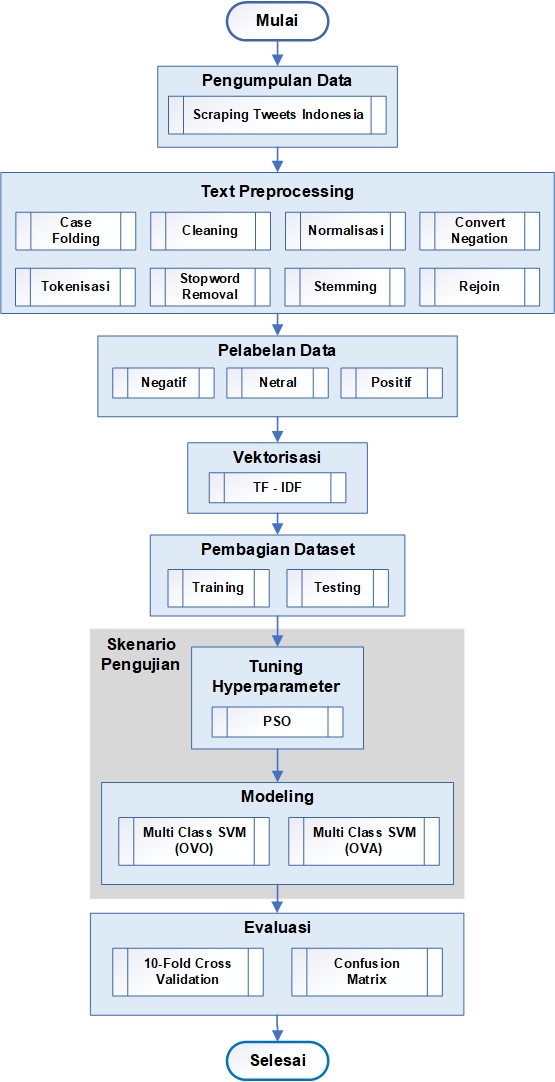
1. *Operating System* : Microsoft Windows 11 Home
2. *Aplication* : JupyterLab, Pycharm, Heroku, Git Bash

## Data Penelitian

Data penelitian yang digunakan pada penelitian ini adalah *tweets* berupa teks terkait sentimen publik terhadap kebijakan Kemendikbud antara lain KIP (Kartu Indonesia Pintar), bantuan UKT (Uang Kuliah Tunggal), dana BOS (Bantuan Operasional Sekolah), kuota belajar, sekolah daring, dan kuliah daring. Tweets ini digunakan sebagai data untuk melatih dan menguji model klasifikasi yang akan dibangun.

## Langkah Penelitian

Terdapat delapan tahap metode penelitian yang akan dilakukan pada penelitian ini yang dapat dilihat pada Gambar III.1 dibawah ini.



**Gambar III.1** Metodologi penelitian

### Pengumpulan Data

Dalam pengumpulan data, dilakukan *scraping tweets* berbahasa Indonesia terhadap Twitter dengan kata kunci pencarian berupa *plaintext*. Untuk seleksi data, *tweets* akan disaring agar tidak terdapat duplikasi *tweets* atau lebih didalamnya (*retweets*) dan *tweet* yang dipakai memiliki topik sesuai dengan penelitian yang dilakukan.

### *Text Preprocessing*

Setiap data *tweet* dilakukan *text preprocessing* agar dataset mentah tersebut dapat menjadi dataset yang sesuai dengan format yang dibutuhkan dalam klasifikasi. Proses dalam *text preprocessing* antara lain *case folding*, *cleaning*, normalisasi, *convert negation*, tokenisasi, *stopword removal*, *stemming*, dan *rejoin*.

### Pelabelan Data

Sebelum dilabeli, setiap dataset hasil *text preprocessing* dilakukan *remove duplicate* agar mengurangi *noise*. Kemudian, setiap data diberikan label apakah sentimen bersifat positif, negatif ataupun netral.

### Vektorisasi

Vektorisasi, yaitu dataset yang sudah dilabeli sebelumnya diubah ke dalam vektor (diberikan nilai pembobot untuk setiap data agar bisa dihitung dan diolah oleh klasifikasi). Vektorisasi dilakukan dengan *Term Frequency-Inverse Document Frequency* (TF-IDF).

### Pembagian Dataset

Dataset yang telah divektorisasi sebelumnya akan dibagi menjadi dua bagian, yaitu *training dataset* untuk melatih model klasifikasi yang dibangun dan *testing dataset* untuk menguji model klasifikasi yang telah dibangun. Jumlah *training dataset* yang digunakan sebesar 90% dan *testing dataset* sebesar 10%.

### *Tuning Hyperparameter*

*Tuning hyperparameter* merupakan pemilihan *hyperparameter* yang optimal pada model klasifikasi yang akan dibuat, yaitu *multi class* SVM kernel RBF dengan menggunakan PSO (*Particle Swarm Optimization*). PSO akan mengkombinasikan *hyperparamater-hyperparameter* yang dimasukkan dan mencari kombinasi *hyperparameter* dengan nilai skor (akurasi rata-rata dari 10-*fold* CV) terbaik sebagai solusi optimal. *Tuning hyperparameter* dilakukan pada *hyperparameter* C dan *gamma*.

### *Modelling*

Metode klasifikasi yang akan digunakan untuk membangun model adalah SVM dengan pendekatan *multi class*, yaitu One Versus One (OVO) dan *One Versus All* (OVA), serta kernel SVM yang digunakan adalah RBF/*gaussian*.

### Skenario Pengujian

Pada penelitian ini, akan dilakukan skenario pembangunan model berdasarkan *tuning hyperparameter* PSO dan pendekatan *multi class* SVM yang dibangun pada Tabel III.1 dibawah ini.

**Tabel III.1** Skenario Pengujian

|  |  |  |
| --- | --- | --- |
| Skenario | Tuning Hyperparameter PSO | Multi Class SVM (Kernel RBF) |
| 1 | Tidak | OVO |
| 2 | Ya | OVO |
| 3 | Tidak | OVA |
| 4 | Ya | OVA |
|  |  |  |

### Evaluasi

10-*fold cross validation* (CV) digunakan sebagai evaluasi data *training* terhadap skenario model yang sedang dibangun. Tolak ukur evaluasi ini adalah nilai rata-rata dari *accuracy*, *precision*, *recall*, dan *f1-score* dalam 10-*fold* CV. Sedangkan, c*onfusion matrix* digunakan sebagai evaluasi data *testing* terhadap skenario model yang telah dibangun/dilatih sebelumnya. Label hasil klasifikasi model akan dibandingkan dengan label aktualnya. Dari *confusion matrix* akan diperoleh tolak ukur antara lain *accuracy*, *precision*, *recall*, dan *f1-score*.

# IMPLEMENTASI DAN HASIL

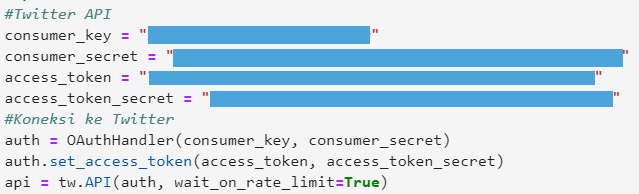
## Implementasi

Setelah dipaparkan metodologi penelitian yang akan dilakukan, langkah selanjutnya adalah implementasi dari metodologi penelitian tersebut.

### Pengumpulan Data

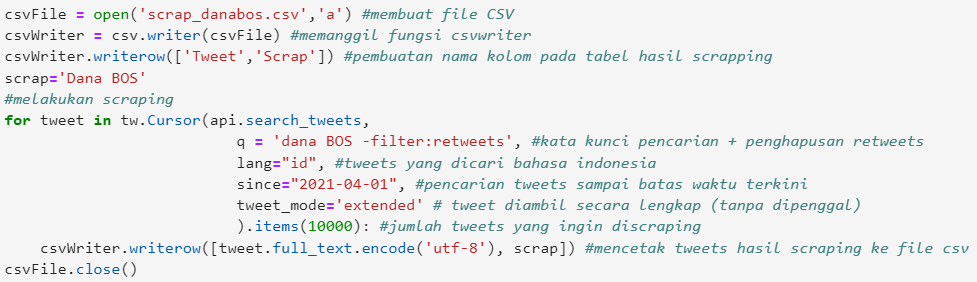
Pengumpulan data dilakukan dengan scraping Twitter dengan memanfaatkan Twitter API. Data yang diambil adalah *tweets* berupa teks yang dilakukan selama tujuh minggu, yaitu dari tanggal 03 April 2021 sampai dengan 20 Mei 2021. Kata kunci dalam pencarian tweet berbahasa Indonesia antara lain ”sekolah daring”, ”kuliah daring”, ”kuota belajar”, ”dana BOS”, ”UKT”, dan ”KIP”.

**Kode Program IV.1** Twitter API



Pada Kode Program IV.1, dilakukan inisialisasi *key* dan token yang dibutuhkan dalam menggunakan Twiiter API. Kemudian, *key* dan token tersebut digunakan untuk koneksi ke Twitter API agar dapat melakukan *scraping*.

**Kode Program IV.2** Scraping



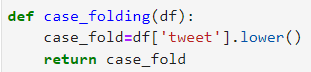
Pada Kode Program IV.2, *scraping* dilakukan dengan memasukkan kata kunci pencarian diikuti *filter retweets*, bahasa *tweet* (Indonesia), waktu dimulainya pencarian *tweet*, panjang *tweet*, dan jumlah *tweet* maksimal yang ingin diambil. Setelah dilakukan *scraping*, setiap *tweet* akan disimpan dalam *file* berformat .csv.

### *Text Preprocessing*

Setelah dilakukan *scraping*, *tweets* tersebut belum dapat dipakai dalam klasifikasi yang akan dilakukan nantinya. Oleh karena itu, perlu dilakukan *text* *preprocessing* terlebih dahulu agar teks yang disiapkan lebih terstuktur.

#### *Case Folding*

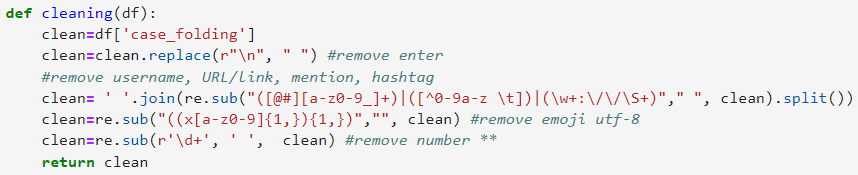
**Kode Program IV.3** Case Folding



Pada Kode Program IV.3, setiap huruf dalam teks diubah menjadi *lowercase* (menjadi huruf kecil) secara keseluruhan.

#### *Cleaning*

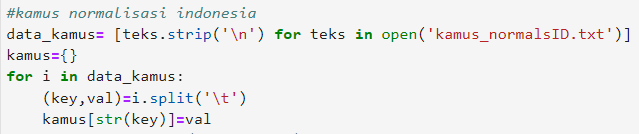
**Kode Program IV.4** Cleaning



Pada Kode Program IV.4, setiap teks dilakukan penghapusan karakter, yaitu *enter*, *username*, URL/link, tanda baca/delimeter, *mention*, *hashtag*, emoji, dan angka.

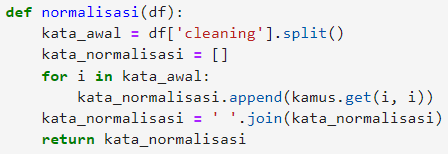
#### Normalisasi

**Kode Program IV.5** Kamus Normalisasi Kata



Pada Kode Program IV.5, dibuat sebuah kamus normalisasi secara manual yang terdiri dari 3095 pasangan kata. Pasangan kata ini adalah kata asal (kata sebelum normalisasi) dan kata tujuan (kata setelah normalisasi). Tiap kata asal dan kata tujuan tersebut dipisahkan dengan *tab*. Kamus normalisasi ini berasal dari pasangan kata yang disimpan dalam *file* berformat *.txt*. Untuk contoh kamus normalisasi terlampir pada Lampiran A.

