DETEKSI SMS *SPAM* BERBAHASA INDONESIA MENGGUNAKAN ALGORITMA *SUPPORT VECTOR MACHINE*

Mohamad Arif Sofyan [1], Nining Rahaningsih [2], Raditya Danar Dana [3]
[1] Teknik Informatika, STIMIK IKMI Cirebon
[2] Komputerisasi Akuntansi, STIMIK IKMI Cirebon
[3] Manajemen Informatika, STMIK IKMI Cirebon
Jl. Perjuangan No.10B, Karyamulya, Kesambi, Kota Cirebon, Jawa Barat 45131, Indonesia
arifsofyan004@gmail.com

ABSTRAK

Setiap individu membutuhkan akses informasi untuk memperluas pengetahuan mereka tentang berbagai hal. Salah satu metode yang populer dalam mengalirkan informasi adalah melalui layanan *Short Message Service* (SMS), Namun, penggunaan SMS dapat menimbulkan masalah dengan munculnya SMS spam (*Sending and Posting Advertisement in Mass*). SMS *spam* merupakan pesan teks yang tidak diinginkan atau diminta, seperti iklan, jasa, dan potensi penipuan yang dapat merugikan pengguna. Indonesia tercatat sebagai negara di Asia dengan jumlah pesan *spam* tertinggi pada tahun 2020. Untuk meminimalisir korban pesan *spam* di Indonesia, berbagai pendekatan perlu dilakukan, salah satunya melalui penyaringan *spam* SMS dengan cara mengklasifikasi SMS *spam*, algoritma yang dapat digunakan dalam masalah klasifikasi adalah *Support Vector Machine* (SVM). Serta menerapkan metode CRISP-DM (*Cross Industry Standard Process for Data Mining*) untuk mengembangkan model yang dapat memprediksi pesan *spam* atau normal, model yang sudah dibangun akan diimplementasikan kedalam Aplikasi Deteksi SMS *Spam* berbasis *Streamlit*, yang memungkinkan pengguna dapat dengan mudah menguji pesan SMS yang mereka terima, untuk mengidentifikasi apakah pesan tersebut termasuk dalam kategori *spam* atau normal. Hasil penelitian ini menunjukan bahwa model SVM yang telah dibangun berjalan dengan baik dengan menghasilkan tingkat *accuracy* sebesar 96,94%.

Kata kunci : *Klasifikasi SMS spam berbahasa Indonesia, Support Vector Machine, Aplikasi Deteksi SMS spam*

1. PENDAHULUAN

Setiap individu membutuhkan akses informasi untuk memperluas pengetahuan mereka tentang berbagai hal, salah satu metode yang populer dalam mengalirkan informasi adalah melalui layanan *Short Message Service* (SMS) [1]. SMS adalah bentuk pengiriman pesan singkat yang menggunakan jenis pesan asinkron, dimana transmisi data SMS dilakukan melalui teknik protokol simpan dan teruskan [2]. Beberapa faktor mengapa SMS sering digunakan. Salah satunya adalah karena biaya pengiriman SMS yang terjangkau, adanya bonus SMS, dan paket internet yang ditawarkan oleh penyedia layanan. Selain itu, SMS juga populer karena kemudahannya dalam penggunaan, pengiriman, penerimaan, dan membuka pesan SMS dapat dilakukan kapan saja dan dimana saja tanpa memerlukan koneksi internet [3], Namun, SMS juga rentan terhadap penyalahgunaan, seperti pengiriman pesan *spam*.

SMS *spam* adalah praktik mengirim pesan elektronik yang tidak diinginkan atau diminta oleh penerima tanpa persetujuan yang sah. Ini bisa melanggar privasi dan hukum dengan memanfaatkan data pribadi tanpa izin yang tepat, dan potensial menimbulkan kerugian [4]. Berdasarkan informasi yang terdapat dalam laporan *Truecaller Insights Report* 2020, Indonesia menempati posisi sebagai negara dengan jumlah pesan *spam* tertinggi di Asia pada tahun 2020 [5]. Data dari *Truecaller* menunjukkan bahwa jenis *spam* yang mendominasi di Indonesia yaitu layanan keuangan, mencakup 52% dari total *spam*, diikuti oleh asuransi dengan 25%, operator seluler dengan 11%, scam sebanyak 9%, dan tagihan hutang sebanyak 3% [5].

Untuk mengurangi korban dan dampak negatif yang timbul akibat pesan *spam*, berbagai pendekatan perlu dilakukan, salah satunya dengan cara mengklasifikasi SMS *spam*, menggunakan suatu metode yang bertujuan untuk mengembangkan model klasifikasi yang mampu memberikan prediksi dengan akurat apakah itu termasuk dalam kategori *spam* atau normal. Terdapat berbagai metode yang digunakan untuk menangani masalah klasifikasi, seperti *XGBoost*, *Naive Bayes*, *Decision Tree*, dan metode lainnya. Beberapa penelitian melaporkan bahwa *Support Vector Machine* (SVM) menunjukkan kinerja yang baik dalam hal akurasi untuk permasalahan klasifikasi [6].

Maka dalam penelitian ini SVM dipilih sebagai metode yang memiliki potensi untuk mengatasi tantangan klasifikasi SMS *spam*. Hasil yang diharapkan, dapat menghasilkan model klasifikasi yang optimal untuk membedakan antara SMS *spam* dan SMS normal. Kemudian model akan diimplementasikan kedalam Aplikasi Deteksi SMS *Spam* berbasis *Streamlit*. Hal ini memungkinkan pengguna dapat dengan mudah menguji pesan SMS dan mengidentifikasi apakah pesan tersebut termasuk dalam kategori *spam* atau normal, sehingga mereka bisa mengambil langkah-langkah pencegahan jika pesan tersebut tergolong sebagai *spam*, yang

diharapkan dapat meminimalisir korban pesan *spam* di Indonesia.

2. TINJAUAN PUSTAKA

Untuk memecahkan permasalahan dalam penelitian ini, beberapa *studi literatur* yang relevan diidentifikasi sebagai sumber referensi dan pertimbangan yang mendukung penelitian ini. Penelitian yang dilakukan oleh Widyawati & Susanto melibatkan perbandingan antara algoritma *Naïve Bayes* dan SVM dalam klasifikasi SMS *spam* berbahasa Indonesia. Temuan penelitian menunjukkan bahwa algoritma *Naïve Bayes* mampu mencapai tingkat akurasi sebesar 97.0%, sedangkan SVM mencapai tingkat akurasi sebesar 96.9%. Hasil ini memberikan wawasan mendalam tentang kinerja dari kedua algoritma tersebut [7].

