Aplikasi Klasifikasi SMS Berbasis Web Menggunakan Algoritma Logistic Regression

Fitran Dwi Pramakrisna[1], Faisal Dharma Adhinata[2*], Nia Annisa Ferani Tanjung[3]

[1,2,3] Program Studi Rekayasa Perangkat Lunak, Institut Teknologi Telkom Purwokerto, Jawa Tengah
Email: [1]18104034@ittelkom-pwt.ac.id, [2*]faisal@ittelkom-pwt.ac.id, [3]nia@ittelkom-pwt.ac.id

(Naskah masuk: 8 Apr 2022, direvisi: 10 Mei 2022, 3 Jun 2022, diterima: 6 Jun 2022)

Abstrak

Jenis *SMS spam* adalah jenis pesan teks yang tidak diinginkan atau tidak diminta yang dikirim ke ponsel pengguna, seringkali untuk tujuan komersial. Untuk mengatasi masalah *spam*, diperlukan teknik untuk memilah kata atau kalimat termasuk *spam* atau bukan *spam*. Pada penelitian ini diusulkan menggunakan *machine learning* untuk mengklasifikasikan pesan mana yang *spam* dan mana yang tidak *spam*. Data yang digunakan pada penelitian ini terdiri dari 1140 pesan, dimana sudah diberi label 0 untuk pesan yang tidak *spam* dan 1 untuk pesan yang *spam*. Algoritma yang digunakan untuk kasus ini adalah *Logistic Regression*. Hasil penelitian menunjukkan model memiliki tingkat akurasi untuk mengklasifikasi pesan, sebesar 97%. Aplikasi yang dikembangkan untuk menerapkan hasil pemodelan *machine learning* menggunakan bentuk sebuah *website* sederhana dengan bantuan *Flask framework* dari *Python*. Hasil akhir dari aplikasi ini adalah model *machine learning* yang dapat dibuka melalui *website*.

Kata Kunci: *SMS, Spam, Machine Learning, Logistic Regression, Flask, Web.*

Web-based Classifying SMS Application Using Logistic Regression Algorithm

Abstract

A type of SMS spam is an unwanted or unsolicited text message sent to a user's cell phone, often for commercial purposes. A technique is needed to sort words or sentences, including spam or not spam, to solve the spam problem. In this study, it is proposed to use machine learning to classify which messages are spam and which are not spam. The data used in this study consists of 1140 messages, which have been labelled 0 for messages that are not spam and 1 for messages that are spam. The algorithm used for this case is Logistic Regression. The results showed that the model has an accuracy rate for classifying messages of 97%. Applications developed to apply machine learning modelling results using the form of a simple website with the help of the Flask framework from Python. The result of this application is a machine learning model that can be accessed through the website.

Keywords: *SMS, Spam, Machine Learning, Logistic Regression, Flask, Web.*

I. PENDAHULUAN

Salah satu teknologi turunan dari pemanfaatan AI adalah *Machine Learning*. *Machine Learning* adalah sebuah pendekatan yang fokus penerapan aplikasinya untuk memprediksi [1]. *Machine Learning* juga banyak diterapkan di berbagai bidang, seperti di bidang kesehatan untuk mendeteksi pasien yang terkena diabetes, di bidang pendidikan untuk pengembangan *Automatic Assessment*.

Dalam penelitian ini, penulis akan berfokus pada penyelesaian masalah menggunakan *Machine Learning*. *Machine Learning* dipilih karena dapat menyelesaikan dengan baik suatu permasalahan yang memiliki banyak data sebagai informasinya. Kasus yang akan diteliti menggunakan *machine*

learning oleh penulis adalah adalah *SMS Spamming*. Singkatan *SMS* berasal dari kependekan dalam bahasa Inggris yaitu *Short Message Service*. *SMS* adalah salah satu metode komunikasi perangkat bergerak (*mobile*) yang paling umum [2]. *SMS spam* secara spesifik adalah jenis pesan teks yang tidak diinginkan atau tidak diminta yang dikirim ke ponsel pengguna, seringkali untuk tujuan komersial. Pesan tersebut dapat berupa pesan sederhana, tautan ke nomor untuk menelepon atau *SMS*, tautan ke situs web untuk informasi lebih lanjut, atau tautan ke situs web untuk mengunduh aplikasi [3]. Oleh karena itu, penulis melakukan penelitian untuk mengatasi masalah *SMS spam* ini.

