PENDETEKSIAN HATE SPEECH PADA SOSIAL MEDIA INDONESIA DENGAN ALGORITMA SUPPORT VECTOR MACHINE (SVM) DAN DECISION TREE

Febrian Andy Kusuma; Endang Wahyu Pamungkas

Prodi Teknik Informatika, Fakultas Komunikasi dan Informasi, Universitas Muhammadiyah Surakarta

Abstrak

Ujaran kebencian (*Hate Speech*) adalah setiap ucapan, gerak tubuh, perilaku, tulisan, atau tampilan yang dapat memicu kekerasan atau tindakan merugikan karena meremehkan dan mengintimidasi individu atau kelompok tertentu dapat mendorong atau memicu kebencian. Ujaran kebencian dapat berupa kata-kata, gambar, dan bentuk ekspresi lainnya. *Hate Speech* menyebabkan sebuah kebencian terhadap suatu kelompok maupun individu tertentu *Hate Speech* biasa dilakukan oleh oknum yang memiliki ketidaksukaan terhadap suatu kelompok yang bertujuan menghasut orang lain untuk ikut membenci kelompok tersebut. Kolom komentar pada *twitter* perlu adanya pembatasan dan kesadaran dari masyarakat tentang berbahayanya sebuah ujaran kebencian yang dapat menyebabkan konflik yang berkepanjangan. Penelitian ini bertujuan untuk mengetahui hasil akurasi pendeteksian bahasa kasar/kotor dari penerapan algoritma *Support Vector Machine* (SVM) dan *Decision Tree*. Metode tersebut diterapkan pada dataset twitter berbahasa Indonesia dengan banyak 13.169 baris data. Penelitian ini menghasilkan pendeteksian *Hate Speech* pada *tweet* berbahasa Indonesia dengan performa teringgi pada model SVM dengan hasil *Precission* 84%, *Recall 89%*, *Accuracy 83%* dan *F1-Score 86%*

Kata Kunci: decision tree, support vector machine, pendeteksian hate speech.

Abstract

Hate Speech is any speech, gesture, behavior, writing, or display that can trigger violence or harmful actions because belittling and intimidating certain individuals or groups can encourage or trigger hatred. Hate speech can be in the form of words, pictures, and other forms of expression. Hate Speech causes hatred for a particular group or individual. Hate Speech is usually carried out by individuals who have a dislike for a group with the aim of inciting others to hate the group. The comment column on Twitter needs restrictions and awareness from the public about the dangers of

hate speech which can lead to prolonged conflict. This study aims to determine the accuracy of the detection of foul language from the application of the Support Vector Machine (SVM) and Decision Tree algorithms. This method is applied to the Indonesian Twitter dataset with a total of 13,169 data lines. This research resulted in the detection of Hate Speech in Indonesian language tweets with the highest performance in the SVM model with a precision of 84%, Recall of 89%, Accuracy of 83% and F1-Score of 86%.

Keywords: decision tree, support vector machine, detection hate speech

1. PENDAHULUAN

Perkembangan teknologi sekarang berbeda dengan 10 tahun kebelakang di zaman serba digital saat ini keterbukaan informasi sangatlah mudah di dapatkan oleh setiap orang. Karena bisa dipastikan bahwa setiap orang memiliki ponsel pintar. Setiap orang terlibat secara online baik diforum website maupun jejaring sosial seperti *Instagram*, *Twitter*, *Facebook* dan jejaring sosial lainya. Selalu ada resiko dari setiap penggunanya yaitu ujaran kebencian. Ujaran kebencian adalah pernyataan atau tindakan yang menyebarluaskan pandangan atau perasaan negatif terhadap suatu kelompok orang atau individu karena faktor-faktor seperti ras, agama, jenis kelamin, orientasi seksual, atau identitas lainnya. Ujaran kebencian dapat berupa kata-kata yang menyinggung, tindakan fisik, atau tindakan online yang menyebarluaskan pandangan *rasisme, antisemit, homofobik*, atau diskriminatif lainnya. Dampak dari ujaran kebencian atau kata-kata kasar dapat sangat merugikan bagi pihak yang ditujukan. Dapat menyebabkan rasa tidak aman, cemas, atau tidak nyaman bagi individu atau kelompok yang ditujukan. Dapat juga menyebabkan diskriminasi atau perlakuan tidak adil terhadap individu atau kelompok tersebut. Ujaran kebencian dapat juga menyebabkan kerusakan pada hubungan sosial dan meningkatkan ketegangan antar kelompok. Dalam jangka panjang, ujaran kebencian dapat menyebabkan kerusakan pada kesehatan mental dan fisik bagi individu yang terkena dampaknya.

Indonesia merupakan salah satu negara dengan pengguna jejaring sosial media terbanyak jejaring sosial sering digunakan untuk berbagai tujuan seperti berbagi informasi, menjalin komunikasi dengan teman,media periklanan atau sekedar hiburan semata. Kata-

kata kasar (*abusive language*) adalah ekspresi berisi kata mengandung kebencian dan frasa kasar atau kotor. Baik lisan maupun tulisan penyebabnya adalah kurangnya alat yang efektif untuk menyaring bahasa kasar di jejaring sosial. Kurangnya empati dari penguna jejaring sosial tersebut yang tidak mematuhi aturan/kebijakan dari aplikasi tersebut. kata-kata kasar (*abusive language*) jejaring sosial harus di filter agar anak-anak dan remaja tidak tahu bahasa kasar dari jejaring sosial yang mereka gunakan.

UU ITE (Undang-Undang Informasi dan Transaksi Elektronik) di Indonesia memberikan dasar hukum untuk mengatasi masalah ujaran kebencian yang dilakukan melalui jaringan internet. UU ITE mengatur bahwa setiap orang yang melakukan tindakan yang dapat merugikan keamanan negara, persatuan, atau ketertiban umum, atau melakukan tindakan yang dapat merugikan hak asasi manusia, melakukan tindakan yang dapat merugikan lingkungan hidup, atau melakukan tindakan yang dapat merugikan kepentingan ekonomi negara, akan dikenakan sanksi pidana.