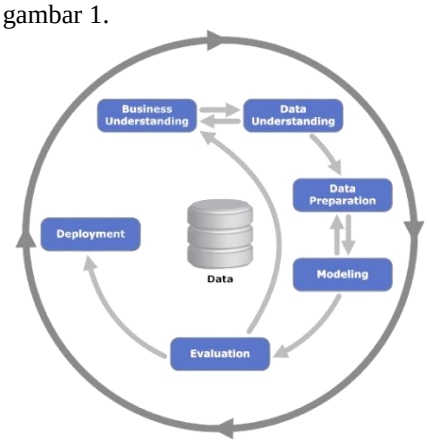
Kemudian penelitian oleh Muslikah, membahas tentang deteksi SMS *spam* menggunakan metode *Long Short-Term Memory* (LSTM). Penelitian ini bertujuan untuk mengatasi masalah peningkatan penggunaan layanan SMS untuk mengirimkan pesan *spam* yang merugikan. Hasil penelitian menunjukkan bahwa model LSTM mencapai tingkat akurasi, presisi, *recall*, dan *f-measure* sebesar 96,09% [2]. Kemudian penelitian yang dilakukan oleh Dwiyansaputra dkk, fokus pada deteksi SMS *spam* berbahasa Indonesia menggunakan metode *Term Frequency-Inverse Document Frequency* (TF-IDF) dan *Stochastic Gradient Descent*. Tujuan penelitian ini adalah untuk menangani serangan SMS *spam* yang mengganggu pengalaman pengguna perangkat seluler. Hasilnya menunjukkan bahwa model yang dibangun mencapai akurasi sebesar 97% [8].

Serta penelitian yang dilakukan oleh Theodorus, memberikan pemahaman mengenai efektivitas berbagai algoritma dalam penanganan klasifikasi SMS *spam*, seperti algoritma *Random Forest*, *Multinomial Logistic Regression*, SVM, dan *XGBoost*. Dari hasil evaluasi tersebut, SVM menunjukkan akurasi sebesar 94.38%, menandakan potensi yang dapat mengatasi tantangan klasifikasi SMS *spam* [9].

Informasi dari *studi literatur* ini memberikan landasan yang kuat bagi penelitian kami untuk mengeksplorasi strategi yang efektif dalam mengatasi masalah *spam* SMS di Indonesia.

3. METODE PENELITIAN

Penelitian ini menggunakan metodologi CRISP-DM sebagai kerangka kerja standar untuk menyelesaikan permasalahan dalam bidang *data mining*. Model CRISP-DM terdiri dari enam tahapan, yaitu *Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation,* dan *Deployment*, metode ini dapat digunakan dengan baik untuk melakukan *data mining* maupun *text mining* [10], perbedaan mendasar antara *data mining dan text mining* terletak pada format data yang diproses. *Data mining* beroperasi dengan data yang terstruktur, sementara *text mining* menghadapi data yang tidak terstruktur [10]. Berikut tahapan CRISP-DM yang akan dilaksanakan dalam penelitian ini diuraikan dalam gambar 1.



Gambar 1. Tahapan Metode CRISP-DM

3.1. *Business Understanding*

Langkah pertama dalam penelitian ini adalah memahami dengan jelas tujuan bisnis yang ingin dicapai. Dengan Indonesia menjadi negara dengan jumlah pesan *spam* tertinggi di Asia pada tahun 2020 [5], masalah pesan *spam* menjadi perhatian utama. Tingginya jumlah pesan *spam* tidak hanya mengganggu pengguna, tetapi juga meningkatkan risiko terhadap keamanan dan privasi [4], Dengan pemahaman ini, penelitian bertujuan untuk meminimalisir adanya korban dari pesan *spam* di Indonesia dengan cara mengklasifikasi SMS *spam*, kemudian model klasifikasi yang sudah dibangun akan diimplementasikan kedalam Aplikasi Deteksi SMS *Spam* berbasis *Streamlit*, yang memungkinkan pengguna dapat dengan mudah menguji pesan SMS yang mereka terima, untuk mengidentifikasi apakah pesan tersebut termasuk dalam kategori *spam* atau normal.

3.2. *Data Understanding*

Pada tahap ini, dilakukan proses pengumpulan data yang akan diolah guna mencapai tujuan penelitian ini, Data dalam penelitian ini diperoleh dari *website* milik Yudi Wibisono dan dapat diakses melalui tautan berikut ini *http://bit.ly/yw_sms_spam_indonesia*, Data ini berisi dua kolom, yaitu kolom "Teks" yang berisi isi pesan SMS, dan kolom "label" yang memiliki tiga kategori awal, SMS normal (label 0), SMS penipuan (label 1), dan SMS promo (label 2). Karena tujuan penelitian ini adalah untuk mengklasifikasikan pesan SMS sebagai SMS *spam* atau SMS normal, untuk kategori SMS penipuan dan promo disatukan menjadi label 1, yang menunjukkan bahwa pesan SMS tersebut termasuk dalam kategori SMS *spam*. Sehingga data dalam penelitian ini memiliki dua kolom, yaitu kolom "Teks," yang berisi pesan-pesan SMS dan kolom "label," yang berisi informasi apakah pesan tersebut termasuk dalam kategori SMS normal dengan label 0 atau SMS *spam* dengan label 1.

Langkah berikutnya adalah melakukan *Exploratory Data Analysis* (EDA), suatu proses yang

bertujuan untuk menyelidiki data melalui representasi visual seperti grafik atau peta. Tujuan dari EDA ini adalah untuk mempermudah analis data dalam mengidentifikasi dan menemukan pola-pola yang mungkin tersembunyi dalam dataset [11]. Beberapa analisis yang dilakukan dalam tahap ini mencakup distribusi label, frekuensi kemunculan kata, dan sebagainya.