Metode yang dapat digunakan untuk mengatasi *SMS spamming* adalah *machine learning*. *Machine learning* dapat mengekstraksi informasi dari teks dengan menggunakan berbagai tipe algoritma statistik dengan proses yang dikelompokan ke dalam *text mining*, dan *text analytics* [4]. Secara khusus, algoritma yang akan digunakan untuk mengatasi masalah *SMS Spamming* adalah *Logistic Regression. Logistic Regression* dapat menyelesaikan masalah *SMS Spam* karena algoritma ini sangat baik dalam melakukan klasifikasi untuk *binary classification*, yaitu mengklasifikasi label untuk nilai 1 dan 0. Pada kasus ini, permasalahan yang akan diteliti adalah memprediksi *SMS* mana yang *spam* (1) dan mana yang tidak *spam* (0).

Penulis nantinya akan mengimplementasikan model dari *machine learning* dengan algoritma *Logistic Regression* tersebut ke dalam bentuk sebuah aplikasi *website* sederhana. *Website* memiliki pengertian, yaitu fasilitas internet yang menghubungkan dokumen dalam lingkup lokal maupun jarak jauh yang menyimpan informasi tertentu seperti iklan, *link* bersponsor, *headers*, dan *footers* [5]. Pembuatan aplikasi *website* akan menggunakan bantuan *Flask*. *Flask* adalah sebuah kerangka kerja (*framework*) pada bahasa pemrograman *Python* [6].

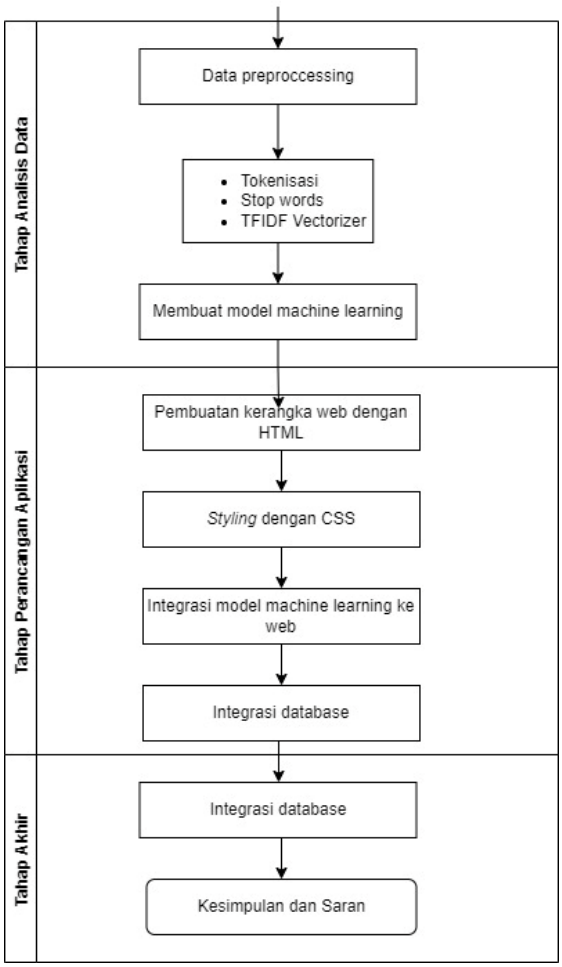
Aplikasi *website* dipilih sebagai media pengaplikasian *machine learning* karena mempermudah dalam melakukan pengecekan hasil *output* dari prediksi *machine learning*, hanya dengan memasukan pesan teks, lalu hasil prediksi akan muncul. Aplikasi web *machine learning* ini nantinya akan menerima hasil input berupa pesan *SMS*, lalu sistem akan memprediksi apakah inputan tersebut *spam* atau tidak *spam*.