Adapun penelitian tentang deteksi bahasa kasar di jejaring sosial telah dilakukan dalam beberapa tahun terakhir penelitian yang dilakukan oleh Aditya Dwitama.(2021). Data yang digunakan adalah dataset *Twitter*. *Dataset* memiliki jumlah kelas sebanyak 12 dimana kelas kelas tersebut merupakan hasil diskusi antara peneliti dengan pihak Bareskrim Polri. Jumlah *tweet* yang terkandung dalam *dataset* adalah sebanyak 131.169 baris *tweet* yang sudah ter-anotasi ke dalam 12 marker didapat berhasil memberikan akurasi yang sangat baik dalam melakukan klasifikasi *multilabel* ujaran kebencian pada teks *twitter* dengan akurasi 98,07%.

Penelitian lainya dilakukan oleh Turaob dkk.(2017) topik pendeteksian hate speech dalam bahasa Thailand dataset yang digunakan adalah dataset postingan dan kolom komentar sosial media facebook dengan mengunakan berbagai algoritma seperti SVM,NB. Fitur yang digunakan dalam penelitian mereka adalah kata *n- gram* pendekatan penilaian dan fitur *sintaksis leksikal*

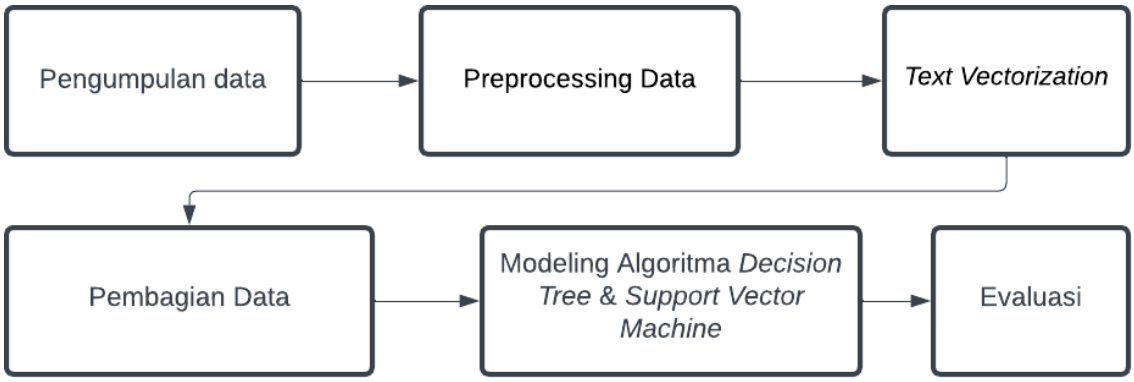
Twiter adalah sebuah platform media sosial yang didirikan oleh Jack Dorsey seorang mahasiswa sarajana di Universitas New York pada tahun 2006 Twitter memungkinkan penggunanya untuk berkomunikasi berupa text, video, foto. *Twitter* sendiri telah menjadi platform *media social* yang paling sering dikunjungi di internet lonjakan pengguna Twitter tentu saja berdampak pada ketentuan dari layanan tersebut salah satunya adalah masalah ujaran kebencian (*Hate Speech*) yang mengakibatkan

kesalahpaham antara pengguna *Twitter* yang dapat menimbulkan kebencian

Sistem diharapkan mampu untuk mengidentifikasi ujaran kebencian dalam sebuah kolom komentar jejaring sosial media. Berdasarkan latar belakang tersebut akan di implementasikan metode *Decicion Tree* dan *Support Vector Machine* untuk mengidentifikasi ujaran kebencian pada sosial media *twitter*. Adapun alasan pemilihan metode *Decision Tree* dan SVM dikarenakan *Decision Tree* sangat mudah dipahami dan diinterpretasikan, sehingga sangat cocok untuk analisis data yang memerlukan interpretasi yang mudah dipahami dan SVM adalah algoritma yang kuat untuk memecahkan masalah klasifikasi linear dan non-linear dengan akurasi yang tinggi. Selain itu, Decision Tree juga dapat digunakan untuk memecahkan masalah klasifikasi dan regresi dengan baik. Adapun tujuan dari penelitian tugas akhir ini adalah menghitung nilai akurasi dari algoritma *Decision Tree* dalam pengklasifikasian *Hate Speech* pada *tweet* berbahasa Indonesia dan menerapkan algoritma *Support Vector Machine* untuk mengklasifikasikan pada dataset *twitter* berbahasa indonesia

2. METODE

Tahapan yang dilakukan dalam penelitian untuk melakukan klasifikasi dengan mengunakan algoritma *Decicion Tree* dan *Support Vector Machine*



Gambar 1. Diagram alur penelitian

2.1. Pengumpulan Data

Data yang digunakan dalam penilitian ini adalah dataset dari Kaggle mengenai teks *Twitter* berbahasa indonesia. *Dataset* terdiri dari 13.169 baris data berbahasa Indonesia dengan 12 label data. Dataset memiliki label 1 dan 0 dengan data dengan

label 1 mengandung ujaran kebencian dan label 0 tidak mengandung ujaran kebencian. Label yang digunakan terdiri dari tiga, yaitu *Hate Speech*, *Abusive, dan Level Hate Speech*

2.2. Data Pre-Processing

Data preprocesing adalah serangkaian teknik dan langkah-langkah yang dilakukan pada data mentah sebelum diproses atau dianalisis. Tujuannya adalah untuk membersihkan, mengorganisir dan mempersiapkan data agar lebih mudah diproses dan dianalisis secara akurat. Teknik pemrosesan data meliputi eliminasi data yang tidak lengkap, penghapusan data yang tidak relevan, transformasi data, normalisasi data, penggabungan data, dan lain-lain. Data *Pre-Processing* sangat penting untuk memastikan kualitas data yang baik, menghindari bias, dan meningkatkan akurasi dan efisiensi analisis data.

2.2.1 Data *Case Folding*

Data *casefolding* proses mengubah semua huruf dalam teks menjadi huruf kecil atau besar yang seragam, sehingga memudahkan pengolahan teks dalam perangkat lunak yang membutuhkan standar tertentu untuk mengenali karakter. Contohnya, dalam proses *case folding*, teks "HeLLo" akan diubah menjadi "hello" atau "HELLO" tergantung pada kebutuhan aplikasi.