3.3. Data *Preparation*

Tahapan berikutnya adalah data *preparation*, yang merupakan serangkaian langkah-langkah untuk mengubah data mentah menjadi data berkualitas, serta menyiapkan data SMS agar sesuai untuk pelatihan model klasifikasi. Beberapa tahapan data *preparation* dalam penelitian ini sebagai berikut.

a. *Case folding* adalah mengubah data teks menjadi bentuk standar, sehingga data akan diubah menjadi huruf kecil atau *lowercase* [11]. Langkah ini dilakukan untuk memastikan konsistensi dalam representasi teks, sehingga kata yang seharusnya dianggap sama tidak memiliki perbedaan karena huruf besar atau kecil.
b. Tokenisasi adalah proses memisahkan kata-kata secara individual dari sebuah kalimat. Tujuan dari tokenisasi adalah untuk mempermudah sistem dalam memproses teks dengan memecahnya menjadi kata-kata terpisah. Sebagai contoh, jika memiliki kalimat "saya pergi ke sekolah", hasil tokenisasinya akan berupa kata-kata "saya", "pergi", "ke", dan "sekolah" [12]. Dalam penelitian ini, tahap tokenisasi melibatkan penghapusan kata *non-alphanumeric* serta menggunakan *library Natural Language Toolkit* (NLTK) untuk melakukan tokenisasi.
c. *Stopword removal* adalah proses filtrasi atau penghilangan kata-kata yang tidak memiliki arti penting dalam sebuah teks. Pada proses tokenisasi, setiap kata akan dicek ke dalam daftar kata *stopword* dalam bahasa Indonesia. Kata-kata dalam daftar *stopword* biasanya termasuk kata-kata umum seperti "dan", "yang", "atau", "di", yang sering muncul dalam teks tetapi tidak memberikan kontribusi penting dalam pemahaman konten. Dengan menghapus *stopword*, dapat memperoleh representasi teks yang lebih fokus pada kata-kata yang memiliki makna atau informasi yang lebih relevan dalam analisis teks atau pemodelan bahasa [10]. Dalam penelitian ini, tahap *stopword removal* melibatkan penghapusan tanda baca (*punctuation*) dan kata-kata umum (*stopword*) dalam bahasa Indonesia.
d. *Stemming* adalah proses mengubah kata-kata yang memiliki imbuhan menjadi kata dasar. Hal ini dilakukan dengan tujuan mengembalikan kata-kata tersebut menjadi bentuk aslinya atau kata induk. Dengan melakukan *stemming*, kata-kata yang memiliki bentuk berimbuhan seperti awalan, akhiran, atau infleksi akan disederhanakan menjadi bentuk dasar yang memiliki arti yang sama [13]. Contohnya, kata "memakannya" menjadi "makan", "memukul" menjadi "pukul" dan "berjalan" menjadi "jalan". Pada penelitian ini, proses *stemming* menggunakan *library Sastrawi* yang menerapkan algoritma Nazief & Andriani untuk menjalankan proses *stemming*.

Tahap selanjutnya adalah pembobotan kata untuk mengkonversi kata-kata menjadi representasi vektor, sehingga dapat digunakan oleh teknologi *machine learning* dalam melakukan klasifikasi data pada tahap selanjutnya [14]. Dalam penelitian ini proses pembobotan kata menggunakan TF-IDF. *Term Frequency* (TF) mencerminkan seberapa sering kata tersebut muncul dalam dokumen, dengan nilai bobot yang semakin tinggi seiring dengan peningkatan frekuensi kemunculan. Sebaliknya, *Inverse Document Frequency* (IDF) memperhitungkan seberapa umumnya kata tersebut muncul di seluruh dokumen semakin sering kata muncul, nilai bobotnya akan semakin rendah. Persamaan TF-IDF yang diterapkan dalam penelitian ini dapat dijelaskan sebagai berikut:

$$wi,j = tfi,j \times \log(\frac{N}{dfi})$$

dengan keterangan:

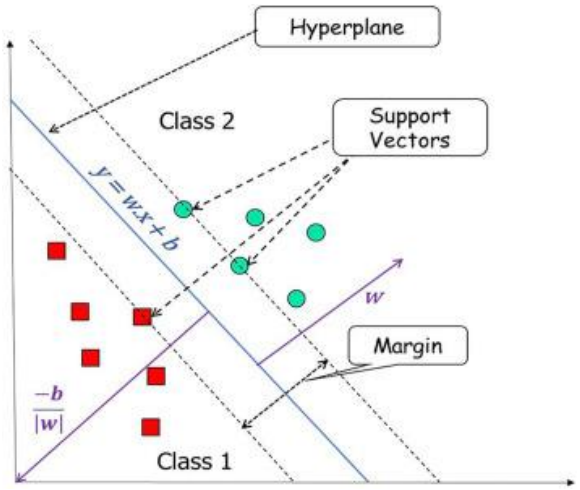
- wi,j adalah bobot dokumen ke-i terhadap kata ke-j.
- tfi,j adalah banyaknya kata i yang dicari pada sebuah dokumen j.
- N adalah total dokumen.
- dfi adalah banyaknya dokumen yang mengandung kata ke-i.

Jika suatu kata muncul dalam setiap dokumen, maka nilai *Inverse Document Frequency* ($IDFj$) akan menjadi 0. Agar dapat menghindari hasil pembobotan wi,j yang 0, strategi yang dilakukan adalah menambahkan nilai 1 pada perhitungan IDF. Penambahan ini dilakukan untuk mencegah pembagian dengan nol dan memastikan bahwa setiap *term* memiliki suatu bobot, bahkan jika *term* tersebut muncul di setiap dokumen. Penambahan nilai 1 pada hasil perhitungan IDF dapat diantisipasi oleh *library Sklearn* [15]. Dalam implementasinya, penelitian ini memanfaatkan *library Scikit-Learn* dalam bahasa pemrograman *Python* untuk membantu proses pembobotan TF-IDF.