**Kode Program IV.6** Normalisasi



Pada Kode Program IV.6, setiap teks akan dilakukan normalisasi kata awalnya (kata asal) untuk diubah ke kata tujuan yang ada di dalam kamus normalisasi yang dibuat sebelumnya.

#### *Convert Negation*

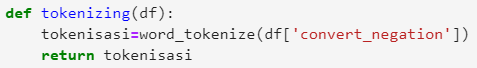
**Kode Program IV.7** Convert Negation



Pada Kode Program IV.7, setiap teks dilakukan *convert negation*, yaitu menggabungkan kata-kata negasi (’tidak’, ’tanpa’, ’jangan’, ’belum’) dengan kata sesudahnya. Kata setelah negasi akan digabungkan jika kata tersebut bukanlah kata negasi dan tidak berada dalam *stoplist* yang dibuat.

#### Tokenisasi

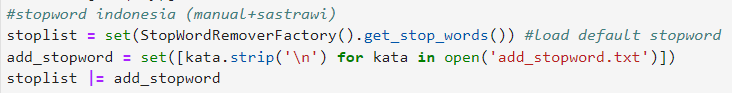
**Kode Program IV.8** Tokenisasi



Pada Kode Program IV.8, setiap teks dilakukan tokenisasi hingga menjadi kumpulan token/*term* yang akan digunakan dalam melakukan *stopword removal* dan *stemming*. Untuk tokenisasi tersebut menggunakan *library* NLTK, yaitu fungsi *word\_tokenize()*.

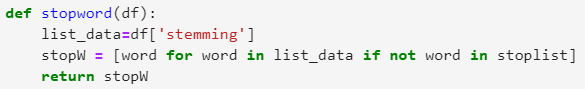
#### *Stopword Removal*

**Kode Program IV.9** Inisialisasi Stoplist



Pada Kode Program IV.9, dibuat sebuah *stoplist* yang terdiri dari 1645 kata. Kumpulan kata tersebut merupakan gabungan dari *stopword* PySastrawi dan *stopword* yang dibuat secara manual. Untuk contoh *stoplist* yang digunakan terlampir pada Lampiran A.

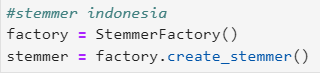
**Kode Program IV.10** Stopword Removal



Pada Kode Program IV.10, setiap token-token dilakukan pengecekan terlebih dahulu. Apabila token tersebut terdapat dalam *stoplist*, maka token tersebut akan dibuang. Jika tidak, token tersebut tetap disimpan.

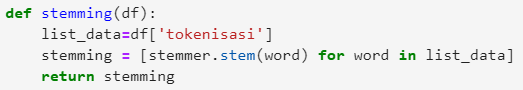
#### *Stemming*

**Kode Program IV.11** Inisialisasi Stemmer



Pada Kode Program IV.11, dilakukan inisialisasi untuk proses *stemming* yang dilakukan menggunakan *library* PySastrawi. Fungsi yang dibutuhkan adalah *StemmerFactory()* dan *create\_stemmer()*.

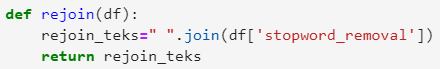
**Kode Program IV.12** Stemming



Pada Kode Program IV.12, setiap token-token dilakukan *stemming* dengan fungsi *stem()*, yaitu mencari kata dasar dari tiap token tersebut.

#### *Rejoin*

**Kode Program IV.13** Rejoin



Pada Kode Program IV.13, setiap token-token digabungkan kembali menjadi kalimat utuh dengan fungsi *join()*. Ini dilakukan agar dapat digunakan untuk proses *remove duplicate* dan pelabelan data.

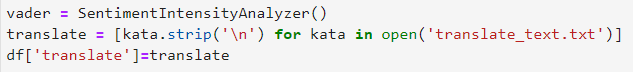
### Pelabelan Data

**Kode Program IV.14** Remove Duplicate

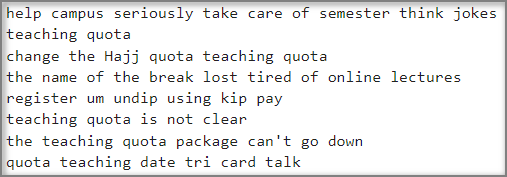


Pada Kode Program IV.14, dilakukan proses menghapus duplikasi data hasil *text preprocessing* sebelumnya agar mengurangi *noise* dalam dataset sebelum dilabeli.

**Kode Program IV.15** VADER dan Teks Terjemahan

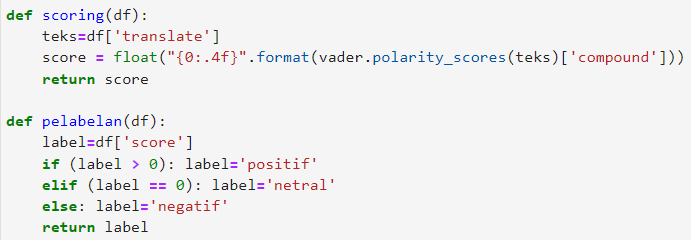


Pada Kode Program IV.15, dibuat inisialisasi fungsi pelabelan oleh Vader dan terjemahan teks bahasa inggris dari data hasil *text preprocessing* sebelumnya (sudah melewati *remove duplicate*). Hasil terjemahan tersebut digunakan untuk mencari skor sentimen oleh Vader nantinya. Contoh hasil terjemahan dapat dilihat pada Gambar IV.1 dibawah ini.



**Gambar IV.1** Terjemahan teks rejoin

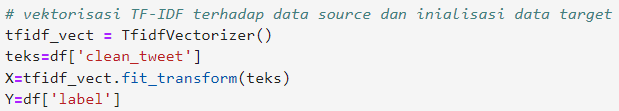
**Kode Program IV.16** Skor dan Label Sentimen



Pada Kode Program IV.16, setiap teks hasil terjemahan dilakukan *scoring* berupa perhitungan intesitas sentimen oleh Vader. Setelah skor diperoleh, teks akan dilabeli berdasarkan skor masing-masing, yaitu ’positif’, ’netral’, ataupun ’negatif’. Skor dengan nilai diatas 0 akan dilabeli ’ positif’, skor dengan nilai sama dengan 0 dilabeli ’ netral’, dan skor dengan nilai dibawah 0 dilabeli ’negatif’.

### Vektorisasi

**Kode Program IV.17** Vektorisasi TF-IDF



Pada Kode Program IV.17, setiap teks dalam dataset dilakukan vektorisasi berupa TF-IDF dengan menggunakan *library* Scikit-Learn, yaitu fungsi *Tfidfvectorizer()* dan *fit\_transform()*. Setelah divektorisasi, hasilnya akan disimpan untuk dijadikan *source data* untuk pembelajaran mesin. Kemudian, dilakukan inisialisasi pada setiap label dalam dataset sebagai *target* *data* pembelajaran mesin.

### Pembagian Dataset

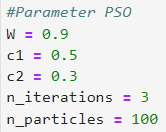
**Kode Program IV.18** Pembagian Dataset



Pada Kode Program IV.18, pembagian dataset menjadi *training data* dan *testing data* dilakukan dengan *library* Scikit-Learn, yaitu fungsi *train\_test\_split()*. Pembagian dataset tersebut berupa 90% untuk training dataset dan 10% untuk testing dataset.

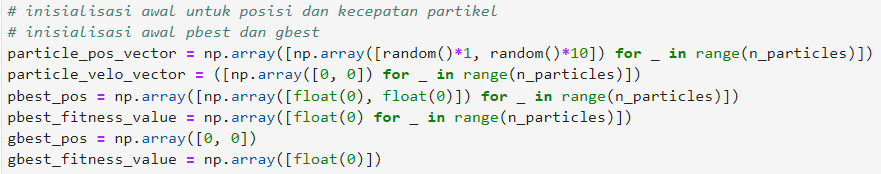
### *Tuning Hyperparameter*

**Kode Program IV.19** Inisialisasi Parameter PSO



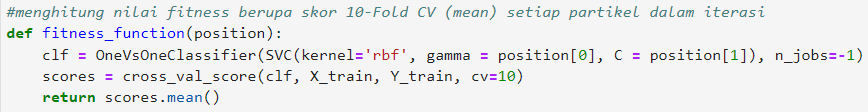
Pada Kode Program IV.19, dilakukan inisialisasi parameter-parameter yang digunakan dalam PSO. Parameternya antara lain berat inertia(*w*) bernilai 0.9, koefisien akselerasi pertama(*c1*) bernilai 0.5, koefisien akselerasi kedua(*c2*) bernilai 0.3, jumlah iterasi dilakukan 3 kali, dan jumlah partikel sebanyak 100.

**Kode Program IV.20** Inisalisasi Partikel PSO



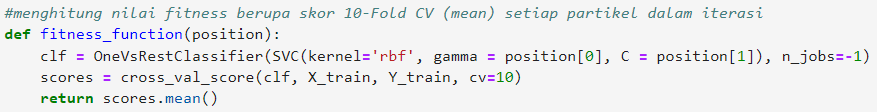
Pada Kode Program IV.20, dilakukan inisialisasi posisi dan kecepatan partikel awal, posisi dan fitness *pbest*, serta posisi dan fitness *gbest*. Posisi partikel awal, posisi *pbest* , dan posisi *gbest* adalah *array* yang menyimpan kombinasi nilai *hyperparameter gamma* dan *C*. Sedangkan, *fitness pbest* dan *fitness* *gbest* adalah *array* yang menyimpan nilai evaluasi tiap kombinasi *hyperparameter*, yaitu skor (nilai *accuracy* rata-rata) oleh 10-fold CV. Kecepatan partikel awal adalah *array* yang dibutuhkan untuk memperbaharui posisi partikel di iterasi selanjutnya. Kecepatan partikel awal, posisi *pbest* dan *gbest*, serta *fitness pbest* dan *gbest* diisi dengan nilai 0. Sedangkan, posisi partikel awal diisi nilai acak dengan rentang 0 sampai 1 untuk *hyperparameter gamma* dan nilai acak dengan rentang 0 sampai 10 untuk *hyperparameter C*.

**Kode Program IV.21** Fitness Function OVO



Pada Kode Program IV.21, *fitness candidate* dibangun untuk skenario kedua dimana fungsi *multi class* SVM pendekatan OVO dengan kernel RBF, *gamma* berupa nilai indeks pertama pada posisi partikel, dan *C* berupa nilai indeks kedua pada posisi partikel.

**Kode Program IV.22** Fitness Function OVA



Pada Kode Program IV.22, *fitness candidate* dibangun untuk skenario keempat dimana fungsi *multi class* SVM pendekatan OVA dengan kernel RBF, *gamma* berupa nilai dari indeks pertama pada posisi partikel, dan *C* berupa nilai indeks kedua pada posisi partikel.