Dari penelitian-penelitian yang pernah dilakukan sebelumnya mengenai *logistic regression*. Diantaranya adalah penelitian dari Muhamad Ichsan Gunawan, Dedy Sugiarto, dan Is Mardianto (2020) [7]. Perbedaan penelitian tersebut dengan penelitian peneliti adalah penelitian tersebut meningkatkan model milik peneliti lain yang sudah ada dengan dengan menggunakan *dataset* diabetes melitus. Sedangkan milik peneliti, *dataset* yang digunakan adalah *dataset SMS*. Penelitian lainnya yang pernah dilakukan adalah penelitian yang dilakukan oleh L. Wu, dan M. Li (2018) [8]. Penelitian ini memiliki pada *dataset* yang digunakan. Terdapat tiga *dataset* mengenai *customer churn*. Sedangkan pada penelitian yang akan dilakukan hanya menggunakan satu *dataset*, dan *dataset* yang digunkan adalah *dataset SMS*. Penelitian lainnya adalah penelitian dari Kemal Polat (2019) [9]. Penelitian membandingkan tingkat akurasi algoritma *logistic regression* dengan lima algoritma lainnya. Sedangkan penelitian yang akan dilakukan berfokus pada satu algoritma saja. Penelitian lainnya mengenai *logistic regression* adalah penelitian dari Liu Lei (2018) [10]. Perbedaan penelitian tersebut dengan penelitian yang akan dilakukan adalah kasus yang diteliti adalah diagnosis kanker payudara pada pasien. Sedangkan penelitian peneliti mengenai kasus *SMS spam*. Penelitian lainnya mengenai *logistic regression* adalah penelitian dari Montu Saw, Tarun Saxena, Sanjana Kaithwas, Rahul Yadav, dan Nidhi Lal (2020) [11]. Perbedaan penelitian ini adalah model evaluasi yang digunakan berupa *Accuracy*, *Recall*, *Precision*, dan *F1-Score*. Sedangkan, untuk penelitian yang akan dilakukan hanya menggunakan *Accuracy*.

Tujuan dari penelitian ini adalah untuk mengetahui tingkat nilai akurasi yang dihasilkan oleh algoritma *logistic regression* dalam mengklasifikasi *SMS spam*. Tujuan lainnya penelitian ini adalah membuat aplikasi berbasis *web* yang berintegrasi dengan model *machine learning* untuk mengklasifikasikan *SMS spam*.

II. METODOLOGI PENELITIAN

Dalam membantu penyusunan penelitian, maka diperlukan susunan kerangka kerja dalam melakukan pembangunan *website* yang berintegrasi dengan *machine learning*. Gambar 1 menampilkan tahapan penelitian dari penelitian ini.



Gambar 1. *Flowchart* Alur Penelitian

Gambar 1 menjelaskan bahwa tahapan awal penelitian dimulai dari menentukan topik penelitian, rumusan masalah, dan metode yang digunakan untuk menyelesaikan masalah. Tahapan penelitian dilanjutkan dengan menganalisis data yang telah diperoleh, lalu memproses data tersebut untuk digunakan dalam pembangunan model *machine learning*. Tahapan berikutnya adalah mengintegrasikan model tersebut ke dalam aplikasi *web* yang telah dirancang. Penelitian diakhiri dengan proses pengambilan simpulan dan saran.

Setelah semua alur penelitian dideskripsikan, diperlukan metode penyelesaian untuk melakukan tahapan penelitian tersebut. Metode penyelesaian diawali dengan membangun model *machine learning* terlebih dahulu, lalu merancang aplikasi *web* yang diusulkan. Berikut adalah metode-metode penyelesaian tersebut:

A. *Logistic Regression*

Logistic Regression adalah jenis regresi analisis yang digunakan untuk menjelaskan hubungan antara variabel dependen dan variabel independen menghubungkan satu atau lebih variabel bebas dengan variabel terikat jenis kategori; bisa 0 dan 1, benar atau salah, besar atau kecil. Tipe variabel bebasnya berupa kategori. Inilah yang membedakan regresi logistik dari regresi berganda atau regresi linear lainnya [12]. Pada beberapa kasus, variabel tidak bebas bersifat kualitatif dan dideskripsikan dalam satu atau dua kategori [9]. Persamaan *Logistic Regression* dinyatakan dalam Persamaan (1).

$$Ln\left(\frac{p}{1-p}\right) = B_o + B_1X \quad (1)$$

B_o = konstanta
B_1 = koefisien dari masing - masing variabel

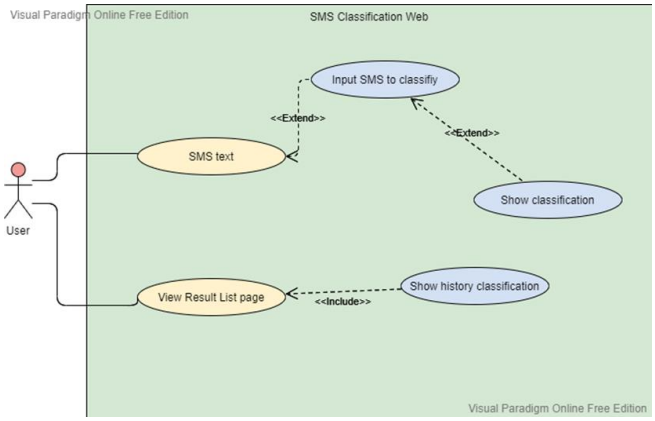
Nilai p atau peluang (Y = 1) dapat dicari dengan persamaan (2).

$$p = \frac{e^{(B_0 + B_1X)}}{(1 + e^{(B_0 + B_1X)})} \quad (2)$$

B. Rancangan Sistem Aplikasi

Tahapan dalam perancangan aplikasi *web* ini adalah dengan membuat *Use Case Diagram*, *Sequence/Activity* Diagram untuk alur kerja sistem, dan pembuatan *ERD* untuk perancangan basis data didalamnya.