2.2.2 Data *cleansing*

Data *cleansing* adalah proses identifikasi, koreksi, dan penghapusan data yang tidak akurat, tidak lengkap, tidak relevan, atau duplikat dalam suatu set data. Tujuannya adalah untuk memastikan bahwa data yang digunakan dalam analisis atau pemrosesan lebih akurat, konsisten, dan dapat diandalkan. Teknik Data *Cleansing* meliputi identifikasi dan penghapusan data yang hilang atau tidak lengkap, koreksi data yang salah atau tidak akurat, penggabungan data dari sumber yang berbeda, dan eliminasi data duplikat. Data Cleansing sangat penting dalam pra-pemrosesan data untuk memastikan kualitas data yang baik dan meningkatkan akurasi analisis data

2.2.3 Data *Normalitation*

Data *Normalization* adalah proses mengorganisir data dalam suatu database relasional untuk meminimalkan redudansi duplikasi data dan mencegah ketidak konsistenan data. Tujuannya adalah untuk memastikan bahwa setiap entitas dalam database memiliki data yang konsisten dan memenuhi syarat integritas referensial. Teknik

normalisasi data meliputi pembuatan tabel dengan struktur yang diatur secara sistematis, pengaturan ketergantungan data, dan pembuatan kunci asing untuk menghubungkan data antar tabel. Normalisasi data penting untuk menghindari anomali data dan meningkatkan efisiensi database dalam pemrosesan, penyimpanan, dan pengambilan data. Normalisasi data biasanya dilakukan pada database relasional, tetapi juga dapat diterapkan pada sumber data lainnya seperti file CSV atau *spreadsheet*.

2.2.4 *Stopwords Removal*

Stopwords Removal adalah proses menghilangkan kata-kata umum yang tidak memiliki nilai makna tertentu dari sebuah teks, seperti kata depan, kata kerja bantu, dan konjungsi. Tujuannya adalah untuk meningkatkan akurasi dan efisiensi analisis teks dengan memfokuskan perhatian pada kata-kata yang lebih penting dan bermakna. Contoh kata penghubung yang sering dihapus adalah *"the"*, *"and"*, *"is"*, *"of"*, *"a"*, *"an"*, dan lain-lain.

2.3. *Text Vectorization*

Text Vectorization adalah proses mengubah teks menjadi bentuk numerik (vectors) agar dapat diolah oleh algoritma pemrosesan bahasa alami dan *machine learning*. Tujuan dari *Text Vectorization* adalah untuk merepresentasikan kata-kata dan dokumen dalam bentuk yang dapat dipahami oleh model pemrosesan bahasa alami dan machine learning. Teknik *text vectorization* meliputi *one-hot encoding, bag-of-words, term frequency-inverse document frequency* (TF-IDF) dan lain-lain.

2.4. Pembagian Data

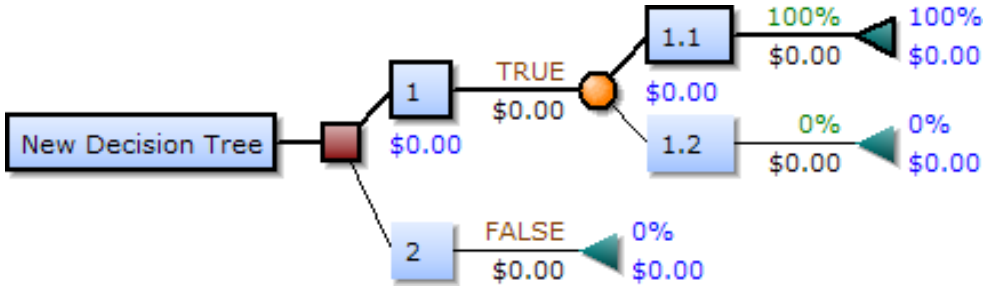
Pembagian data pada klasifikasi adalah proses membagi dataset menjadi subset data latihdan subset data pengujian untuk melatih dan menguji model klasifikasi. Pembagian data terdiri dari 80% *data set* dan 20% digunakan untuk data *testing* dari model yang sudah dibuat

2.5. Modeling

Modeling pada klasifikasi adalah proses membuat model klasifikasi yang dapat mempelajari pola dan hubungan antara fitur (features) dan label (kelas) dari dataset yang ada, sehingga dapat melakukan klasifikasi pada data yang belum dilihat sebelumnya.

2.6. Algoritma Decision Tree

Decision tree adalah sebuah algoritma yang digunakan dalam pembelajaran mesin untuk membuat keputusan. Algoritma ini menggunakan sebuah pohon biner yang terdiri dari kondisi (atau pertanyaan) dan keputusan (atau aksi) untuk membuat prediksi atau klasifikasi. Setiap kondisi dalam pohon biner ini akan membawa ke kondisi atau keputusan berikutnya sampai akhirnya mencapai sebuah keputusan akhir. Di dalam membuat *Decision Tree*, algoritma dapat menggunakan beberapa teknik seperti ID3, C4.5, dan CART untuk membuat dan mengevaluasi pohon keputusan. Algoritma ini juga dapat diterapkan dengan beberapa metode seperti *pruning* dan *boosting* untuk meningkatkan kinerja dari pohon keputusan yang dihasilkan. Berikut adalah elemen pada *Decision Tree*



Gambar 2. Element *Decicion Tree*

Proses *Decision Tree* dapat digunakan untuk mengubah data dalam format tabel dengan menggunakan struktur pohon untuk membuat fitur atau atribut baru dari data asli. Proses menggunakan pohon keputusan untuk transformasi data juga dikenal sebagai *tree-based feature engineering*. Untuk menghitung berasamaan *Entrophy* dapat dilihat pada persamaan (1)

$$Entropy(s) = \sum_{i=l}^{N} -Pi * Log_2\, pi \qquad (1)$$

Dengan keterangan: S : Himpunan kasus n : Jumlah partisi S pi : Proporsi dari Si terhadap S. Sedangkan untuk menghitung persamaan atribut *gain* dapat dilihat pada persamaan (2):

$$Gain(S, A) = Entrophy(S) - \sum_{i=1}^{n} \frac{|Si|}{|S|} * Entrophy\,(Si) \qquad (2)$$

Dengan:

S : Himpunan kasus A : Atribut

n : Jumlah partisi atribut A

|Si| : Jumlah kasus pada partisi ke i

|S| : Jumlah kasus dalam S

2.5.2 Algoritma Support Vector Machine

Support Vector Machine (SVM) adalah jenis algoritma pembelajaran mesin yang sering digunakan untuk masalah klasifikasi dan regresi. Ide utama di balik SVM adalah menemukan *hyperplane* yang memisahkan titik data ke dalam kelas yang berbeda sebaik mungkin. *Hyperplane* yang memaksimalkan margin, yaitu jarak antara titik data terdekat dari setiap kelas dan hyperplane. SVM dapat berupa pengklasifikasi *linier* atau *non-linier*. SVM linier menemukan hyperplane linier untuk memisahkan data, sedangkan SVM non-linear menggunakan teknik yang disebut trik kernel untuk memetakan data ke ruang dimensi yang lebih tinggi di mana hyperplane linier dapat ditemukan. Kernel umum yang digunakan dalam SVM termasuk kernel linier, kernel *polinomial*, dan *kernel radial basis function* (RBF).