**Kode Program IV.23** Algoritma PSO



Pada Kode Program IV.23, algoritma PSO dimulai dengan membandingkan nilai *fitness candidate* dari tiap partikel terhadap *fitness pbest*. Apabila *fitness pbest* lebih kecil dibanding *fitness candidate*, maka nilai *fitness candidate* menjadi nilai *fitness pbest* yang baru dan nilai posisi partikel menjadi nilai posisi *pbest* yang baru. Kemudian, dilakukan perbandingan nilai *fitness pbest* dari tiap partikel terhadap *fitness gbest*. Apabila *fitness gbest* lebih kecil dibanding *fitness pbest*, maka nilai *fitness pbest* menjadi nilai *fitness gbest* yang baru dan nilai posisi *pbest* menjadi nilai posisi *gbest* yang baru. Terakhir, dilakukan *update* terhadap kecepatan dan posisi partikel yang akan digunakan pada iterasi selanjutnya. Langkah-langkah tersebut akan diulangi kembali hingga mencapai iterasi maksimal.

**Kode Program IV.24** Menampilkan Gbest

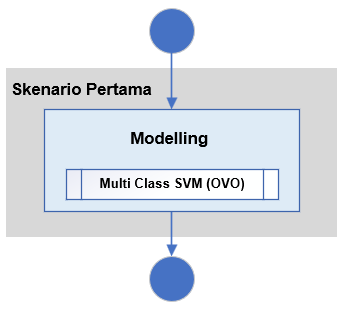


Pada Kode Program IV.24, hasil dari *tuning hyperparameter* PSO yang diambil adalah posisi *gbest*. Posisi *gbest* tersebut adalah kombinasi *hyperparameter* (*gamma* dan *C*) terbaik berdasarkan evaluasi yang dilakukan. Posisi *gbest* dari *multi class* SVM (OVO) akan digunakan pada skenario kedua dan posisi *gbest* dari *multi class* SVM (OVA) akan digunakan pada skenario keempat.

### *Modelling*

#### Skenario Pertama

Skenario pertama adalah model yang dibangun dengan metode klasifikasi *multi class* SVM pendekatan OVO dan tidak menggunakan *tuning hyperparameter* PSO (lihat pada Gambar IV.2 dibawah ini).



**Gambar IV.2** Model skenario pertama

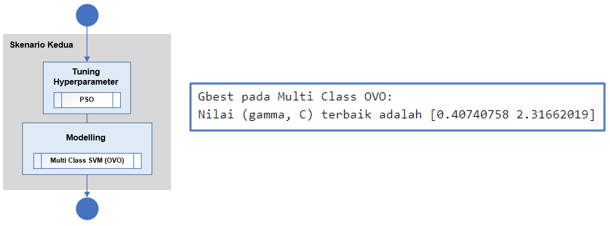
**Kode Program IV.25** Model Skenario Pertama



Pada Kode Program IV.25, skenario pertama dibangun dengan *multi class* SVM pendekatan OVO dengan kernel RBF serta *gamma* dan *C* bernilai *default*.

#### Skenario Kedua

Skenario kedua adalah model yang dibangun dengan metode klasifikasi *multi class* SVM pendekatan OVO dan menggunakan *tuning hyperparameter* PSO (lihat pada Gambar IV.3 dibawah ini).



**Gambar IV.3** Skenario model kedua

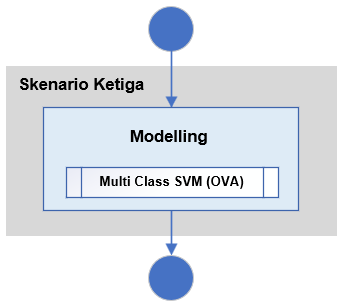
**Kode Program IV.26** Model Skenario Kedua



Pada Kode Program IV.26, skenario kedua dibangun dengan *multi class* SVM pendekatan OVO dengan kernel RBF dan hasil *tuning hyperparameter*, yaitu *gamma* bernilai 0.40740758 dan *C* bernilai 2.31662019.

#### Skenario Ketiga

Skenario ketiga adalah model yang dibangun dengan metode klasifikasi *multi class* SVM pendekatan OVA dan tidak menggunakan *tuning hyperparameter* PSO (lihat pada Gambar IV.4 dibawah ini).



**Gambar IV.4** Skenario model ketiga

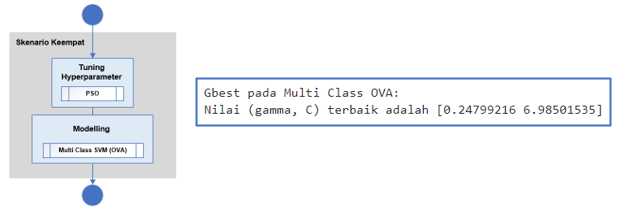
**Kode Program IV.27** Model Skenario Ketiga



Pada Kode Program IV.27, skenario ketiga dibangun dengan *multi class* SVM pendekatan OVA dengan kernel RBF serta *gamma* dan *C* bernilai *default*.

#### Skenario Keempat

Skenario keempat adalah model yang dibangun dengan metode klasifikasi *multi class* SVM pendekatan OVA dan menggunakan *tuning hyperparameter* PSO (lihat pada Gambar IV.5 dibawah ini).



**Gambar IV.5** Skenario model keempat

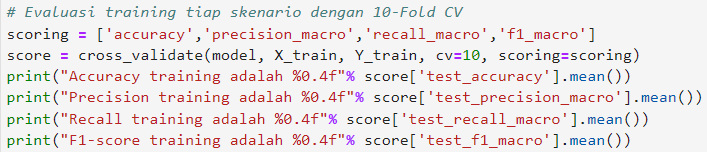
**Kode Program IV.28** Model Skenario Keempat



Pada Kode Program IV.28, skenario keempat dibangun dengan *multi class* SVM pendekatan OVA dengan kernel RBF dan hasil *tuning hyperparameter*, yaitu *gamma* bernilai 0.24799216 dan *C* bernilai 6.98501535.

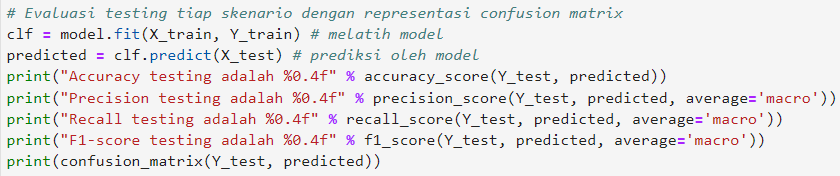
### Evaluasi

**Kode Program IV.29** Evaluasi Data Training Tiap Skenario



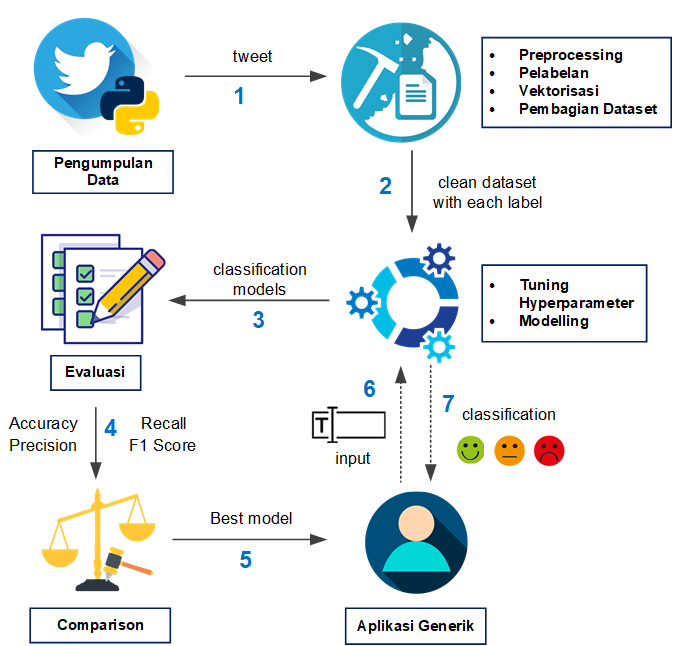
Pada Kode Program IV.29, dilakukan evaluasi data *training* terhadap tiap skenario model dengan 10-*fold* CV. 10-*fold* CV membagi *training* dataset awal menjadi *sub* *training set* dan *sub* *validation set* untuk pada tiap *fold* serta dilakukan perhitungan nilai tolak ukur rata-rata didalamnya, yaitu *accuracy*, *precision*, *recall*, dan *f1-score*.

**Kode Program IV.30** Evaluasi Data Testing Tiap Skenario



Pada Kode Program IV.30, dilakukan evaluasi data *testing* terhadap tiap skenario model. Model dilatih dengan data *training* dan dilanjutkan dengan klasifikasi terhadap data *testing*. Kemudian, membandingkan label aktual dengan label hasil klasifikasi model agar diperoleh nilai performa model, yaitu *accuracy*, *precision*, *recall*, dan *f1-score* serta representasi *confusion matrix* dari tiap skenario.

### Arsitektur Sistem



**Gambar IV.6** Arsitektur sistem

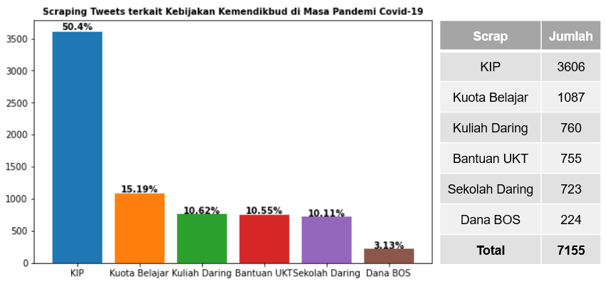
Pada Gambar IV.6, disajikan arsitektur sistem yang dibuat dalam penelitian ini. Pertama, pengumpulan data berupa tweets dilakukan terhadap Twitter dengan cara *scraping*. Kemudian, tweets tersebut *preprocessing*, pelabelan data, vektorisasi, dan pembagian dataset diperoleh dataset bersih yang sudah dilabeli dan terbagi menjadi *training* serta *testing dataset*. Setelah itu, dilakukan tuning hyperparameter ataupun *modeling* terhadap dataset tersebut ke dalam empat skenario model yang dirancang. Setiap skenario model dilakukan evaluasi data *training* dengan 10-*fold* CV dan evaluasi data *testing* dengan *confusion matrix* . Tolak ukur yang digunakan pada evaluasi adalah *accuracy*, *precision*, *recall*, dan *f1-score*. *Accuracy* sebagai nilai pembanding untuk mengecek *overfitting* pada *training* terhadap *testing*, serta sebagai acuan untuk menemukan model dengan performa terbaik. Model dengan performa terbaik digunakan sebagai model klasifikasi untuk aplikasi generik yang dibangun. Aplikasi generik tersebut dirancang agar *user* dapat melakukan percobaan dengan memasukkan teks dan model akan memberikan hasil klasifikasi berupa kelas sentimen (positif, netral, atau negatif).

## Hasil

Setelah implementasi dilakukan, maka akan dilakukan pemaparan berupa hasil dari implementasi tersebut. Hasil tersebut berupa gambar/grafik ataupun tabel yang mendukung implementasi yang telah dikerjakan sebelumnya.

### Pengumpulan Data

Pengumpulan data berupa *scraping* yang telah dilakukan menghasilkan sebesar 7155 data berdasarkan kata kunci pencarian serta terlampir pada Lampiran B. Kata kunci ”KIP” berjumlah 3606 data dengan persentase 50,4%, ”Kuota Belajar” berjumlah 1087 data dengan persentase 15,19%, ”Kuliah Daring” berjumlah 760 data dengan persentase 10,62%, ”Bantuan UKT” berjumlah 755 data dengan persentase 10,55%, ”Sekolah Daring” berjumlah 723 data dengan persentase 10,11%, dan ’”Dana BOS” berjumlah 224 data dengan persentase 3,13%. Hasil pengumpulan data juga ditampilkan pada Gambar IV.7 dibawah ini.