3.4. *Modeling*

Langkah berikutnya adalah proses modeling, yang melibatkan pengembangan model klasifikasi SMS *spam* menggunakan algoritma SVM. Pertama-tama, data harus dibagi menjadi dua set terpisah yaitu data latih dan data uji. Data latih akan digunakan untuk melatih model, sementara data uji akan digunakan untuk menguji kinerja model. Proporsi pembagian data yang digunakan dalam penelitian ini adalah 80% untuk data latih dan 20% untuk data uji. Setelah membagi data, langkah berikutnya adalah mengembangkan model klasifikasi SMS *spam* menggunakan algoritma SVM. SVM adalah algoritma *machine learning* yang

digunakan untuk klasifikasi dan regresi yang bertujuan membuat batas keputusan antara kelas-kelas data untuk memprediksi label dari *vektor* data [16]. SVM diperkenalkan oleh Vapnik, dkk., pada tahun 1992 dalam *Annual Workshop on Computational Learning Theory*. Pengembangan SVM memadukan konsep-konsep komputasi yang sudah ada selama puluhan tahun, seperti *margin, hyperplane*, dan *kernel*. Berbeda dengan pendekatan *neural network* yang berusaha menemukan *hyperplane* pemisah antar kelas, SVM fokus pada pencarian *hyperplane* terbaik dalam ruang *input*. Prinsip dasar SVM adalah sebagai *linear classifier*, namun selanjutnya dikembangkan untuk menangani masalah *non-linear* dengan memperkenalkan konsep *kernel trick* dalam ruang kerja berdimensi tinggi [17].



Gambar 2. Algoritma SVM [18]

Langkah selanjutnya adalah optimasi model yang melibatkan penentuan *hyperparameters*, yakni *parameter-parameter* pada metode yang diatur sebelum melibatkan proses pelatihan pada data, baik secara manual maupun menggunakan algoritma pencarian. Penentuan *hyperparameters* ini bertujuan untuk mencapai hasil optimal yang dapat memiliki pengaruh terhadap akurasi dalam model ketika melakukan prediksi pada data yang sama. Dalam konteks umum, hampir semua metode *machine learning* tidak dapat menghasilkan hasil yang optimal jika *parameter* tidak disesuaikan dengan tepat [19].

Optimasi model dalam penelitian ini akan menggunakan metode *GridSearch*. *Grid search* adalah metode pencarian yang menyeluruh berdasarkan *sub-set hyper-parameter* yang telah ditentukan sebelumnya. *Hyper-parameters* ini ditentukan dengan mempertimbangkan rentang nilai dari yang terendah hingga yang tertinggi, sehingga mencakup seluruh kemungkinan nilai yang dapat diuji selama proses pencarian [19].

Selain itu parameter SVM yang digunakan adalah *parameter C*, *C* merupakan *parameter* yang menentukan sejauh mana kesalahan dalam klasifikasi data akan dikenai sanksi. Semakin tinggi nilai *C*, semakin besar sanksi yang diberikan terhadap kesalahan dalam klasifikasi [19].

Kemudian *kernel* yang digunakan adalah *radial basis function* (RBF). *Kernel* ini memiliki *parameter* yang dapat memengaruhi tingkat akurasi, yaitu *gamma* (γ). *Gamma* berperan dalam memetakan data latih ke dalam ruang *input* berdimensi tinggi (*feature space*), dan juga menentukan sejauh mana pengaruh setiap data terhadap hasil akhir [19]. Serta menggunakan metode *cross-validation* (CV), CV merupakan suatu metode evaluasi kinerja model yang melibatkan pembagian data menjadi beberapa *subset* atau *fold*, di mana model dilatih pada beberapa *subset* dan diuji pada *subset* lainnya secara berulang untuk mengestimasi kesalahan prediksi dalam mengevaluasi kinerja model [20].

Setelah melakukan optimasi model dengan menentukan *hyperparameters* menggunakan metode *GridSearch* dan mengatur *parameter* SVM seperti *parameter C* dan *kernel* RBF dengan *gamma* (γ), Langkah berikutnya adalah *training* model, pada tahap ini, model akan belajar dari data latih untuk meningkatkan keterampilanya dalam mengenali dan mengklasifikasikan pesan SMS. Kemudian model akan diuji dengan data uji, yang merupakan dataset terpisah dari data latih. Hal ini bertujuan untuk memberikan gambaran sejauh mana model dapat memgeneralisasi dan melakukan prediksi pada data yang belum pernah dilihat sebelumnya.

3.5. *Evaluation*

Pada tahap ini model akan di evaluasi, menggunakan *Confusion Matrix*, *confusion matrix* adalah suatu bentuk *matriks* yang digunakan untuk memberikan laporan hasil dari suatu klasifikasi (*classification report*) [21]. Fungsi ini mengevaluasi klasifikasi berdasarkan prediksi dan nilai *actual*, beberapa istilah penanda yang terdapat pada *confusion matrix* yaitu, *True Positive* (TP) yang menyatakan jumlah data positif yang benar diklasifikasikan sebagai positif, *True Negative* (TN) yang menyatakan jumlah data negatif yang benar diklasifikasikan sebagai negatif, *False Positive* (FP) yang menyatakan jumlah data negatif yang salah diklasifikasikan sebagai positif, dan *False Negative* (FN) yang menyatakan jumlah data positif yang salah diklasifikasikan sebagai negatif [22]. Setelah mendapatkan nilai dari *confusion matrix* tersebut dapat menghitung beberapa metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *f1-score* dengan rumus sebagai berikut:

1. $Rumus\ Accuracy$: $\frac{TP+TN}{TP+TN+FP+FN}$
2. $Rumus\ Precision$: $\frac{TP}{TP+FP}$
3. $Rumus\ Recall$: $\frac{TP}{TP+FN}$
4. $Rumus\ F1-Score$: $2 \times \frac{Precision \times Recall}{Precision + Recall}$

3.6. *Deployment*

Setelah tahap evaluasi langkah selanjutnya adalah *deployment* model. Pada tahapan ini, model

yang telah dibangun akan diimplementasikan dalam sebuah aplikasi deteksi SMS *spam* berbasis *Streamlit, sebuah* pustaka *Python* sumber terbuka yang memudahkan untuk membuat dan berbagi aplikasi web yang sesuai kebutuhan untuk pembelajaran mesin dan ilmu data [23]. Aplikasi ini didesain untuk memberikan kemudahan bagi pengguna dalam mengidentifikasi apakah pesan SMS yang diterima termasuk dalam kategori *spam* atau normal. Fitur-fitur yang ada dalam aplikasi ini mencakup *input* pesan SMS, di mana pengguna dapat memasukkan pesan yang ingin mereka identifikasi, dan hasil deteksi yang jelas yang akan memberikan informasi tentang apakah pesan tersebut dikategorikan sebagai *spam* atau tidak, beserta nilai probabilitasnya.