2.5.3 Support Vector Machine Linier

Support Vector Machine (SVM) linear adalah varian dari algoritma SVM yang digunakan untuk klasifikasi data linier. Dalam klasifikasi linier SVM mencari garis atau *hyperplane* pembatas yang memisahkan dua kelas data dengan baik. Garis pembatas ini ditentukan dengan mencari vektor yang memiliki jarak terjauh dari kedua kelas data. Data yang berada di dekat garis pembatas ini disebut sebagai vektor dukungan (*support vectors*). SVM linear cocok digunakan untuk data yang memiliki klasifikasi yang jelas dan tidak terlalu kompleks. Namun SVM linear tidak dapat digunakan untuk data yang tidak linier. Jika data yang tidak linier maka SVM harus digunakan dengan kernel tertentu.

$$(w.\,xi + b) \geq +1 \text{ untuk } yi = +1 \; (w.\,xi + b) \leq -1 \text{ untuk } yi = -1$$

$$yi(w.\,xi + b) \geq 1,\; i = 1,2,3,\ldots,n$$

Kemudian dapat dijadikan *Quadratic Programming* dengan cara mencari titik minimal dari kedua fungsi. Untuk mendapatkan *hyperlane* dengan margin terbesar

adalah dengan mengubah persamaan (3)

$$Min = 1\ 2\ ||w||\ 2$$

(3)

Setelah itu syarat $yi(w.\ xi\ +\ b) \geq 1$ dipergunakan untuk mengatasi masalah pada optimasi yang dapat diatasi dengan fungsi Lagrange. Fungsi Lagrange yang digunakan yaitu fungsi Lagrange multiplier dengan Persamaan (4).

$$Min\ L(w, b, a) = 1\ 2\ ||w||\ 2 \sum ai\ [yi\ (w.\ xi + b) - 1]\ n\ i=1$$

(4)

AI sebagai nilai bobot untuk setiap data. Variable memiliki nilai nol atau positif ($ai \geq 0$) dengan meminimalkan nilai L terhadap w dan b, maka untuk melakukan perhitungan terhadap variabel w dan b dapat dilihat pada persamaan (5)

$$w = \sum aiyixi\ n\ i=1 \text{ dan } b = -\ 1\ 2\ (w.\ x+ + w.\ x-)$$

(5)

Dan x + dan x- masing-masing adalah nilai dari kelas positif dan kelas negatif sehingga menjadi sebuah fungsi untuk mencari *hyperplane* atau bidang pemisah terbaik dapat dilihat pada persamaan (6)

$$max \sum a1 - 1\ 2\ aiajyiyjxixj\ n\ i=1$$

(6)

Dengan $ai \geq 0(i = 1,2, \dots, n)$ dan $\sum aiyi = 0$

Hasil klasifikasi kemudian diperoleh menggunakan Persamaan (7)

$$f(x) = \sum a1yixi\ .\ x + b\ n\ i=1\ (2.7)\ f(x) = \sum (w.\ x + b)\ n\ i=1 \qquad (7)$$

Fungsi Klasifikasi = $sign(f(x))$

2.6. Evaluasi

Evaluasi model adalah proses pengukuran kinerja suatu model *machine learning* atau *statistik* dengan menggunakan *matrix* evaluasi untuk memeriksa sejauh mana model dapat memprediksi dengan akurat nilai target. Dalam mengevaluasi performa dari algoritma pada model bertujuan untuk menghitung akurasi dalam klasifikasi. Acuan *Confusion Matrix* digunakan untuk mempresentasikan prediksi dan kondisi sebenarnya (*actual).*

Dalam *confusion matrix* terdapat empat parameter yaitu *True Positive* (TP), *False Positive* (FP), *True Negative* (TN), dan *False Negative* (FN).

$$Precision = \frac{TP}{TP + FP} x\ 100$$

$$Recall = \frac{(TP)}{TP + FN} x\ 100$$

$$F1\ Score = \frac{2x(Recall\ x\ Precision)}{Recall + Precision} x\ 100$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} x\ 100$$

Keterangan:

TP = *True positive* (prediksi benar dari data positif)

TN = *True negative* prediksi benar dari data negatif)

FP = *False positive (*Prediksi salah dari data positif)

FN = *False negative* (Prediksi salah dari data negatif)

3. HASIL DAN PEMBAHASAN

3.1 Hasil

Penelitian ini mengunakan *dataset Twitter* berbahasa Indonesia sebanyak 13.169 baris data. *Dataset* terdiri dari 7068 berlabel 0 dan 5561 berlabel 1. *Dataset* tersebut harus dilakukan *Preprocessing* terlebih dahulu seperti mengubah huruf dalam bentuk kecil kemudian menghilangkan atau menghapus karakter- karakter yang tidak diperlukan seperti karakter baris baru, simbol yang berulang, *username*, URL, spasi dan karakter angka sebelum dilakukan prosess selanjutnya. Hasil dari *Preprocessing* dapat dilihat pada gambar 3

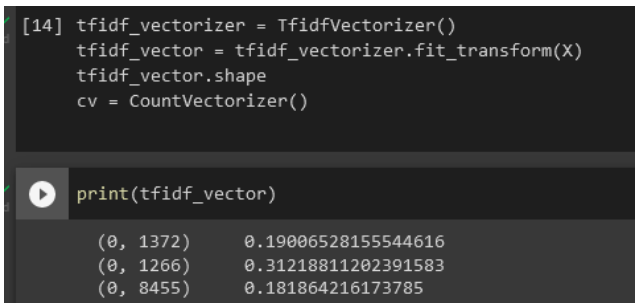
```
[ ] data_tweet = data_tweet[['Tweet', 'HS']]
    data_tweet['Tweet'] = data_tweet['Tweet'].apply(process)

[ ] data_tweet
```

	Tweet	HS
0	cowok berusaha melacak perhatian gue lantas re...	1
1	telat tau edan sarap gue bergaul cigax jifla c...	0
2	kadang berpikir percaya tuhan jatuh berkali ka...	0
3	ku tau matamu sipit	0
4	kaum cebong kafir dongoknya dungu haha	1
...	...	...
13164	berbicara ndasmu congor sekate anjing	1
13165	kasur enak kunyuk	0
13166	hati hati bisu bosan duh xf xf x xaa	0
13167	bom real mudah terdeteksi bom terkubur dahsyat...	0
13168	situ foto ya kutil onta	1