**Gambar IV.7** Hasil scraping

### *Text Preprocessing*

Hasil *text preprocessing* yang telah dilakukan terhadap 7155 data, yaitu *case folding*, *cleaning*, normalisasi, *convert negation*, tokenisasi, *stopword removal*, *stemming*, dan *rejoin* telah terlampir pada Lampiran B. Untuk melihat *text preprocessing*, diberikan sampel data pada Tabel IV.1 dibawah ini.

**Tabel IV.1** Text Preprocessing

|  |  |
| --- | --- |
| Teks | Preprocessing |
| b'Mindset kayak gini kok gak pernah berakhir ya. Padahal di masa pandemi ini anak\xc2\xb2 sekolah daring dan orangtua bisa merasakan bukan susahnya mengajari anaknya. Kok guru masih disuruh terima ikhlas aja \xf0\x9f\x98\x92 <https://t.co/oYTPOFZdAy>' | Default |
| b'mindset kayak gini kok gak pernah berakhir ya. padahal di masa pandemi ini anak\xc2\xb2 sekolah daring dan orangtua bisa merasakan bukan susahnya mengajari anaknya. kok guru masih disuruh terima ikhlas aja \xf0\x9f\x98\x92 [https://t.co/oytpofzday](https://t.co/oYTPOFZdAy)' | Case Folding |
| b mindset kayak gini kok gak pernah berakhir ya padahal di masa pandemi ini anak sekolah daring dan orangtua bisa merasakan bukan susahnya mengajari anaknya kok guru masih disuruh terima ikhlas aja | Cleaning |
| b polapikir seperti begini kok tidak pernah berakhir ya padahal di masa pandemi ini anak sekolah daring dan orangtua bisa merasakan bukan susahnya mengajari anaknya kok guru masih disuruh terima ikhlas aja | Normalisasi |
| b polapikir seperti begini kok tidakpernah berakhir ya padahal di masa pandemi ini anak sekolah daring dan orangtua bisa merasakan bukansusahnya mengajari anaknya kok guru masih disuruh terima ikhlas aja | Convert Negation |
| ['b', 'polapikir', 'seperti', 'begini', 'kok', 'tidakpernah', 'berakhir', 'ya', 'padahal', 'di', 'masa', 'pandemi', 'ini', 'anak', 'sekolah', 'daring', 'dan', 'orangtua', 'bisa', 'merasakan', 'bukansusahnya', 'mengajari', 'anaknya', 'kok', 'guru', 'masih', 'disuruh', 'terima', 'ikhlas', 'aja'] | Tokenisasi |
| ['polapikir', 'tidakpernah', 'pandemi', 'sekolah', 'daring', 'merasakan', 'bukansusahnya', 'mengajari', 'anaknya', 'guru', 'disuruh', 'terima', 'ikhlas'] | Stopword Removal |
| ['polapikir', 'tidakpernah', 'pandemi', 'sekolah', 'daring', 'rasa', 'bukansusahnya', 'ajar', 'anak', 'guru', 'suruh', 'terima', 'ikhlas'] | Stemming |
| polapikir tidakpernah pandemi sekolah daring rasa bukansusahnya ajar anak guru suruh terima ikhlas | Rejoin |

Pada Tabel IV.1, sampel data dari hasil *scraping* dilakukan *text preprocessing*. Pada *case folding*, sampel data diubah menjadi *lowercase* (semua huruf menjadi bentuk huruf kecil). Pada *cleaning*, sampel data dibersihkan dari (,), (’), (/), link/URL, dan emoji (\xc2\xb2 dan \xf0\x9f\x98\x92). Pada normalisasi, dilakukan normalisasi kata, yaitu ’gak’ menjadi ’tidak’, ’mindset’ menjadi ’polapikir’, dan ’kayak’ menjadi ’seperti’. Pada tahap convert negation, kata ’tidak’ dan kata ’pernah’ digabung menjadi ’tidakpernah’, serta kata ’bukan’ dan kata ’susahnya’ menjadi ’bukansusahnya’. Pada tahap tokenisasi, sampel data dipecah menjadi token-token. Pada *stopword removal*, sampel data hasil tokenisasi sebelumnya dilakukan penghapusan token yang berada dalam *stoplist*, yaitu ’b’, ’seperti’,’begini’, ’ya’, ’padahal’, ’di’, ’masa’, ’ini’, ’anak’, ’dan’, ’orangtua’, ’bisa’, ’masih’, dan ’aja’. Pada tahap *stemming*, dilakukan transformasi token-token menjadi bentuk dasar (penghapusan imbuhan), yaitu ’anaknya’ menjadi ’anak’. Kemudian, pada tahap *rejoin* sampel data berupa token-token digabungkan kembali menjadi kalimat utuh.

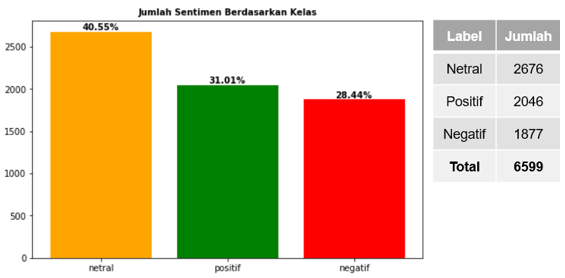
### Pelabelan Data



**Gambar IV.8** Hasil remove duplicate

Pada Gambar IV.8, disajikan gambar hasil *remove duplicate* dari data *text preprocessing* sebelumnya. Data yang awalnya berjumlah 7155 berubah menjadi 6599 data dikarenakan ditemukan duplikasi data pada hasil *text preprocessing* berjumlah 556 data. Data berjumlah 6599 hasil *remove duplicate* tersebut terlampir pada Lampiran B dan digunakan untuk pelabelan data.

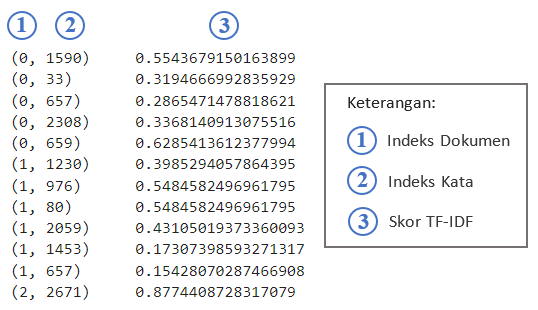
Pelabelan data yang memanfaatkan Vader menghasilkan data berlabel ’netral’ berjumlah 2676 data dengan persentase 40,55%, ’positif’ berjumlah 2046 data dengan persentase 31,01%, dan ’negatif’ berjumlah 1877 data dengan persentase 28,44%. Hasil pelabelan data juga terlampir pada Lampiran B serta ditampilkan pada Gambar IV.9 dibawah ini.



**Gambar IV.9** Hasil pelabelan data

Dari Gambar IV.9, hasil pelabelan juga menunjukkan bahwa terjadi *imbalanced data*, yaitu kelas ’netral’ dengan persentase 40,55%, sebagai kelas mayoritas serta kelas ’positif’ dengan persentase 31,01% dan ’negatif’ dengan persentase 28,44% sebagai kelas minoritas. Berdasarkan studi literatur terkait *imbalanced data*, maka dataset tersebut tidak dilakukan *balancing* data, yaitu pemerataan jumlah label dikarenakan *imbalanced data* yang terjadi dikategorikan rendah (*mild*).

### Vektorisasi



**Gambar IV.10** TF-IDF

Pada Gambar IV.10, ditampilkan hasil vektorisasi TF-IDF terhadap dataset yang digunakan. Terdapat tiga penanda dalam gambar tersebut, yaitu indeks dokumen, indeks kata, dan skor TF-IDF yang diperoleh.

### Pembagian Dataset

Pembagian dataset yang awalnya berjumlah 6599 dilakukan ke dalam dua bagian, yaitu *training* dataset berjumlah 5939 dan *testing* dataset berjumlah 660 dengan rasio 90%:10%. Kedua bagian dataset tersebut dibagi secara merata berdasarkan label, baik netral, positif, maupun negatif. Pada *training dataset* didalamnya data berlabel netral berjumlah 2408, positif berjumlah 1842, dan negatif berjumlah 1689. Sedangkan, pada *testing* *dataset* didalamnya data berlabel netral berjumlah 268, positif berjumlah 204, dan negatif berjumlah 188. Hasil pembagian dataset tersebut ditampilkan pada Tabel IV.2 dibawah ini.

**Tabel IV.2** Hasil Pembagian Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| Label | Jumlah Data | | Jumlah |
| Training | Testing |
| Netral | 2408 | 268 | 2676 |
| Positif | 1842 | 204 | 2046 |
| Negatif | 1689 | 188 | 1877 |
| Total | 5939 | 660 | 6599 |
|  |  |  |  |

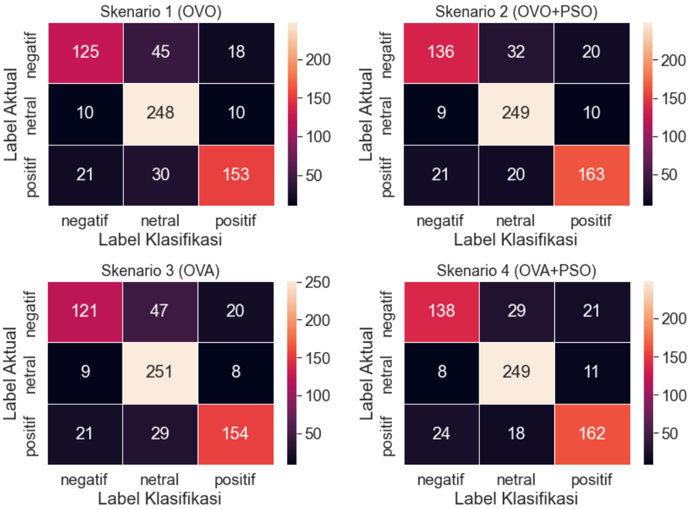
### *Tuning Hyperparameter*

**Tabel IV.3** Hasil Tuning Hyperparameter PSO

|  |  |  |
| --- | --- | --- |
| Multi Class SVM (Kernel RBF) | Tuning Hyperparameter PSO | |
| *gamma* | *C* |
| OVO | 0.40740758 | 2.31662019 |
| OVA | 0.24799216 | 6.98501535 |

Pada Tabel IV.3, ditampilkan hasil *tuning hyperparameter* PSO pada *multi* *class* SVM kernel RBF, baik pendekatan OVA maupun OVO. *Multi class* SVM pendekatan OVO dengan *tuning hyperparameter* PSO dilakukan pada skenario kedua dan *multi class* SVM pendekatan OVA dengan *tuning hyperparameter* PSO dilakukan pada skenario keempat. *Fitness candidate* partikel dan nilai *Gbest* yang diperoleh pada *multi class* SVM pendekatan OVA terlampir pada Lampiran C, serta *fitness candidate* partikel dan nilai *Gbest* yang diperoleh pada *multi class* SVM pendekatan OVO terlampir pada Lampiran D.

### Evaluasi



**Gambar IV.11** Confusion matrix data testing tiap skenario

Pada Gambar IV.11, ditampilkan hasil evaluasi testing terhadap tiap skenario model dalam representasi *confusion matrix* dan juga *classification report* yang terlampir pada Lampiran E. Kemudian, untuk hasil evaluasi *training* dalam performa rata-rata (*mean*) dengan 10-*fold* CV dan evaluasi *testing* dengan *confusion matrix* dirangkum dan ditampilkan pada Tabel IV.4 dibawah ini dengan tolak ukur *accuracy*, *precision*, *recall*, dan *f1-score*.