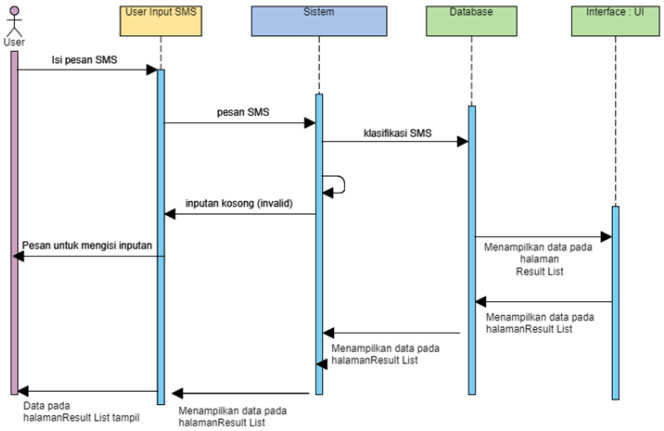
1. *Use Case Diagram*



Gambar 2. *Use Case Diagram* Aplikasi

Pada Gambar 2 menjelaskan bagaimana *user* pertama kali akan berinteraksi dengan menginput pesan *SMS* yang mana jika fitur ini dijalankan maka akan masuk ke fitur selanjutnya yaitu mengklasifikasi pesan *SMS*. Setelah pesan terklasifikasi maka akan muncul hasil klasifikasi pesan *SMS* tersebut. *User* juga dapat mengakses halaman *Result List* yang mana akan menampilkan hasil dari riwayat klasifikasi dari pesan yang diinputkan oleh *user*.

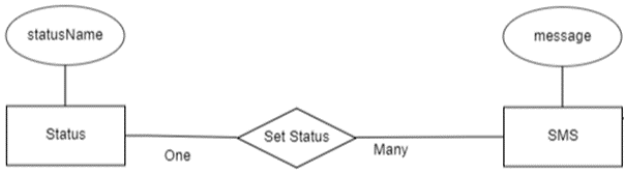
2. *Sequence/Activity Diagram*



Gambar 3. *Sequence / Activity Diagram* Aplikasi

Gambar 3 menjelaskan bahwa terdapat satu aktor (*user*) dan empat objek, yaitu *User Input SMS*, Sistem, *Database*, dan *Interface*: *UI*. Pertama-tama mahasiswa akan masuk ke tampilan *User Input SMS* pada halaman *Home*, lalu menginputkan input dengan menggunakan pesan *SMS*. Lalu, sistem akan mengirimkan data dan sistem akan melakukan klasifikasi pesan tersebut dengan menggunakan *machine learning* yang sudah diintegrasikan pada sistem. Jika data yang diinputkan kosong, maka sistem akan mengirimkan pesan peringatan pada *user* untuk mengisi inputan terlebih dahulu. Lalu, data hasil klasifikasi akan dikirimkan ke *database*. Langkah terakhir adalah *Interface* akan mengirimkan data hasil klasifikasi yang ada pada *database* untuk ditampilkan pada halaman *Result List.*

3. *ERD Diagram*



Gambar 4. *ERD Diagram* Aplikasi

Gambar 4 menampilkan dua entitas, yaitu Status, dan *SMS*. Status memiliki atribut *statusName*, dan *SMS* memiliki atribut *message*. Entitas Status melakukan relasi terhadap *SMS* dengan memberikan status pada *SMS* (*spam* atau tidak *spam*). Relasi yang dilakukan kedua entitas ini adalah *One-to-Many*, karena untuk setiap nama status (*spam* atau tidak *spam*) dapat berhubungan dengan lebih dari satu pesan teks.

C. *Dataset SMS*

Sumber data diperoleh dari hasil penelitian peneliti lain. Untuk kasus ini, peneliti memperoleh data untuk *dataset SMS* dari peneliti Rahmi, F. dan Wibisono, Y [13]. Jumlah keseluruhan *SMS* pada *dataset* adalah 1140 pesan. Tabel 1 menampilkan contoh dari isi *dataset SMS* yang digunakan.