Proses *deployment* ini melibatkan beberapa langkah. Pertama-tama, model klasifikasi SMS *spam* disimpan dalam format *pickle*. Model yang telah disimpan kemudian diintegrasikan ke dalam aplikasi deteksi SMS *spam* yang telah dirancang. Integrasi model ke dalam aplikasi memungkinkan aplikasi untuk melakukan prediksi apakah suatu pesan SMS termasuk dalam kategori *spam* atau normal berdasarkan pola yang telah dipelajari oleh model. Integrasi ini memungkinkan aplikasi untuk melakukan prediksi secara otomatis saat pengguna memasukkan pesan. Dengan adanya aplikasi ini diharapkan dapat memberikan bantuan kepada pengguna ponsel untuk dengan cepat mengidentifikasi pesan yang diterima. Hal ini memungkinkan mereka untuk mengambil langkah-langkah pencegahan yang tepat jika pesan tersebut tergolong sebagai *spam*, sehingga dapat meminimalisir korban pesan *spam* di Indonesia.

4. HASIL DAN PEMBAHASAN

4.1. Implementasi Algoritma SVM

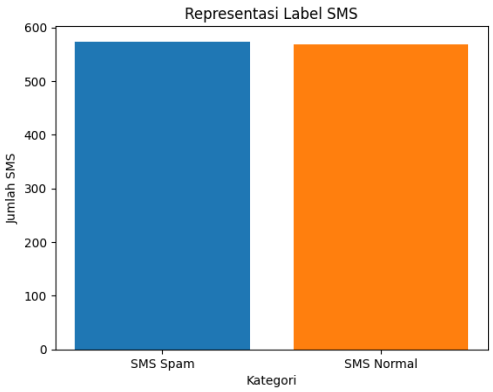
	Teks	label
0	[PROMO] Beli paket Flash mulai 1GB di MY TELKO...	1
1	2.5 GB/30 hari hanya Rp 35 Ribu Spesial buat A...	1
2	2016-07-08 11:47:11.Plg Yth, sisa kuota Flash ...	1
3	2016-08-07 11:29:47.Plg Yth, sisa kuota Flash ...	1
4	4.5GB/30 hari hanya Rp 55 Ribu Spesial buat an...	1
...	...	...
1138	Yooo sama2, oke nanti aku umumin di grup kelas	0
1139	sebelumnya ga ad nulis kerudung. Kirain warn...	0
1140	Mba mau kirim 300 ya	0
1141	nama1 beaok bwrangkat pagi...mau cas atay tra...	0
1142	No bri atas nama kamu mana	0

1143 rows × 2 columns

Gambar 3. Sample Isi Data

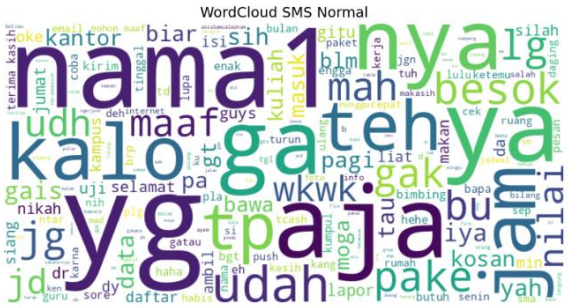
Pada tahap awal, dilakukan proses pengumpulan data yang akan diolah guna mencapai tujuan penelitian, kemudian dilakukan pemeriksaan untuk mengidentifikasi data duplikat. Setiap data yang teridentifikasi sebagai duplikat akan dihapus, sehingga hanya tersisa satu SMS untuk setiap data tersebut. Total keseluruhan dataset dalam penelitian ini berjumlah 1623 baris dan terdiri dari 2 kolom. Berikut adalah sample isi dataset dapat dilihat pada gambar 3.

Kemudian dilakukan EDA, hasil temuan dari proses EDA, total jenis SMS normal terdiri dari 794 data, sementara total jenis SMS *spam* terdiri 829 data. Hasil ini menunjukkan bahwa dataset memiliki distribusi yang relatif seimbang antara SMS normal dan SMS *spam*, memungkinkan model klasifikasi untuk mendapatkan wawasan yang baik dari kedua kategori tanpa adanya ketidakseimbangan yang signifikan. Berikut representasi label sms dapat dilihat pada gambar 4.



Gambar 4 Representasi Label SMS

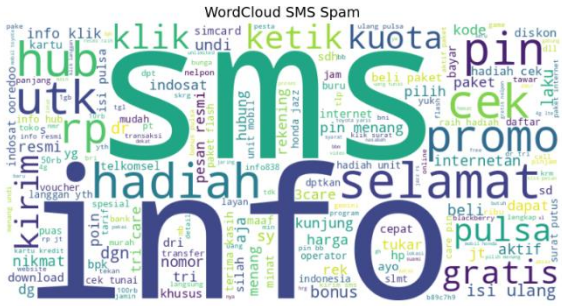
Serta visualisasi kata, yang bertujuan untuk mengidentifikasi frekuensi kemunculan kata dalam data. Ukuran font pada visualisasi menunjukkan seberapa sering suatu kata muncul, dengan ukuran font yang lebih besar menandakan frekuensi yang lebih tinggi. Ditemukan bahwa kata-kata seperti "sms," "info," "selamat" dan "hadiah" sering muncul dalam SMS *spam*. Sementara kata-kata seperti "yg," "ya", "nama," dan "aja" sering muncul dalam SMS normal. Berikut adalah gambar visualisasi SMS normal dapat dilihat pada gambar 5 dan SMS *spam* pada gambar 6



Gambar 5. Visualisasi kata SMS Normal

Tahapan berikutnya adalah data *preparation*, yang merupakan serangkaian langkah-langkah untuk mempersiapkan teks yang akan digunakan dalam proses *modeling*. Langkah-langkah ini *meliputi Case Folding*, Tokenisasi, *Stopword removal*, dan *Stemming*. Tujuannya adalah untuk menyederhanakan teks dan meningkatkan kualitasnya sebelum masuk ke

tahap *modeling*. Berikut adalah rangkaian *code* data *preparation* yang dapat dilihat dalam gambar 7.