13169 rows × 2 columns

Gambar 3. Hasil *Preprocessing* data

Setelah data berhasil *dipreprocecing* maka proses selanjutnya adalah *Text Vectorization* Tujuan dari *Text Vectorization* adalah untuk merepresentasikan kata-kata dan dokumen dalam bentuk yang dapat dipahami oleh model pemrosesan bahasa alami dan *machine learning*. Proses pada *dataset Vectorization* bersih dapat dilihat pada Gambar 4



Gambar 4. *Text Vectorization*

Setelah data direpresentasikan ke dalam *numerik* proses selanjutnya adalah membagi data membagi data menjadi data latih dan data uji dengan menggunakan fungsi *train_test_split* dari library *scikit-learn*. Fungsi ini membagi data menjadi dua bagian yaitu data latih (X_train) dan data uji (X_test). Serta membagi label (y_train) dan label (y_test) dari data tersebut. Proses pembagian data dapat dilihat pada gambar b n

Gambar 5. Proses pembagian data latih dan data uji

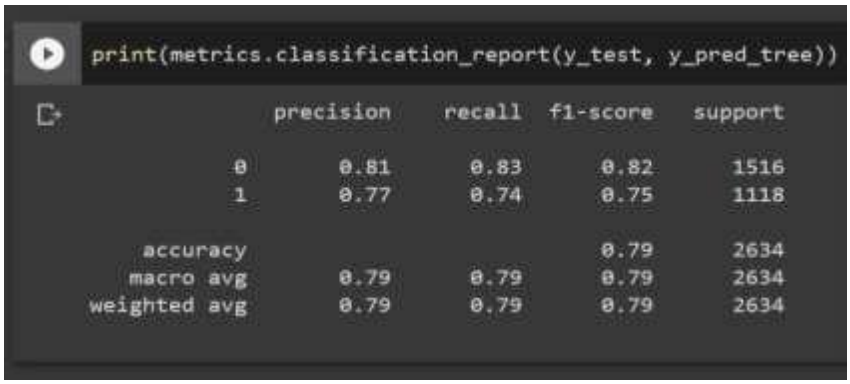
3.1.1 Model Decision Tree

Model yang direalisasikan akan menerima hasil evaluasi antara nilai 1 dan 0. Pada penelitian ini algoritma *Support Vector Machine* dan *Decision Tree* digunakan untuk klasifikasi. *Precision, Recall*, dan *F1-Score* digunakan untuk mengukur kinerja model. Hal ini dikarenakan penelitian ini hanya mengidentifikasi *Tweet* yang mengandung ujaran kebencian. Algoritma ini akan membuat keputusan berdasarkan pohon biner yang mengandung kriteria untuk mengklasifikasikan data. Pohon biner ini akan mengandung beberapa kriteria untuk mengklasifikasikan data seperti kata-kata yang berhubungan

dengan *Hate Speech*, serta konteks situasi yang mungkin berhubungan dengan hate speech. Algoritma ini akan membuat keputusan berdasarkan pohon biner dan mengklasifikasikan data sebagai *Hate Speech* atau bukan *Hate Speech*. Nilai hasil evaluasi menggunakan model *Decision Tree* dimana akurasi dihasilkan adalah 0.79 ditunjukkan pada Tabel 1

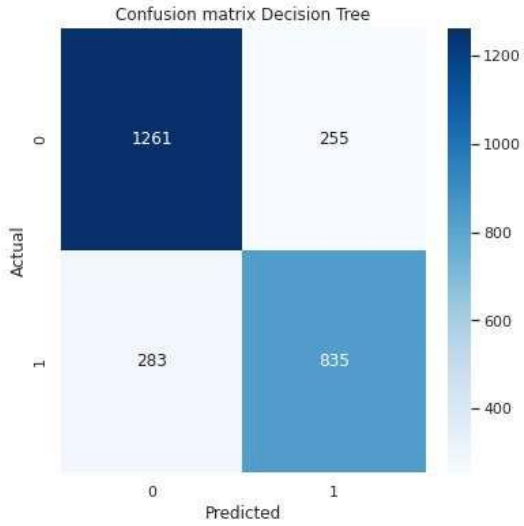
Tabel 1. Hasil Performa Model *Decision Tree*

Measures	***Decision Tree***
Accuracy	79%
Recall	83%
Precision	81%
F-1-Score	82%



Gambar 6. Hasil Performa Model *Decision Tree*

Decision Tree memiliki hasil *Accuracy* 0.79 . Model *Decision Tree* meraih nilai *Recall* 0.83 untuk label 0 dan 0.74 untuk label 1. *Precision* 0.81 untuk label 0 dan 0.77 untuk label 1, nilai *f-measure* 0.82. Untuk menghitung evaluasi model klasifikasi, biasanya digunakan beberapa *matrix* seperti *Accuracy*, *Precision*, *Recall*, dan *F1-score*. Akurasi menunjukkan seberapa baik model dalam mengklasifikasikan data *Precision* menunjukkan seberapa baik model dalam mengidentifikasi data positif. *Recall* menunjukkan seberapa baik model dalam menemukan semua data positif dan *F1-score* adalah rata-rata harmonis dari *Precision* dan *Recall*. Ada juga *Confusion Matrix* lain yang bisa digunakan tergantung pada konteks dan tujuan analisis. Metode *Decision tree* menghasilkan *Confusion Matrix* sebagai berikut



Gambar 7. *confusion matrix decision tree*

3.1.2 Decision Tree Bagging Classifier

Bagging Classifier adalah metode pembelajaran mesin yang digunakan untuk meningkatkan akurasi dari model pembelajaran mesin dengan menggunakan beberapa model yang dibangun dari data yang diambil secara acak dari *dataset* asli. Metode ini dikenal juga dengan sebutan *Bootstrap Aggregating. Bagging Classifier* biasanya digunakan pada model pembelajaran mesin yang memiliki variansi tinggi seperti *Decision Tree*. Nilai hasil evaluasi *Bagging Classifier* menggunakan dimana akurasi dihasilkan meningkat menjadi adalah 0.82 ditunjukkan pada gambar

	precision	recall	f1-score	support
0	0.82	0.87	0.84	1516
1	0.81	0.73	0.77	1118
accuracy			0.81	2634
macro avg	0.81	0.80	0.81	2634
weighted avg	0.81	0.81	0.81	2634