**Tabel IV.4** Hasil Evaluasi Tiap Skenario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tolak Ukur | Skenario | Hasil Evaluasi  Multi Class SVM (Kernel RBF) | | Gap  (Jarak) |
| Train | Test |
| Accuracy | 1 (OVO) | 76,86% | 79,70% | +2.84% |
| 2 (OVO+PSO) | 77,94% | 83,03% | +5.09% |
| 3 (OVA) | 76,66% | 79,70% | +3.04% |
| 4 (OVA+PSO) | 78,62% | 83,18% | +4.56% |
| Precision | 1 (OVO) | 77,06% | 80,48% | +3.42% |
| 2 (OVO+PSO) | 77,50% | 83,04% | +5.54% |
| 3 (OVA) | 76,85% | 80,50% | +3.65% |
| 4 (OVA+PSO) | 78,06% | 82,93% | +4.87% |
| Recall | 1 (OVO) | 74,91% | 78,01% | +3.10% |
| 2 (OVO+PSO) | 76,26% | 81,72% | +5.46% |
| 3 (OVA) | 74,67% | 77,84% | +3.17% |
| 4 (OVA+PSO) | 77,18% | 81,91% | +4.73% |
| F1-score | 1 (OVO) | 75,38% | 78,69% | +3.31% |
| 2 (OVO+PSO) | 76,57% | 82,16% | +5.59% |
| 3 (OVA) | 75,14% | 78,52% | +3.38% |
| 4 (OVA+PSO) | 77,43% | 82,27% | +4.84% |

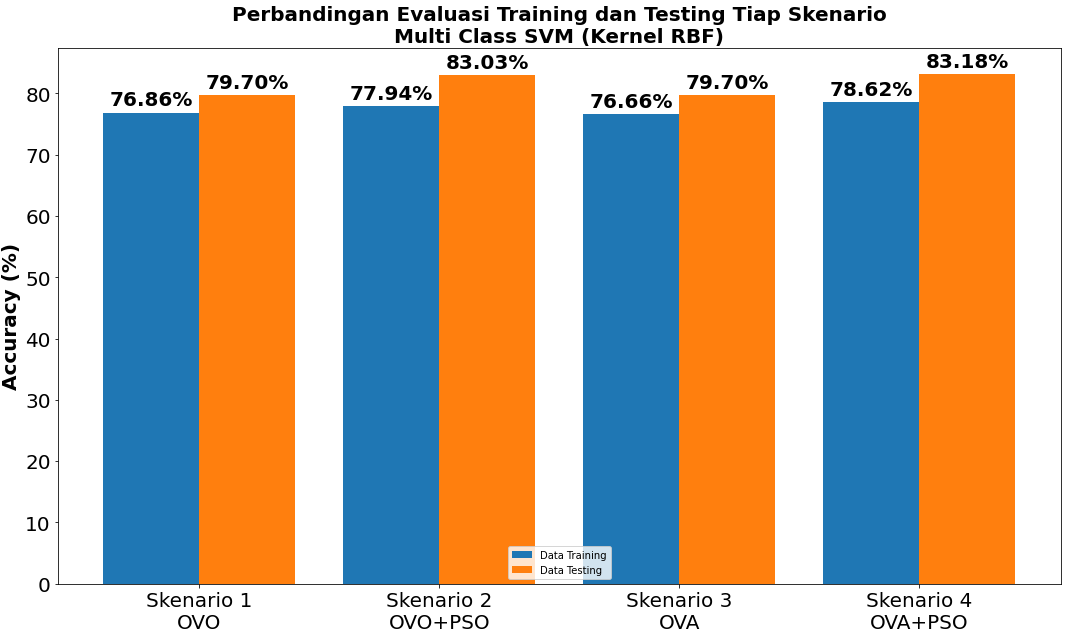
Pada Tabel IV.4, ditampilkan hasil evaluasi tiap skenario, baik evaluasi training dengan 10-fold CV maupun testing dengan confusion matrix. Train adalah hasil evaluasi data training menggunakan 10-Fold CV dan test adalah hasil evaluasi data testing menggunakan confusion matrix. Sedangkan, gap adalah jarak perbedaan nilai performa train terhadap test. Jika performa training lebih tinggi dibanding testing, maka nilai gap diberikan tanda (-). Sedangkan, jika performa training lebih rendah dibanding testing, maka nilai gap diberikan tanda (+). Berdasarkan hasil tersebut, terdapat beberapa poin yang diperoleh antara lain:

PSO dapat meningkatkan nilai evaluasi *training* dan *testing* terhadap tiap skenario model. Ini dikarenakan PSO sebagai *tuning hyperparameter* melakukan optimasi terhadap *hyperparameter* C dan *gamma* agar menemukan *hyperlane* pada SVM yang lebih dinamis. Semakin dinamis *hyperlane* yang terbentuk, maka semakin baik juga model dalam mengklasifikasikan data. Pada evaluasi *training*, terjadi kenaikan 1,08 hingga 1,96% pada *accuracy*, 0,44 hingga 1,21% pada *precision*, 1,35 hingga 2,51% pada *recall*, dan 1,19 hingga 2,29% pada *f1-score*. Sedangkan pada evaluasi *testing*, terjadi kenaikan 3,33 hingga 3,48% pada accuracy, 2,43 hingga 2,56% pada *precision*, 3,71 hingga 4,07% pada *recall*, dan 3,47 hingga 3,75% pada *f1-score*. Dalam pendekatan *multi class* OVO, kenaikan nilai evaluasi *training* dan *testing* masing-masing 1,08% dan 3,33% pada *accuracy*, 0,44% dan 2,56% pada *precision*, 1,35% dan 3,71% pada *recall*, serta 1,19% dan 3,47% pada *f1-score* (skenario kedua terhadap skenario pertama). Dalam pendekatan *multi class* OVA, kenaikan nilai evaluasi *training* dan *testing* masing-masing 1,96% dan 3,48% pada *accuracy*, 1,21% dan 2,43% pada *precision*, 2,51% dan 4,07% pada *recall*, serta 2,29% dan 3,75% pada *f1-score* (skenario keempat terhadap skenario ketiga).

Pendekatan *multi class* OVA memiliki beberapa nilai evaluasi *training* dan *testing* lebih baik dibanding pendekatan *multi class* OVO hanya jika menggunakan PSO. Evaluasi *training* terjadi kenaikan 0,68% pada *accuracy*, 0,56% pada *precision*, 0,92% pada *recall*, dan 0,86% pada *f1-score* serta evaluasi *testing* terjadi kenaikan 0,15% pada *accuracy*, 0,19% pada *recall*, dan 0,11% pada *f1-score* dimana *precision* menurun 0,11% (skenario keempat terhadap skenario kedua). Sebaliknya, pendekatan *multi class* OVO memiliki beberapa nilai evaluasi *training* dan *testing* lebih baik dibanding pendekatan *multi class* OVA hanya jika tidak menggunakan PSO. Evaluasi *training* terjadi kenaikan 0,2% pada *accuracy*, 0,21% pada *precision*, 0,24% pada *recall*, dan 0,24% pada *f1-score* serta evaluasi *testing* terjadi kenaikan 0,02% pada *recall* dan 0,17% pada *f1-score* dimana *precision* menurun 0,02% dan *accuracy* bernilai sama (skenario pertama terhadap skenario ketiga).

Semua nilai *gap* pada setiap tolak ukur model skenario memiliki tanda (+) menandakan keseluruhan skenario model memiliki hasil evaluasi *testing* lebih baik dibanding *training*. Skenario pertama memiliki *gap*, yaitu +2.84% pada *accuracy*, +3.42% pada *precision*, +3.10% pada *recall*, dan +3.31% pada *f1-score*. Skenario kedua memiliki *gap*, yaitu +5.09% pada *accuracy*, +5.54% pada *precision*, +5.46% pada *recall*, dan +5.59% pada *f1-score*. Skenario ketiga memiliki *gap*, yaitu +3.04% pada *accuracy*, +3.65% pada *precision*, +3.17% pada *recall*, dan +3.38% pada *f1-score*. Skenario keempat memiliki *gap*, yaitu +4.56% pada *accuracy*, +4.87% pada *precision*, +4.73% pada *recall*, dan +4.84% pada *f1-score*.

### Perbandingan Evaluasi *Training* dan *Testing*



**Gambar IV.12** Perbandingan training dan testing tiap skenario

Perbandingan evaluasi *training* dan *testing* tiap skenario dilakukan dengan acuan akurasi (*accuracy*). Akurasi *training* adalah *accuracy* rata-rata oleh 10-*fold* CV dan akurasi *testing* adalah nilai *accuracy* oleh *confusion matrix* (lihat Gambar IV.12 diatas). Berdasarkan hasil perbandingan tersebut, terdapat beberapa poin yang diperoleh antara lain:

PSO dapat meningkatkan nilai akurasi *training* sebesar 1,06 hingga 1,96%, serta akurasi *testing* sebesar 3,33 hingga 3,48%. Ini dikarenakan PSO sebagai *tuning hyperparameter* melakukan optimasi terhadap *hyperparameter* C dan *gamma* agar menemukan *hyperlane* pada SVM yang lebih dinamis. Semakin dinamis *hyperlane* yang terbentuk, maka semakin baik juga model dalam mengklasifikasikan data. Pada pendekatan *multi class* OVO, kenaikan nilai akurasi *training* dan *testing* adalah 1,08% dan 3,33% (skenario kedua terhadap skenario pertama). Sedangkan, pada pendekatan *multi class* OVA, kenaikan nilai akurasi *training* dan *testing* adalah 1,96% dan 3,48% (skenario keempat terhadap skenario ketiga).

Pendekatan *multi class* OVA memiliki nilai akurasi *training* dan *testing* lebih tinggi dibandingpendekatan *multi class* OVO hanya jika menggunakan PSO, yaitu 0,68% dan 0,15% (skenario keempat terhadap skenario kedua). Sebaliknya, pendekatan *multi class* OVO memiliki nilai akurasi *training* lebih tinggi dibandingpendekatan *multi class* OVA hanya jika tidak menggunakan PSO, yaitu 0,2% dimana keduanya memiliki nilai akurasi *testing* yang sama (skenario pertama terhadap skenario ketiga).

Setiap model skenario yang dibangun tidak mengalami *overfitting* karena akurasi *testing* lebih baik dibanding *training*. Skenario pertama mengalami kenaikan sebesar 2,84%, skenario kedua mengalami kenaikan sebesar 5,09%, skenario ketiga mengalami kenaikan sebesar 3,04%, dan skenario keempat mengalami kenaikan sebesar 4,56%.

Skenario keempat merupakan model dengan performa terbaik dengan nilai akurasi *testing* 83,18% diikuti akurasi *training* 78,62%. Sedangkan, model dengan performa terendah adalah skenario ketiga dengan akurasi *testing* 79,70% diikuti akurasi *training* 76,66%.

### Aplikasi Generik



**Gambar IV.13** Aplikasi generik

Pada Gambar IV.13, ditampilkan aplikasi generik yang dibangun berdasarkan model skenario keempat, yaitu model klasifikasi *multi class* SVM pendekatan OVA. *Hyperparameter* *gamma* bernilai 0.24799216 dan *hyper-parameter C* bernilai 6.98501535 yang diperoleh dari hasil *tuning hyperparameter* PSO sebelumnya. Aplikasi akan memberikan hasil klasifikasi berupa gambar yang mewakilkan satu kelas sentimen, yaitu negatif, netral, ataukah positif setelah *user* memberikan input berupa teks dan menekan tombol “proses”. Aplikasi tersebut dapat diakses dengan link https://klasifikasi-sentimen-cl.herokuapp.com.