Gambar 6. Visualisasi kata SMS *Spam*

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from nltk.corpus import stopwords
import string
import nltk

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

# Fungsi untuk langkah stemming bahasa Indonesia
def stemming(text):
    # Melakukan proses stemming pada teks
    return stemmer.stem(text)

# Fungsi untuk mengubah teks SMS
def transform_text(text):
    # 1. Mengubah teks menjadi huruf kecil
    text = text.lower()

    # 2. Tokenisasi teks dan menghapus karakter non-alphanumeric
    tokens = [i for i in nltk.word_tokenize(text) if i.isalnum()]

    # 3. Menghapus stopwords (kata-kata umum) dalam bahasa Indonesia dan tanda baca
    tokens = [i for i in tokens if i not in stopwords.words('indonesian') + list(string.punctuation)]

    # 4. Stemming (menghilangkan imbuhan) dari kata-kata dalam teks
    tokens = [stemming(i) for i in tokens]

    return " ".join(tokens)
```

Gambar 7. Rangkaian *Code Preprocessing* Data

Tabel 1 menunjukkan hasil dari tahap data *preparation*. Teks sebelum proses data *preparation* ditransformasi menjadi teks yang telah diproses, seperti yang terlihat dalam tabel tersebut. Proses ini menghasilkan representasi teks yang lebih bersih dan terstruktur, yang dapat meningkatkan kinerja model klasifikasi yang akan dibangun.

Tabel 1. Hasil Tahap Data *preparation*

Teks sebelum tahap data *preparation*	**Teks setelah tahap *data preparation***
Anda Telah Resmi meraih Hadiah Ke 2 Mendapat Cek Tunai Rp.37 juta,Kode ID PIN pemenang Anda (02844254) Untuk Info Lebih Jelas Kunjungi www. galerypemenang2018.cf	resmi raih hadiah 2 cek tunai juta kode id pin menang 02844254 info kunjung www

Tahap selanjutnya adalah proses *modeling*, Pada tahap ini, dilakukan langkah-langkah untuk mengembangkan model klasifikasi yang optimal menggunakan algoritma SVM. Berikut adalah rangkaian langkah-langkah yang dilakukan dalam tahap *modeling*:

a. Pembagian Data latih dan Data uji
   Hasil pembagian data dengan proporsi 80:20 pada penelitian ini, menghasilan data latih sebanyak 1298 data dan data uji sebanyak 325 data.

b. Optimasi Model
   Optimasi model dalam penelitian ini melibatkan pembentukan *parameter grid* untuk meningkatkan kinerja algoritma SVM. Dalam hal ini, *parameter* SVM yang dioptimalkan adalah *C* dan *gamma*. Nilai-nilai yang akan diuji untuk kedua *parameter* ini dapat dilihat pada Tabel 2.

Tabel 2. Daftar Nilai *Parameter Grid* yang Diuji

No	***Parameter***	**Nilai**
1	*C*	[1e-3, 1e-2, 1e-1, 1, 10, 100, 1000]
2	*Gamma*	[1e-3, 1e-2, 1e-1, 1, 10, 100, 1000]

Selanjutnya, pembuatan *pipeline* untuk menggabungkan serangkaian langkah pemrosesan dan pembentukan model secara terstruktur. Langkah-langkah ini dapat divalidasi silang secara bersamaan, sambil mengatur *parameter* yang berbeda pada setiap langkahnya. Dalam penelitian ini, *pipeline* mencakup pembobotan kata menggunakan metode TF-IDF, dengan menggunakan dua jenis *n-gram*, yaitu *unigram* dan *bigram*. Selanjutnya, implementasi algoritma SVM dengan *kernel radial basis function* (RBF), serta mengaktifkan fitur *probability*, Pengaktifan opsi *probability* ini memungkinkan model SVM memberikan prediksi bersamaan dengan nilai probabilitasnya. Serta menggunakan metode *GridSearchCV*. Proses *GridSearchCV* melibatkan pelatihan model dengan setiap kombinasi pada *parameter grid* menggunakan teknik *Cross-Validation* (CV).

Dalam penelitian ini, menerapkan 10 CV, yang berarti data dibagi menjadi 10 bagian, dan model dilatih serta diuji sebanyak 10 kali. Model dengan *parameter* terbaik, yang menghasilkan kinerja tertinggi akan dipilih. Berikut adalah gambar rangkaian *code modeling*.

```
# Definisikan parameter grid yang ingin diuji
svm_params = {
    'algo__C': [1e-3, 1e-2, 1e-1, 1, 10, 100, 1000],
    'algo__gamma': [1e-3, 1e-2, 1e-1, 1, 10, 100, 1000]
}

# Buat pipeline
pipeline = Pipeline([
    ('prep', TfidfVectorizer(ngram_range=(1, 2))),
    ('algo', SVC(kernel='rbf', probability=True))
])

# Buat model GridSearchCV
model = GridSearchCV(pipeline, svm_params, cv=10, n_jobs=-1, verbose=1)

# Latih model
model.fit(X_train, y_train)

# Tampilkan hasil
print("Best Parameters:", model.best_params_)
print("Training Accuracy:", model.score(X_train, y_train))
print("Model Best Score:", model.best_score_)
print("Test Accuracy:", model.score(X_test, y_test))
```

Gambar 8. Rangkaian *Code Modeling*

c. *Training* Model
   Dalam proses *training* model dengan daftar nilai *parameter grid* dan metode *GridSearchCV* yang menerapkan 10-*fold* CV, dihasilkan 49 kombinasi *parameter* yang diuji dengan total iterasi *fitting* mencapai 490. Berdasarkan hasil evaluasi

menunjukkan bahwa *parameter* terbaik untuk model dalam penelitian ini adalah *parameter C* dengan nilai 10 dan *parameter gamma* dengan nilai 1. Hasil akurasi pelatihan mencapai 100%, menunjukkan bahwa model dapat mempelajari data latih dengan baik. Serta model terbaik pada proses validasi silang (*cross-validation*) mencapai sekitar 96.6%, mengindikasikan performa model yang baik pada data yang belum pernah dilihat sebelumnya. Selanjutnya, pada tahap pengujian dengan menggunakan data uji, akurasi mencapai 96.94%, menunjukkan kemampuan model untuk mengklasifikasikan data dengan tingkat keakuratan yang tinggi pada dataset yang tidak digunakan dalam proses pelatihan. Berikut gambar hasil *training* model dapat dilihat pada gambar 9.