Gambar 8. Hasil performa *bagging classifier*

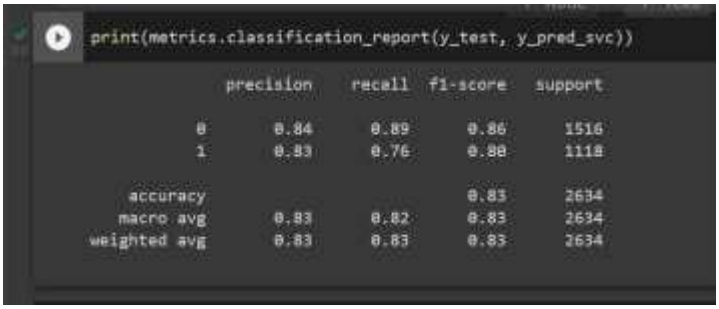
Bagging Classsifier Decision Tree memiliki hasil akurasi 0.81. Model *BaggingClassifier Decision Tree* meraih nilai *Recall* 0.87 untuk label 0 dan 0.73 untuk label 1. *Precision* 0.82 untuk label 0 dan 0.81 untuk label 1, nilai *f-measure* 0.84

3.1.3 Model Support Vector Machine

Penelitian ini menggunakan algoritma SVM pada penerapannya dalam klasifikasi, data dipisahkan menjadi data *training* dan data *testing*. Data training digunakan untuk melatih model SVM dengan memilih parameter dan kernel yang sesuai. Setelah model dilatih, data *testing* digunakan untuk menguji keakuratan model pada data yang belum pernah dilihat sebelumnya. Pada penerapan model SVM dihasilkan *Precision,Accuracy,Recall* dan *F1-Score* yang bisa dilihat pada table 2

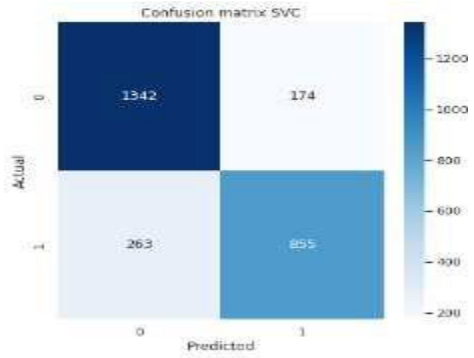
Tabel 2. Hasil Performa Model *Support Vector Machine*

Measures	**SVM**
Accuracy	83%
Recall	89%
Precision	84%
F-1-Score	86%



Gambar 9. Hasil performa *Support Vector Machine*

Support Vector Machine memiliki hasil akurasi 0.83. Model *Support Vector Machine* meraih nilai *Recall* 0.89 untuk label 0 dan 0.76 untuk label 1. *Precision* 0.84 untuk label 0 dan 0.83 untuk label 1. nilai *f-measure* 0.86

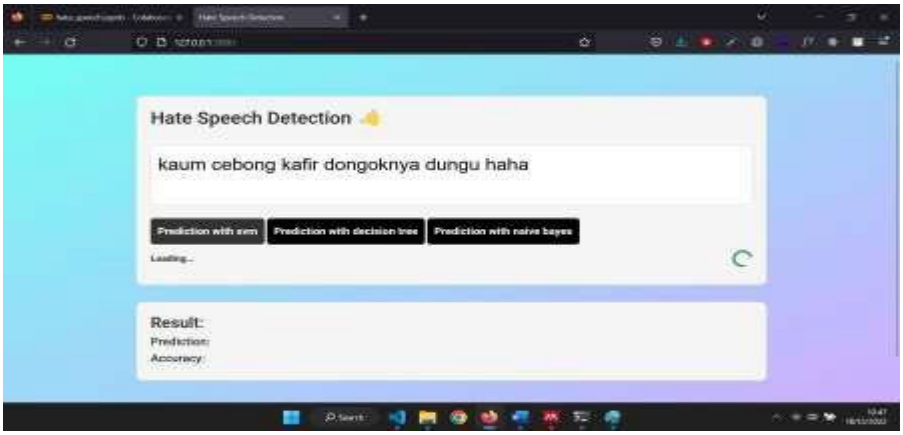


Gambar 10. *Confusion matrix support vector machine*

3.1.4 Implementasi di website

Framework Flask dapat digunakan untuk membuat aplikasi web pendeteksi ujaran kebencian. Aplikasi tersebut akan memungkinkan pengguna untuk memasukkan teks dan kemudian akan menganalisis teks tersebut menggunakan algoritma pembelajaran mesin untuk mendeteksi ujaran kebencian aplikasi kemudian dapat memberikan laporan ringkasan tentang ujaran kebencian yang terdeteksi.

Ajax (Asynchronous JavaScript and XML) adalah teknologi yang memungkinkan aplikasi web mengirim dan menerima data dari server tanpa mengganggu status halaman web saat ini. Ini adalah teknologi sisi klien yang digunakan untuk membuat halaman web yang cepat dan dinamis. Itu menggunakan *JavaScript*, XML, dan HTML. Dengan bantuan *jQuery*, permintaan *Ajax* dapat dilakukan dengan mudah.website terdiri dari halaman utama dan element form dan beberpa opsi algoritma yang dipilih



Gambar 11. Contoh halaman utama aplikasi flask

Gambar 12. Contoh hasil prediksi diaplikasi berbasis website

3.2 Pembahasaan

Pengujian pada model *Decision Tree* dilakukan dalam dua skema yaitu menggunakan pengklasifikasi *BaggingClassifier* dan *Single Classifier* untuk melihat mana yang kinerjanya lebih baik. Ini dapat membantu untuk memahami dampak penggunaan metode pada performa model. *BaggingClassifier* menghasilkan model yang lebih stabil dan kokoh. Model *Decision Tree* tanpa *BaggingClassifier* menghasilkan perbandingan hasil yang dapat dilihat pada tabel 3

Table 3. Hasil Perbandingan Akurasi *Decision Tree* dan *Bagging Classifier*

	Decision Tree	
	Dengan *BaggingClassifier*	**Tanpa *Bagging Classifier***
Accuracy	0.8189	0.7889

Dari table perbandingan tersebut menunjukkan bahwa menggunakan *BaggingClassifier* untuk mendeteksi *Hate Speech* bisa menghasilkan akurasi menjadi lebih baik dan stabil pada model *Decision Tree*. Dari perbandingan tersebut bisa dilihat dari *Confusion Matrix* sebagai berikut