**Tabel IV.5** Hasil Klasifikasi Teks Aplikasi Generik

|  |  |  |
| --- | --- | --- |
| No. | Contoh Teks | Hasil Klasifikasi |
| 1 | @94afi Program KIP ini sungguh membantu. Saya sebagai guru juga banyak menyaksikan siswa yang terbantu dengan program ini | Positif |
| 2 | w dah cape ya ajg sekolah daring. nugas mulu, paham kagak https://t.co/LFyf0sYXET | Negatif |
| 3 | b' mau nanya kalian yang daftar simak pake kip pas masukin kartu kip nya langsung begini atau dicrop jadi yang kotak aja? makasih https://t.co/g8PJ6UQ1gW' | Netral |
| 4 | b'kalian yg nomor tri nya udh ada paketan trs juga dapet bantuan pemerintah ntar kalo mau pake kuota belajar pemerintah kek gini gmn caranya ya? kan kecampur gini yak? takutnya tetep sedot kuota paketan bkn kuota belajar \xf0\x9f\x98\xad https://t.co/wsBqr9l5xd' | Negatif |
| 5 | @collegemenfess Coba ajukan kipk? Pengurangan ukt? Sayang aja sih udah semester 2 klo keluar ngulang lagi, tapi balik lagi ke diri sendiri sama coba kompromi sama ortu dulu^^ | Positif |
| 6 | @SoundOfYogi saat pendapatan berkurang karena pandemi, sekolah daring meringankan beban ekonomi, karena memangkas biaya transportasi yg cukup besar. Effort utk antar-jemput anak sekolah itu banyak, apalagi yg punya 4-5 anak usia sekolah | Netral |
| 7 | b'@eLAmaravati Enak asli kuliah daring, pulang kerja tinggal buka laptop gausah ngejar waktu ke kampus wkwk' | Negatif |
| 8 | Mahasiswa suka dengan program kuota belajar kemendikbud | Positif |
| 9 | Mahasiswa tidak suka dengan program kuota belajar kemendikbud | Negatif |
| 10 | Mahasiswa suka dengan program kuota belajar, tapi tidak suka kuliah daring dari kemendikbud | Positif |
|  |  |  |

Pada Tabel IV.5, disajikan pengujian teks dan hasil klasifikasi oleh aplikasi generik. Pada nomor satu, diklasifikasikan positif karena mendukung program KIP pemerintah. Pada nomor dua, diklasifikasikan negatif karena tidak suka terhadap adanya belajar daring. Pada teks nomor tiga, diklasifikasikan netral karena bertanya seputar program KIP. Pada nomor empat, diklasifikasikan negatif yang mana pada ekspektasinya teks lebih cenderung bersifat netral karena bertanya seputar kuota belajar. Pada nomor lima, diklasifikasikan positif yang mana ekspektasinya teks lebih cenderung bersifat netral karena bertanya seputar bantuan UKT. Pada nomor enam, diklasifikasikan netral yang mana ekspektasinya teks lebih cenderung bersifat positif karena membahas dampak positif adanya sekolah daring. Pada nomor tujuh, diklasifikasikan negatif yang mana ekspektasinya teks lebih cenderung bersifat positif karena kuliah daring membantu mahasiswa. Pada nomor delapan, diklasifikasikan positif karena mahasiwa memberi respon positif terhadap program kuota belajar Kemendikbud. Pada nomor sembilan, diklasifikasikan negatif karena mahasiswa memberi respon negatif terhadap program kuota belajar Kemendikbud. Pada nomor sepuluh, diklasifikasikan positif yang mana ekspektasinya teks lebih cenderung bersifat netral karena adanya kata “suka” terhadap program kuota belajar dan “tidak suka” terkait kuliah daring dalam satu teks tersebut. Ini dapat terjadi karena kata “suka” memiliki bobot sentimen lebih tinggi dibanding kata “tidak suka” sehingga aplikasi mengklasifikasikan teks tersebut ke dalam kelas positif.

# PENUTUP

## Kesimpulan

Berdasarkan penelitian yang telah dilakukan sebelumnya, maka dapat disimpulkan sebagai berikut:

1. Model klasifikasi *multi class* SVM pendekatan OVA yang menggunakan PSO memberikan performa terbaik dengan nilai akurasi *training* dan *testing* sebesar 78,62% dan 83,18%. Sedangkan, model klasifikasi *multi class* SVM pendekatan OVA yang tidak menggunakan PSO memberikan performa terendah dengan nilai akurasi *training* dan *testing* sebesar76,66% dan 79,70%.
2. Penggunaan *tuning hyperparameter* PSO dapat meningkatkan nilai akurasi *training* dan *testing* terhadap pendekatan *multi class*, yaitu OVO sebesar 1,08% dan 3,33%, serta OVA sebesar 1,96% dan 3,48%.
3. Pendekatan *multi class* (OVA) memiliki akurasi *training* dan *testing* lebih tinggi dibandingpendekatan *multi class* (OVO) hanya jika menggunakan PSO, yaitu 0,68% dan 0,15%. Sedangkan, pendekatan *multi class* (OVO) memiliki nilai akurasi *training* lebih tinggi dibandingpendekatan *multi class* (OVA) hanya jika tidak menggunakan PSO, yaitu 0,2% dan keduanya memiliki nilai akurasi *testing* yang sama.

## Saran

Hasil penelitian menunjukkan model klasifikasi *multi class* SVM pendekatan OVA dengan *tuning hyperparameter* PSO memiliki performa terbaik dengan akurasi *training* dan *testing* sebesar 78,62% dan 83,18%. Kemudian, setelah menerapkan model klasifikasi tersebut terhadap aplikasi generik yang telah dibangun, masih ada hasil klasifikasi dari teks yang diinputkan yang tidak sesuai ekspektasi oleh *user*. Oleh karena itu, perlu pengembangan lebih lanjut agar dapat meningkatkan performa tersebut, seperti menambah jumlah dataset, perbandingan dengan metode klasifikasi lainnya (*Naive Bayes*, KNN, *Desicion Tree*, dan sejenisnya), penambahan *feature selection*, dan perbandingan terhadap *tuning hyperparameter* lainnya (*grid search*, CSO, dan sejenisnya).

**DAFTAR PUSTAKA**

Aaputra, S. A. (2019). Sentiment Analysis Analisis Sentimen E-Wallet Pada Google Play Menggunakan Algoritma Naive Bayes Berbasis Particle Swarm Optimization. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *3*(3), 377–382.

Alhaq, Z., Mustopa, A., Mulyatun, S., & Santoso, J. D. (2021). PENERAPAN METODE SUPPORT VECTOR MACHINE UNTUK ANALISIS Informatika Universitas Amikom Yogyakarta Abstraksi Keywords : Pendahuluan Tinjauan Pustaka. *Jurnal of Information System Management*, *Vol. 3*(2), 16–21. https://jurnal.amikom.ac.id/index.php/joism/article/view/558

Alita, D., Fernando, Y., & Sulistiani, H. (2020). Implementasi Algoritma Multiclass Svm Pada Opini Publik Berbahasa Indonesia Di Twitter. *Jurnal Tekno Kompak*, *14*(2), 86–91. https://doi.org/10.33365/jtk.v14i2.792

Ayunin, P. Q. (2019). *Rekomendasi Anotasi Otomatis Pada Konten Pembelajaran pada Konten Pembelajaran MOOC*. https://repository.its.ac.id/60444/

Buntoro, G. A. (2017). Analisis Sentimen Calon Gubernur DKI Jakarta 2017 Di Twitter. *INTEGER: Journal of Information Technology*, *1*(1), 32–41. https://www.researchgate.net/profile/Ghulam\_Buntoro/publication/316617194\_Analisis\_Sentimen\_Calon\_Gubernur\_DKI\_Jakarta\_2017\_Di\_Twitter/links/5907eee44585152d2e9ff992/Analisis-Sentimen-Calon-Gubernur-DKI-Jakarta-2017-Di-Twitter.pdf

Dietterich, T. (n.d.). *Overfitting and Undercomputing in Machine Learning*. 2–3.

Ghudafa, M., Akbar, T., & Srisulistiowati, D. B. (2021). Analisa Sentimen Efektifitas Vaksin terhadap Varian COVID 19 Omicron Berbasis Leksikon. *Journal of Information and Information Security (JIFORTY*, *2*(2), 251–258. http://ejurnal.ubharajaya.ac.id/index.php/jiforty

Google. (2019). *Imbalanced Data | Data Preparation and Feature Engineering for Machine Learning | Google Developers*. https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data

Hadna, N. M. S., Santosa, P. I., & Winarno, W. W. (2016). Studi Literatur Tentang Perbandingan Metode Untuk Proses Analisis Sentimen Di Twitter. *Seminar Nasional Teknologi Informasi Dan Komunikasi (SENTIKA)*, 57–64. https://fti.uajy.ac.id/sentika/publikasi/makalah/2016/95.pdf

Hakim, I., Nugroho, A., Sukmana, S. H., & Gata, W. (2020). Sentimen Analisis Stay Home menggunakan metode klasifikasi Naive Bayes , Support Vector Machine , dan k-Nearest Neighbor. *Paradigma - Jurnal Informatika Dan Komputer*, *22*(2), 169–174.

Hayuningtyas, R. Y., & Sari, R. (2019). Analisis Sentimen Opini Publik Bahasa Indonesia Terhadap Wisata Tmii Menggunakan Naïve Bayes Dan Pso. *Jurnal Techno Nusa Mandiri*, *16*(1), 37–42. https://doi.org/10.33480/techno.v16i1.115

Hernandez-Suarez, A., Sanchez-Perez, G., Toscano-Medina, K., Martinez-Hernandez, V., Sanchez, V., & Perez-Meana, H. (2018). *A Web Scraping Methodology for Bypassing Twitter API Restrictions*. 1–7. http://arxiv.org/abs/1803.09875

Husada, H. C., & Paramita, A. S. (2021). Analisis Sentimen Pada Maskapai Penerbangan di Platform Twitter Menggunakan Algoritma Support Vector Machine (SVM). *Teknika*, *10*(1), 18–26. https://doi.org/10.34148/teknika.v10i1.311

Istikomah, L., Cholissodin, I., & Marji. (2017). Implementasi Algoritma Particle Swarm Optimization (PSO) untuk Optimasi Pemenuhan Kebutuhan Gizi Balita. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, *1*(11), 1321–1330.

Luqyana, W. A., Cholissodin, I., & Perdana, R. S. (2018). Analisis Sentimen Cyberbullying Pada Komentar Instagram dengan Metode Klasifikasi Support Vector Machine. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, *2*(11), 4704–4713.

Muhidin, A., & Burhan, A. (2018). Klasifikasi Tingkat Produksi Tanaman Padi Kabupaten Karawang Menggunakan Metode Naive Bayes dan K-Fold Cross Validation. *Jurnal Teknologi Pelita Bangsa- SIGMA*, *8*(2), 225–232.

Nasution, M. R. A., & Hayaty, M. (2019). Perbandingan Akurasi dan Waktu Proses Algoritma K-NN dan SVM dalam Analisis Sentimen Twitter. *Jurnal Informatika*, *6*(2), 226–235. https://doi.org/10.31311/ji.v6i2.5129

Negara, A. B. P., Muhardi, H., & Sajid, F. (2021). Perbandingan Algoritma Klasifikasi terhadap Emosi Tweet Berbahasa Indonesia. *Jurnal Edukasi Dan Penelitian Informatika (JEPIN)*, *7*(2), 242–249. https://jurnal.untan.ac.id/index.php/jepin/article/view/48198

Ningrum, H. C. S. (2018). Perbandingan Metode Support Vector Machine (SVM) Linear, Radial Basis Function (RBF), dan Polinomial Kernel dalam Klasifikasi Bidang Studi Lanjut Pilihan Alumni UII. In *Tugas Akhir Statistika Universitas Islam Indonesia*.