```
Fitting 10 folds for each of 49 candidates, totalling 490 fits
Best Parameters: {'algo__C': 10, 'algo__gamma': 1}
Training Accuracy: 1.0
Model Best Score: 0.9660535117056857
Test Accuracy: 0.9694323144104804
```

Gambar 9. Hasil *Training* Model

4.2. Evaluasi menggunakan Matrik Klasifikasi

Dari hasil evaluasi model menggunakan *confusion matrix*, model klasifikasi menghasilkan 108 TP, 6 FP, 1 FN, dan 114 TN. Hasil evaluasi lainnya seperti *precision*, *recall*, dan *F1-score* untuk kedua kelas menunjukkan performa yang sangat baik. Dengan *precision* sebesar 99%, model mampu memberikan hasil yang akurat dalam mengklasifikasikan pesan sebagai *spam* atau normal. *Recall* sebesar 95% menunjukkan bahwa model dapat mendeteksi sebagian besar pesan yang seharusnya diklasifikasikan sebagai *spam*. Serta *F1-score*, yang mencapai nilai 97%, memberikan gambaran holistik tentang keseimbangan antara *precision* dan *recall*. Hasil evaluasi ini menunjukan bahwa model klasifikasi yang diimplementasikan memiliki kinerja yang sangat baik dalam menangani klasifikasi pesan *spam*. Dengan demikian, model ini dapat dianggap sebagai solusi yang handal dan efektif dalam mengatasi permasalahan klasifikasi pesan *spam* pada dataset yang digunakan dalam penelitian ini. Berikut hasil *confusion matrix* dapat dilihat pada Gambar 10, dan *classification report* dapat dilihat pada Gambar 11 berikut ini:

```
Confusion Matrix:
             Predicted Ham  Predicted Spam
Actual Ham             108               6
Actual Spam              1             114