Tabel 4. Hasil *Confusion Matrix Model Decision Tree*

True positive **TP** 1252	*False Positive* **FP** 264
False negative **FN** 292	*True negative* **TN** 826

Tabel 5. Hasil *Confusion Matrix BaggingClassifier*

True positive **TP** 1338	*False Positive* **FP** 178
False negative **FN** 299	*True negative* **TN** 819

Table diatas menunjukan bahwa tanpa *Bagging Classifier* dapat melakukan klasifikasi data sebanyak 1252 data pada *True positive* atau *tweet* yang mengandung *Hate Speech.* 264 data pada *false positive* atau kesalahan dalam prediksi yang mengandung *hate speech*. 826 data pada *true negative* atau *tweet* yang tidak mengandung *hate speech*. Kemudian 229 data pada *false negative* atau kesalahan prediksi pada *tweet* yang tidak mengandung *Hate Speech*. Sedangkan pada model yang menggunakan *BaggingClassifier* dapat melakukan klasifikasi data sebanyak 1334 data pada *True positive* atau tweet yang mengandung *hate speech*. Kemudian sebanyak 182 data pada *false positive* atau kesalahan dalam prediksi yang mengandung *hate speech*. Kemudian sebanyak 819 data pada *true negative* atau tweet yang tidak mengandung *hate speech*. Kemudian 299 data pada *false negative* atau kesalahan prediksi pada tweet yang tidak mengandung Hate *Speech*. Hasil *confusion matrix Decision Tree* tanpa *Baggingclassifier* diperoleh nilai *Accuracy, Precision, Recall,* dan *F-1 Score.*

$$Precision = \frac{TP}{TP+FP} x100 = \frac{1261}{1261+283} x100 = \frac{1261}{1.544} x100 = 81$$

$$F\text{-}1\ Score = \frac{2xRecallxPrecision}{Recall+Precision} = \frac{2x83x81}{83+81} = \frac{13446}{164} = 82$$

$$Recall = \frac{TP}{TP+FN} x100 = \frac{1261}{1261+255} x100 = \frac{1261}{1516} x100 = 83$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} x100 = \frac{1261+835}{1261+835+283+255} x100 = \frac{2096}{2634} x100 = 79$$

Hasil *Confusion Matrix Decision Tree* dengan *Baggingclassifier* diperoleh nilai *Accuracy, Precision, Recall,* dan *F-1 Score*

$$Precision = \frac{TP}{TP+FP} x100 \qquad F\text{-}1\ Score = \frac{2xRecallxPrecision}{Recall+Precision}$$

$$= \frac{1338}{1338+178} x100 \qquad = \frac{2x81x88}{81+88}$$

$$= \frac{1338}{1.516} x100 = 88 \qquad = \frac{14256}{169} = 85$$

$$Recall = \frac{TP}{TP+FN} x100 \qquad Accuracy = \frac{TP+TN}{TP+TN+FP+FN} x100$$

$$= \frac{1338}{1338+299} x100 \qquad = \frac{1338+819}{1338+819+178+299} x100$$

$$= \frac{1338}{1637} x100 = 81 \qquad = \frac{2157}{2634} x100 = 82$$

Sedangkan pada model *Support Vector Machine* Proses pemodelan menggunakan SVM peneliti memilih linear kernel karena dari kernel yang lain linear kernel mendapatkan hasil performa terbaik dari kernel lain dengan *Confusion Matrix* sebagai berikut

Table 6. *Confusion Matrix* SVM

True positive **TP** 1342	False Positive **FP** 174
False negative **FN** 263	True negative **TN** 855

$$Precision = \frac{TP}{TP+FP} x100 \qquad F\text{-}1\ Score = \frac{2xRecallxPrecision}{Recall+Precision}$$

$$= \frac{1342}{1342+174} x100 \qquad = \frac{2x83x84}{83+84}$$

$$= \frac{1342}{1516} x100 = 84 \qquad = \frac{13944}{167} = 82$$

$$Recall = \frac{TP}{TP+FN} x100 \qquad Accuracy = \frac{TP+TN}{TP+TN+FP+FN} x100$$

$$= \frac{1342}{1342+263} x100 \qquad = \frac{1341+855}{1342+855+174+263} x100$$

$$= \frac{1342}{1605} x100 = 83 \qquad = \frac{2096}{2634} x100 = 83$$

Perbandingan hasil evaluasi dari semua model dilihat pada table 7.

Table 7. Hasil Perbandingan Performa Seluruh Model

Models	Accuracy	Recall	Precision	F1-Score
Decision Tree	79%	83%	81%	82%
Support Vector Machine	83%	89%	84%	86%
BaggingClassifier	82%	81%	88%	85%

Berdasarkan pengujian yang dilakukan pada bagian sebelumnya, model *Support Vector Machine* terbaik untuk mengklasifikasikan ujaran kebencian dalam teks berbahasa Indonesia diperoleh dengan komposisi *F1-score* 86%, *Precision* 83%, *Recall* 89% dan *Accuracy* sebesar 83%. Beberapa faktor yang dapat mempengaruhi performa model diantaranya adalah kualitas dan kuantitas data yang digunakan, pemilihan algoritma yang sesuai, pengaturan parameter model, dan kualitas implementasi kode

3.2.1 *Data set* yang digunakan

Data set yang digunakan dalam penelitian ini dapat mempengaruhi hasil akhir dari model pendeteksian hate speech. Data set yang berkualitas tinggi dan representatif dari hate speech dapat meningkatkan akurasi model.

3.2.2 *Metode preprocessing*

Metode preprocessing yang digunakan dapat mempengaruhi hasil akhir dari model. Metode yang tepat dapat membersihkan data dan meningkatkan kualitas data yang digunakan dalam model.

3.2.3 *Teknik pemodelan*

Teknik pemodelan yang digunakan dapat mempengaruhi hasil akhir dari model. Beberapa teknik seperti deep learning, machine learning, dan NLP dapat digunakan dalam pendeteksian hate speech dan hasilnya dapat berbeda-beda.

3.2.4 *Ukuran data*

Ukuran data yang digunakan dalam penelitian ini dapat mempengaruhi hasil akhir dari model. Model yang dibangun dengan data yang cukup besar dapat meningkatkan akurasi model.