Okwu, M. O., & Tartibu, L. K. (2021). Particle Swarm Optimisation. *Studies in Computational Intelligence*, *927*, 5–13. https://doi.org/10.1007/978-3-030-61111-8\_2

Pawluszek-Filipiak, K., & Borkowski, A. (2020). On the importance of train-test split ratio of datasets in automatic landslide detection by supervised classification. *Remote Sensing*, *12*(18). https://doi.org/10.3390/rs12183054

Peryanto, A., Yudhana, A., & Umar, R. (2020). Klasifikasi Citra Menggunakan Convolutional Neural Network dan K Fold Cross Validation. *Journal of Applied Informatics and Computing (JAIC)*, *4*(1), 45–51. https://doi.org/10.30871/jaic.v4i1.2017

Piotrowski, A. P., Napiorkowski, J. J., & Piotrowska, A. E. (2020). Population size in Particle Swarm Optimization. *Swarm and Evolutionary Computation*, *58*(May), 100718. https://doi.org/10.1016/j.swevo.2020.100718

Prangga, S. (2017). *Optimasi Parameter Pada Support Vector Machine Menggunakan Pendekatan Metode Taguchi Untuk Data High-Dimensional Parameter Optimization of Support Vector Machine Using Taguchi Approach for High- Dimensional Data*.

Putra, A. P., Debataraja, N. N., & Kusnandar, D. (2020). Tingkat Akurasi Klasifikasi Jarak Kelahiran Di Kampung Keluarga Berencana (KB) Dengan Metode Support Vector Machine (SVM). *Buletin Ilmiah Mat. Stat. Dan Terapannya (Bimaster)*, *9*(3), 361–370.

Putra Nuansa, E. (2017). Analisis Sentimen Pengguna Twitter Terhadap Pemilihan Gubernur Dki Jakarta Dengan Metode Naïve Bayesian Classification Dan Support Vector Machine. In *Institut Teknologi Sepuluh Nopember Surabaya*.

Rofiqoh, U., Perdana, R. S., & Fauzi, M. A. (2017). Analisis Sentimen Tingkat Kepuasan Pengguna Penyedia Layanan Telekomunikasi Seluler Indonesia Pada Twitter Dengan Metode Support Vector Machine dan Lexion Based Features. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, *1*(12), 1725–1732. http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/628

Salekhah, C. I. (2016). *Implementasi Metode Multi Class Support Vector Machine Untuk Klasifikasi Emosi Pada Lirik Lagu Bahasa Indonesia*.

Sigala, M., Beer, A., Hodgson, L., & O’Connor, A. (2019). *Big Data for Measuring the Impact of Tourism Economic Development Programmes: A Process and Quality Criteria Framework for Using Big Data*.

Sumitro, P. A., Rasiban, Mulyana, D. I., & Saputro, W. (2021). Analisis Sentimen Terhadap Vaksin Covid-19 di Indonesia pada Twitter Menggunakan Metode Lexicon Based. *J-ICOM - Jurnal Informatika Dan Teknologi Komputer*, *2*(2), 50–56. https://doi.org/10.33059/j-icom.v2i2.4009

Syafar, D. N. (2016). Negasi Dalam Bahasa Indonesia Dan Bahasa Inggris. *Jurnal Arbitrer*, *3*(1), 1–11. https://doi.org/10.25077/ar.3.1.1-11.2016

Tane, O. Z. A., Lhaksmana, K. M., & Nhita, F. (2019). Analisis Sentimen pada Twitter Tentang Calon Presiden 2019 Menggunakan Metode SVM (Support Vector Machine). *E-Proceeding of Engineering*, *6*(2), 9716–9725.

Teknologi, D. A. N., Sultan, U., Tirtayasa, A., & Pascasarjana, F. (2021). *Kementerian pendidikan, kebudayaan riset, dan teknologi*.

Tempola, F., Muhammad, M., & Khairan, A. (2018). Perbandingan Klasifikasi Antara KNN dan Naive Bayes pada Penentuan Status Gunung Berapi dengan K-Fold Cross Validation. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(5), 577–584. https://doi.org/10.25126/jtiik.201855983

Thabtah, F., Hammoud, S., Kamalov, F., & Gonsalves, A. H. (2020). Data imbalance in classification: Experimental evaluation. *Information Sciences*, *513*, 429–441. https://doi.org/10.1016/j.ins.2019.11.004

Tineges, R., Triayudi, A., & Sholihati, I. D. (2020). Analisis Sentimen Terhadap Layanan Indihome Berdasarkan Twitter Dengan Metode Klasifikasi Support Vector Machine (SVM). *JURNAL MEDIA INFORMATIKA BUDIDARMA*, *4*(3), 650–658. https://doi.org/10.30865/mib.v4i3.2181

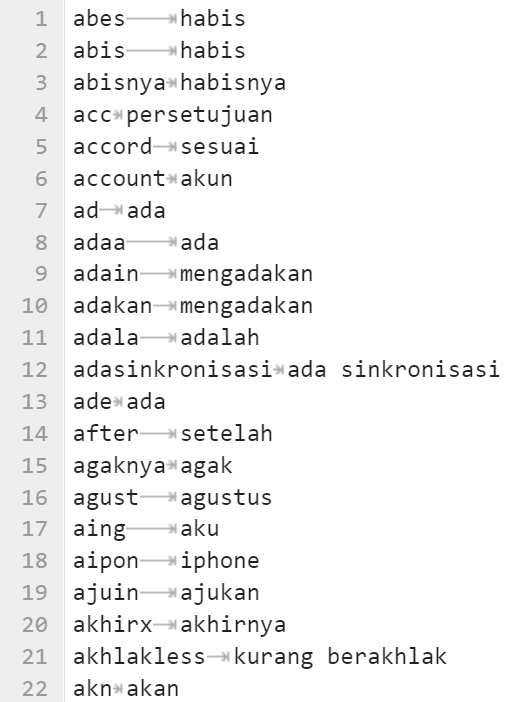
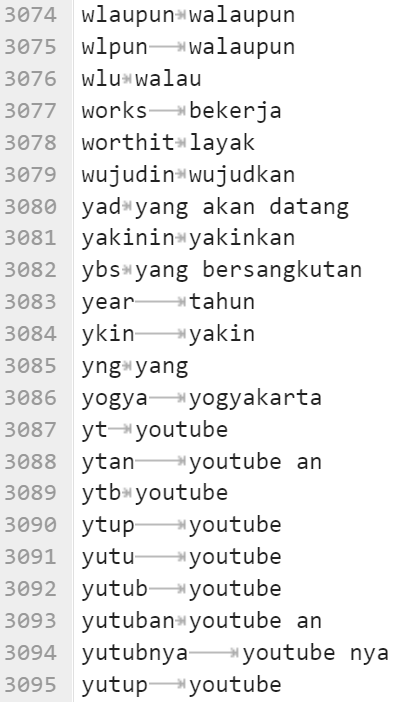
Ulfa, M., Haryanto, H., & Wibisono, K. A. (2019). Desain Sistem Pengenalan Dan Klasifikasi Kopi Bubuk Bermerek Dengan Menggunakan Electronic Nose Berbasis Artifical Neural Network (ANN). *Jurnal Elektronika, Listrik, Telekomunikasi, Komputer, Informatika, Sistem Kontrol (J-Eltrik)*, *1*(2), 51–60. https://doi.org/10.30649/j-eltrik.v1i2.15

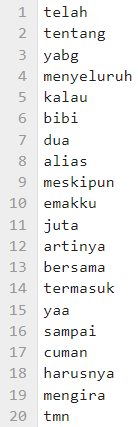
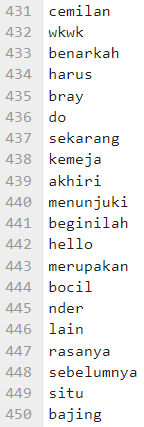
Viva.co.id. (2020). *Survei Pendidikan Dunia, Indonesia Peringkat 72 dari 77 Negara*. Kamis, 5 Desember 2019.

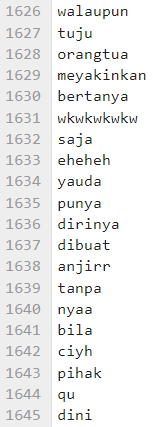
Ying, X. (2019). An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series*, *1168*(2). https://doi.org/10.1088/1742-6596/1168/2/022022

**LAMPIRAN A  
KAMUS NORMALISASI DAN STOPLIST**

1. **Kamus Normalisasi (3095 Pasangan Kata)**

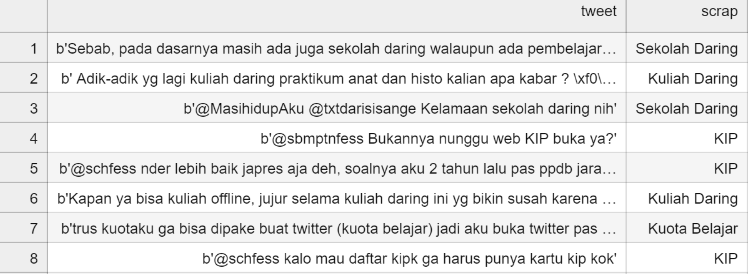
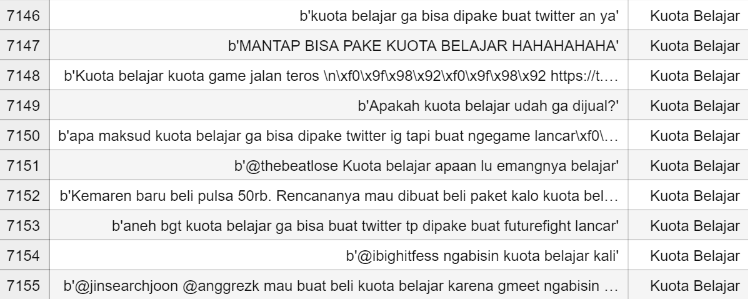
****

1. **Stoplist (1645 Kata)**

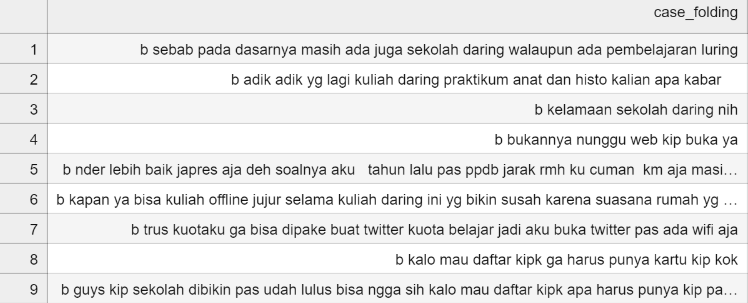
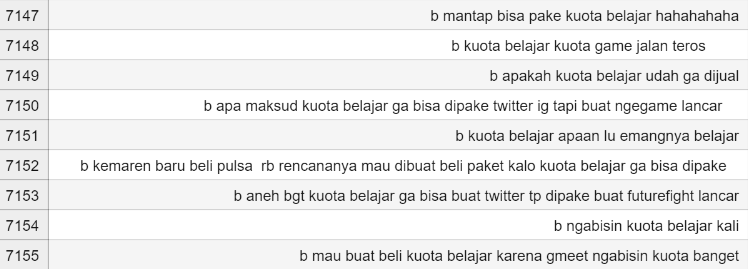