Metrics:
====================
True Negative (TN): 108
False Positive (FP): 6
False Negative (FN): 1
True Positive (TP): 114
```

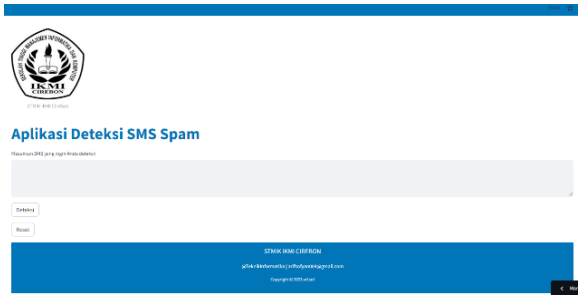
Gambar 10. Hasil *Confusion Matrix*

	precision	recall	f1-score	support
0	0.99	0.95	0.97	114
1	0.95	0.99	0.97	115
accuracy			0.97	229
macro avg	0.97	0.97	0.97	229
weighted avg	0.97	0.97	0.97	229

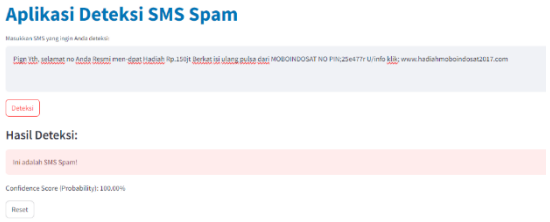
Gambar 11. Hasil *Classification Report*

4.3. Implementasi Model

Setelah memperoleh model yang optimal, langkah selanjutnya adalah mengimplementasikan model kedalam Aplikasi Deteksi SMS *Spam* berbasis *Streamlit.* Dengan adanya aplikasi ini, pengguna dapat dengan mudah mengidentifikasi pesan yang mereka terima, apakah pesan tersebut termasuk dalam kategori SMS *spam* atau SMS normal. Sehingga mereka bisa mengambil langkah-langkah pencegahan jika pesan tersebut teridentifikasi sebagai *spam*. Serta diharapkan dapat meminimalisir korban pesan *spam* di Indonesia. Untuk mencoba aplikasi ini silahkan mengunjungi link berikut *https://aplikasideteksismsspam.streamlit.app/*. Berikut adalah gambaran aplikasi deteksi sms *spam* dapat dilihat pada gambar 12 dan contoh penggunaan aplikasi deteksi sms *spam* dapat dilihat pada gambar 13.



Gambar 12. Tampilan Aplikasi Deteksi SMS *Spam*



Gambar 13. Contoh Penggunaan Aplikasi Deteksi SMS *Spam*

5. KESIMPULAN DAN SARAN

Kesimpulan dari penelitian ini adalah bahwa algoritma SVM berhasil dalam mengklasifikasikan SMS *spam* berbahasa Indonesia dengan baik, *parameter* terbaik yang ditemukan adalah *C* dengan nilai 10 dan *gamma* dengan nilai 1, dan model menghasilkan tingkat *accuracy* sebesar 96,94%. Serta model dapat diimplementasi kedalam Aplikasi Deteksi SMS *Spam* berbasis *Streamlit* yang memungkinkan pengguna untuk mengidentifikasi pesan *spam* dengan

cepat dan akurat, dengan harapan dapat mengurangi korban pesan *spam* di Indonesia. Saran untuk penelitian selanjutnya mencakup penggunaan dataset yang lebih besar dan representatif, eksplorasi metode pengolahan data dan pemilihan fitur yang lebih baik, serta penambahan sistem bot untuk memberikan laporan terhadap pesan-pesan *spam* yang tidak terdeteksi.

DAFTAR PUSTAKA

[1] D. Irawan, E. B. Perkasa, Y. Yurindra, D. Wahyuningsih, and E. Helmud, "Perbandingan Klassifikasi SMS Berbasis Support Vector Machine, Naive Bayes Classifier, Random Forest dan Bagging Classifier," *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, vol. 10, no. 3, 2021, doi: 10.32736/sisfokom.v10i3.1302.

[2] A. N. Muslikah, "SMS Spam Detection Menggunakan Metode Long Short Term Memory," 2021.

[3] E. A. Pranata, S. Subari, and G. F. Gunawan, "Penerapan Metode Naïve Bayes Untuk Klasifikasi Sms Spam Menggunakan Java Rogramming," *J-INTECH*, vol. 7, no. 02, 2019, doi: 10.32664/j-intech.v7i02.435.

[4] M. A. and R. A. Ramadhan, *KLASIFIKASI TEXT SPAM MENGGUNAKAN METODE SUPPORT VECTOR MACHINE DAN NAÏVE BAYES*. 2022. Accessed: Aug. 28, 2023. [Online]. Available: https://scholar.google.com/scholar?scilib=1&scioq=related:1sYQOOuW1B4J:scholar.google.com/&hl=id&as_sdt=0,5&gmla=AOV7GLNtD2Rv2hAyVLpdoSO7mop3fzX8bKh3ipXHzJp2fsXYOeL5yniqgNSd2H9MiNas74QEyJZD_vTxinsLPB7eS--rRB2yZ-4&sciund=3965223889334162719

[5] Kim Fai Kok, "Top 20 Countries Affected by Spam Calls in 2020."

[6] M. Fajri and A. Primajaya, "Komparasi Teknik Hyperparameter Optimization pada SVM untuk Permasalahan Klasifikasi dengan Menggunakan Grid Search dan Random Search," *Journal of Applied Informatics and Computing*, vol. 7, no. 1, 2023, doi: 10.30871/jaic.v7i1.5004.

[7] Widyawati and Susanto, "Perbandingan Algoritma Naïve Bayes Dan Support Vector Machine (Svm) Dalam Klasifikasi Sms Spam Berbahasa Indonesia," *Jurnal Ilmiah Sains Dan Teknologi*, vol. 3, no. 2, 2019.

[8] R. Dwiyansaputra, G. S. Nugraha, F. Bimantoro, and A. Aranta, "Deteksi Sms Spam Berbahasa Indonesia Menggunakan Tf-Idf Dan Stochastic Gradient Descent Classifier," *Jurnal Teknologi Informasi, Komputer dan Aplikasinya*, vol. 3, no. 2, 2021.

[9] A. Theodorus, T. K. Prasetyo, R. Hartono, and D. Suhartono, "Short Message Service (SMS) Spam Filtering using Machine Learning in Bahasa Indonesia," in *3rd 2021 East Indonesia Conference on Computer and Information Technology, EIConCIT 2021*, 2021. doi: 10.1109/EIConCIT50028.2021.9431859.

[10] H. Herwanto, N. L. Chusna, and M. S. Arif, "Klasifikasi SMS Spam Berbahasa Indonesia Menggunakan Algoritma Multinomial Naïve Bayes," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 5, no. 4, 2021, doi: 10.30865/mib.v5i4.3119.

[11] F. Reviantika, Y. Azhar, G. I. Marthasari, Wicaksono, and Triyono, "Analisis Klasifikasi SMS Spam Menggunakan Logistic Regression," *Jurnal Sistem Cerdas*, vol. 4, no. 2, 2021.

[12] K. A. Nugraha and H. Herlina, "Klasifikasi Pertanyaan Bidang Akademik Berdasarkan 5W1H menggunakan K-Nearest Neighbors," *Jurnal Edukasi dan Penelitian Informatika (JEPIN)*, vol. 7, no. 1, 2021, doi: 10.26418/jp.v7i1.45322.

[13] M. A. Rofiqi, Abd. C. Fauzan, A. P. Agustin, and A. A. Saputra, "Implementasi Term-Frequency Inverse Document Frequency (TF-IDF) Untuk Mencari Relevansi Dokumen Berdasarkan Query," *ILKOMNIKA: Journal of Computer Science and Applied Informatics*, vol. 1, no. 2, 2019, doi: 10.28926/ilkomnika.v1i2.18.

[14] N. A. Susanti, M. Walid, and H. Hoiriyah, "KLASIFIKASI DATA TWEET UJARAN KEBENCIAN DI MEDIA SOSIAL MENGGUNAKAN NAIVE BAYES CLASSIFIER," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 6, no. 2, 2022, doi: 10.36040/jati.v6i2.5174.

[15] A. Apriani, H. Zakiyudin, and K. Marzuki, "Penerapan Algoritma Cosine Similarity dan Pembobotan TF-IDF System Penerimaan Mahasiswa Baru pada Kampus Swasta," *Jurnal Bumigora Information Technology (BITe)*, vol. 3, no. 1, 2021, doi: 10.30812/bite.v3i1.1110.

[16] V. Phoan and J. Setiawan, "SENTIMENT ANALYSIS OF COMMENTS ON SEXUAL HARASSMENT IN COLLEGES ON FOUR POPULAR SOCIAL MEDIA," *Journal of Multidisciplinary Issues*, vol. 2, no. 2, 2022, doi: 10.53748/jmis.v2i2.33.

[17] Teguh Arifianto, S. Sunaryo, and Lady Silk Moonlight, "PENGGUNAAN METODE SUPPORT VECTOR MACHINE (SVM) PADA TEKNOLOGI MOBIL MASA DEPAN MENGGUNAKAN SIDIK JARI," *Jurnal Teknik Informatika dan Teknologi Informasi*, vol. 2, no. 2, 2022, doi: 10.55606/jutiti.v2i2.363.

[18] A. Rani, N. Kumar, J. Kumar, and N. K. Sinha, "Machine learning for soil moisture assessment," in *Deep Learning for Sustainable Agriculture*, 2022. doi: 10.1016/B978-0-323-85214-2.00001-X.

[19] M. Alkaff, A. R. Baskara, and I. Maulani, "Klasifikasi Laporan Keluhan Pelayanan Publik Berdasarkan Instansi Menggunakan Metode LDA-SVM," *Jurnal Teknologi Informasi dan*

Ilmu Komputer, vol. 8, no. 6, 2021, doi: 10.25126/jtiik.2021863768.

[20] L. Mardiana, D. Kusnandar, and N. Satyahadewi, "Analisis Diskriminan Dengan K Fold Cross Validation Untuk Klasifikasi Kualitas Air Di Kota Pontianak," *Buletin Ilmiah Mat. Stat. dan Terapannya (Bimaster)*, vol. 11, no. 1, 2022.

[21] E. Dwi Pratama, "Implementasi Model Long-Short Term Memory (LSTM) pada Klasifikasi Teks Data SMS Spam Berbahasa Indonesia," *The Journal on Machine Learning and Computational Intelligence (JMLCI)*, vol. 1, no. 2, 2022.

[22] D. Putra and A. Wibowo, "Prediksi Keputusan Minat Penjurusan Siswa SMA Yadika 5 Menggunakan Algoritma Naïve Bayes," *Prosiding Seminar Nasional Riset Dan Information Science (SENARIS)*, vol. 2, 2020.

[23] Inc. Streamlit, "Streamlit: The fastest way to create data apps.," https://www.streamlit.io/.