3.2.5 *Keseimbangan data*

Range jumlah antar label dari sebuah *dataset* sangat berpengaruh terhadap hasil model. Range yang terlalu jauh di setiap labelnya akan membuat model yang dibangun menjadi kurang

3.2.6 *Evaluasi*

Metode evaluasi yang digunakan dapat mempengaruhi hasil akhir dari model. Beberapa metode seperti *Accuracy, Precision, Recall,* dan *F1-Score* dapat digunakan dalam evaluasi model dan hasilnya dapat berbeda-beda.

4. PENUTUP

4.1 Kesimpulan

Dari penelitian ini merupakan sebuah aplikasi pendeteksian *hate speech* atau ujaran kebencian Mengunakan metode *Decision Tree* dan *Support Vector Machine* dengan hasil sebagai berikut:

a. Aplikasi berbasis web dengan mengunakan *framework flask* dimana peneliti bisa memasukan data atau kalimat yang mengandung *hate speech* atau pun tidak mengandung *hate speech* dengan memanfaatkan *library javascript jquery* yang digunakan untuk mengambil *value* dari *from* yang diinput kemudian melakukan klasifikasi sesuai endpoint yang telah ditentukan
b. Penerapaan metode *Decision Tree* dan *support vector machine* mengahsilkan akurasi tertinggi sebesar 83% berdasarkan evaluasi model dan prediksi yang dilakukan menunjukan hasil yang baik
c. Penerapan *BaggingClassifier* mampu meningkatkan akurasi model hal tersebuat bisa dilihat pada evaluasi model dengan mengunakan *BaggingClassifier* dan tanpa *BaggingClassifier*
d. Penerapan model *Support Vector Machine* dalam penelitian menghasilkan performa terbaik dengan menerapkan varian dari algoritma SVM yang digunakan untuk klasifikasi data linier. Dalam klasifikasi linier, SVM mencari garis atau *hyperplane* pembatas yang memisahkan dua kelas data dengan baik
e. Keseimbangan data *Range* jumlah antar label dari sebuah *dataset* sangat berpengaruh terhadap hasil model. *Range* yang terlalu jauh di setiap labelnya akan membuat model yang dibangun menjadi kurang

4.2 Saran

Adapun saran dari peneliti untuk penelitian berikutnya adalah dengan mengevaluasi model pada *dataset* yang lebih besar atau mencoba metode pembelajaran yang

berbeda.adapun saran lainya adalah bagaimana meningkatkan akurasi model atau mengurangi kompleksitas model sehingga model yang dibangun mempunyai akurasi yang maksimal

DAFTAR PUSTAKA

Perwira, A., Dwitama, J., & Hidayat, S. (2021). Identifikasi Ujaran Kebencian Multilabel Pada Teks Twitter Berbahasa Indonesia Menggunakan Convolution Neural Network. *Jurnal Sistem Komputer Dan Informatika (JSON)*, *3*, 117–127. https://doi.org/10.30865/json.v3i2.3610

Tjahyanti, L. P. A. S. (2020). Pendeteksian Bahasa Kasar (Abusive Language) Dan Ujaran Kebencian (Hate Speech) Dari Komentar Di Jejaring Sosial. *Journal of Chemical Information and Modeling*, *07*(9), 1689–1699.

Rosandy, T. (2016). Perbandingan Metode Naive Bayes Classifier dengan Metode Decision Tree Untuk Menganalisa Kelancaran Pembiayaan. *Jurnal TIM Darmajaya*, *02*(01), 52–62.

Antariksa, K., Purnomo WP, Y. S., & Ernawati, E. (2019). Klasifikasi Ujaran Kebencian pada Cuitan dalam Bahasa Indonesia. *Jurnal Buana Informatika*, *10*(2), 164. https://doi.org/10.24002/jbi.v10i2.2451

Firdaus, M., & C, T. A. (2019). *Implementasi algoritma decision tree untuk klasifikasi pola serangan pada log file*. http://repository.unmuhjember.ac.id/7150/1/JURNAL.pdf

Ihsan, F. (2021). *Implementasi Algoritma Decision Tree untuk Mendeteksi Multi-Label Hate Speech dan Abusive Language pada Twitter Bahasa Indonesia*. 1–98.

Ramadhanty, D. R. (2021). *Implementasi Algoritma Support Vector Machine Pada Analisis Sentimen Data Twitter*. 1–83. https://dspace.uii.ac.id/handle/123456789/36015

Ula, M., Ulva, A. F., Mauliza, M., Ali, M. A., & Said, Y. R. (2022). Application of Machine Learning in Determining the Classification of Children'S Nutrition With Decision Tree. *Jurnal Teknik Informatika (Jutif)*, *3*(5), 1457–1465. https://doi.org/10.20884/1.jutif.2022.3.5.599

Wijaya, Y. A., Bahtiar, A., Kaslani, & R, N. (2021). Analisa Klasifikasi menggunakan Algoritma Decision Tree pada Data Log Firewall. *Jurnal Sistem Informasi Dan Manajemen*, *9*(3), 256–264. https://ejournal.stmikgici.ac.id/

Rizky Ade Putranto, Triastuti Wuryandari, S. (2015). Perbandingan Analisis Klasifikasi Antara Decision Tree Dan Support Vector Machine Multiclass. *Jurnal Gaussian*, *4*(4), 1007–1016. Data Mining

Sari, B. N. (2016). Implementasi teknik seleksi fitur information gain pada algoritma

klasifikasi machine learning untuk prediksi performa akademik siswa. *Seminar Nasional Teknologi Informasi Dan Multimedia 2016*, 55–60.

Khadafy, A. R., & Wahono, R. S. (2015). Penerapan Naïve Bayes Untuk Mengurangi Data Noise Pada Klasifikasi Multi Kelas Dengan Decision Tree. *Journal of Intelligent Systems*, *1*(2), 11–17.

Pamungkas, F. S., Prasetya, B. D., & Kharisudin, I. (2020). Perbandingan Metode Klasifikasi Supervised Learning pada Data Bank Customers Menggunakan Python. *PRISMA, Prosiding Seminar Nasional Matematika*, *3*, 692–697. https://journal.unnes.ac.id/sju/index.php/prisma/article/view/37875

Hana, F. M. (2020). Klasifikasi Penderita Penyakit Diabetes Menggunakan Algoritma Decision Tree C4.5. *Jurnal SISKOM-KB (Sistem Komputer Dan Kecerdasan Buatan)*, *4*(1), 32–39. https://doi.org/10.47970/siskom-kb.v4i1.173