**LAMPIRAN B  
DATASET PENELITIAN**

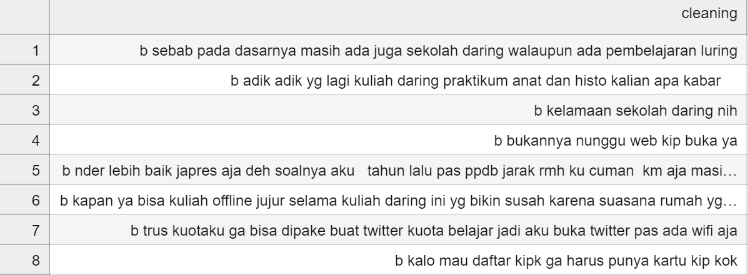
1. **Hasil *Scraping***



1. **Hasil *Text Preprocessing* (*Case Folding*)**

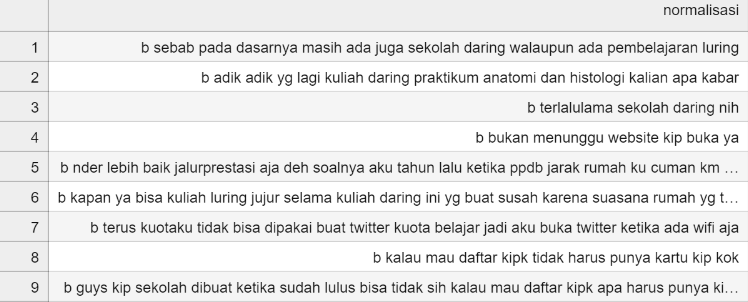


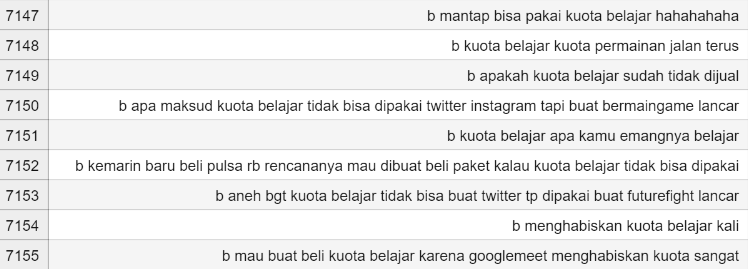
1. **Hasil *Text Preprocessing* (*Cleaning*)**

****

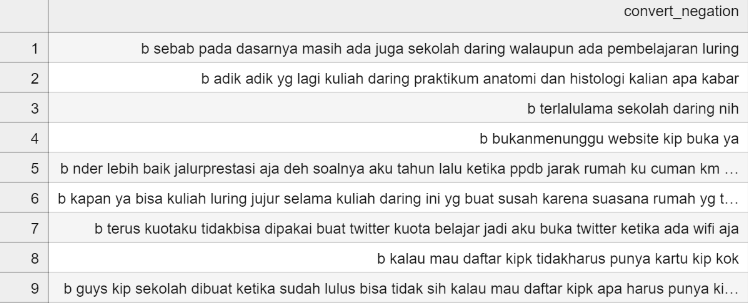


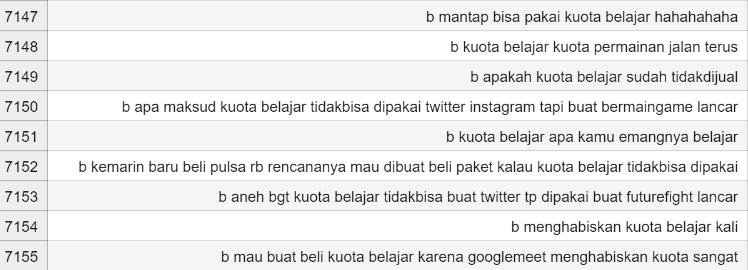
1. **Hasil *Text Preprocessing* (*Normalisasi*)**





1. **Hasil *Text Preprocessing* (*Convert Negation*)**





1. **Hasil *Text Preprocessing* (Tokenisasi)**





1. **Hasil *Text Preprocessing* (*Stopword Removal*)**



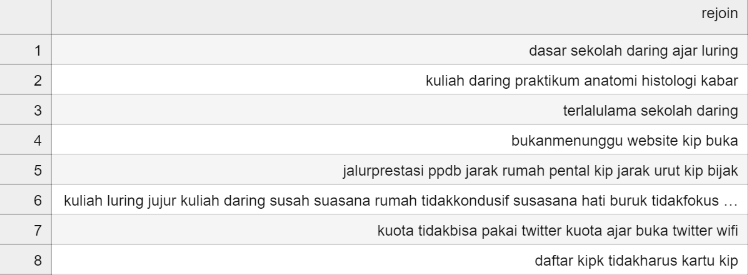


1. **Hasil *Text Preprocessing* (*Stemming*)**



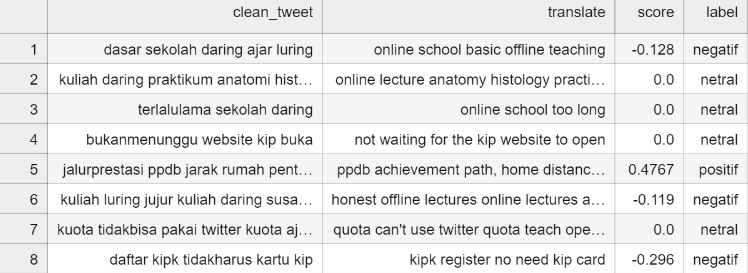


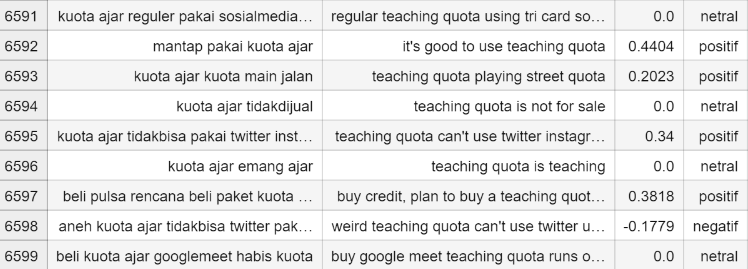
1. **Hasil *Text Preprocessing* (*Rejoin*)**



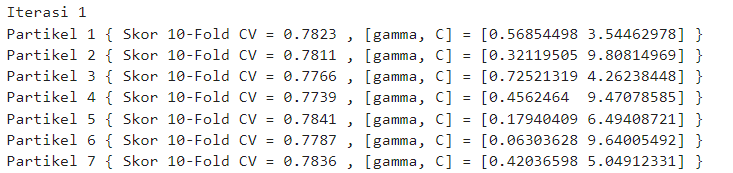
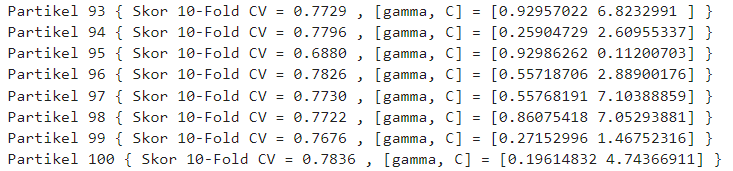


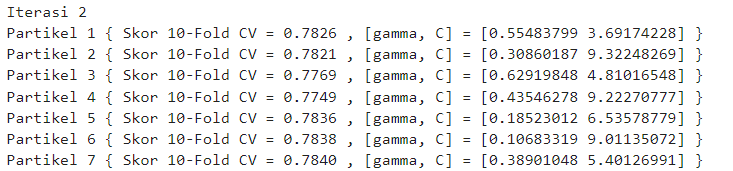
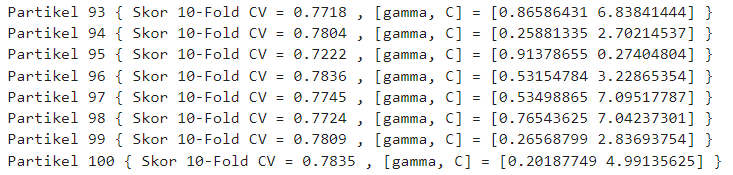
1. **Hasil *Remove Duplicate* dan Pelabelan Data**

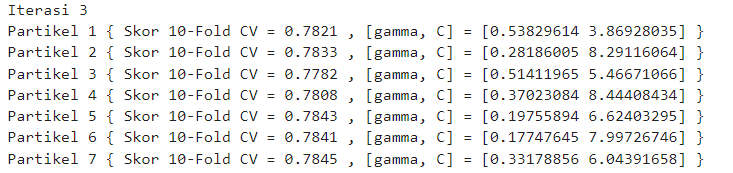
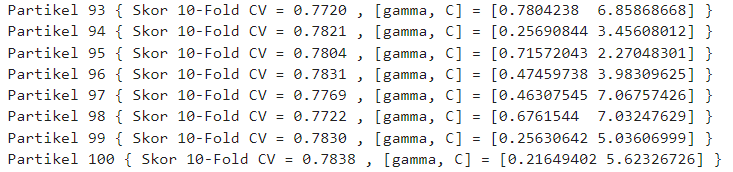




**LAMPIRAN C  
FITNESS CANDIDATE DAN GBEST (OVA)**

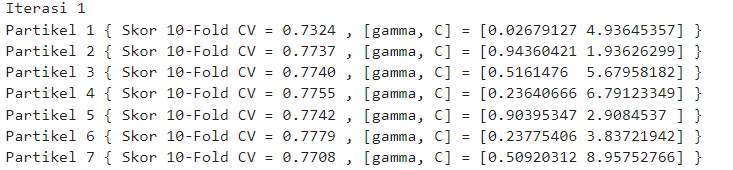
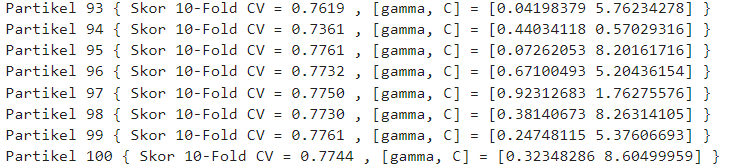


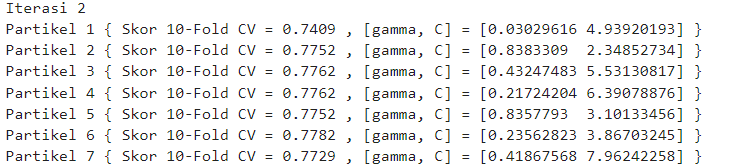
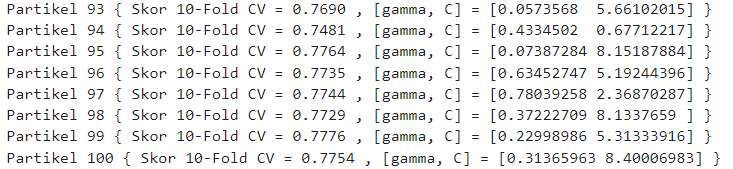
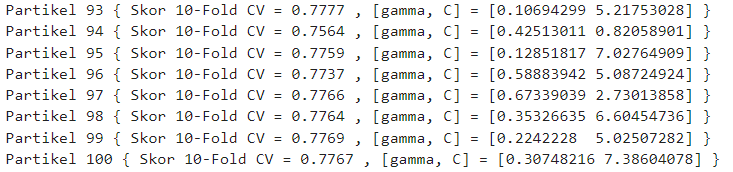


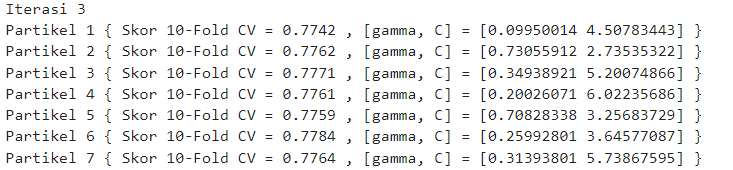




**LAMPIRAN D  
FITNESS CANDIDATE DAN GBEST (OVO)**









**LAMPIRAN E  
CLASSIFICATION REPORT**