Tabel 1. Contoh *Dataset SMS*

Teks	Label
Butuh Piknik?Ada nih liburan GRATIS ke Hongkong!Cara:Pake pulsa AXIS di HP Androidmu buat transaksi sebanyak2nya items Pokemon Go,BIGO,COC, dll!Info838 LD328	1
BONUS PULSA 50% dari Indosat Ooredoo, MAU?? Ketik *345# dan call dari HP kamu sebelum 28 April 2016	1
Aku jg blm berangkt wkwk	0
Sama ingetin ya, ini kan web nya mau di publish, kita kan ikut develop tuh, nah di web nya ada nama kita atau jurusan ga?	0

Keterangan pada label:
1 = *spam*
0 = bukan *spam*

D. Tahap Pembangunan Model

1. *Import* Modul

Modul-modul yang penulis gunakan pada pembangunan model *machine learning* ini adalah *pandas*, *numpy*, *nltk*, *jcopml*, dan *scikit-learn*.

```
import numpy as np
import pandas as pd
from nltk.tokenize import word_tokenize

from nltk.corpus import stopwords
from string import punctuation

from sklearn.model_selection import
train_test_split

from sklearn.pipeline import Pipeline

from sklearn.compose import
ColumnTransformer

from sklearn.linear_model import
LogisticRegression

from sklearn.model_selection import
RandomizedSearchCV

from jcopml.pipeline import num_pipe,
cat_pipe

from jcopml.tuning import
random_search_params as rsp
```

Modul *pandas* dan *numpy* penulis gunakan untuk memamanipulasi data. Untuk modul *nltk* digunkaan untuk mengekstraksi dan memproses teks. Modul *sklearn* digunakan untuk melakukan *data preprocessing*, sedangkan modul *jcopml* digunakan untuk proses pembuatan *pipeline* pada tahapan *data preprocessing*.

2. *Dataset Splitting*

Tahapan selanjutnya yang penulis lakukan setelah *import* modul dan *dataset*. *Dataset* dibagi menjadi data latihan dan data ujian dengan proporsi pembagiannya adalah 40%.

```
X = df.Teks
y = df.label

X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.4,
stratify=y, random_state=42)
```

Masing-masing kolom (teks dan label) dibagi dalam dua variabel yaitu x dan y. Kedua variabel tersebut akan dibagi lagi menjadi *Data Training* dan *Data Testing*.

3. *Data Preprocessing*

Pada tahapan ini, untuk mempermudah dalam melakukan *tunning* pada *model learning* untuk mendapatkan skor tertinggi, penulis membungkus *TfidfVectorizer* bersamaan dengan algoritma yang penulis pakai yaitu *Logistic Regression*. Metode *tunning* yang digunakan penulis adalah *RandomizedSearch*. *RandomizedSearch* lebih cepat dibandingkan *GridSearch* karena pada *RandomizedSearch* hanya akan melakukan pencarian kombinasi parameter tertentu saja.

III. HASIL DAN PEMBAHASAN

A. Hasil Pengujian

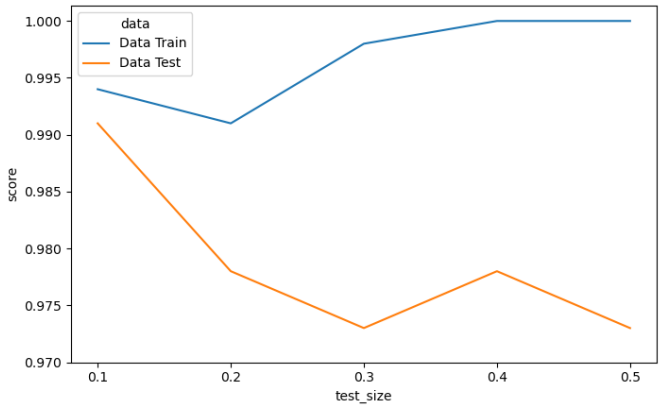
1. *Test Size*

Pada pengujian ini, penulis memberikan nilai 0,1; 0,2; 0,3; 0,4 dan 0,5 untuk *test size* pada *dataset splitting*. Pada algoritma *logistic regression*, penulis memberi nilai tetap pada *solver* yaitu *lbfgs*. Berikut adalah hasil pengujiannya:

Tabel 2. Hasil Pengujian *Test Size*

Test Size	Data	Nilai Akurasi
0,1	*Data Train*	0,994
0,1	*Data Test*	0,991
0,2	*Data Train*	0,991
0,2	*Data Test*	0,978
0,3	*Data Train*	0,998
0,3	*Data Test*	0,973
0,4	*Data Train*	1
0,4	*Data Test*	0,978
0,5	*Data Train*	1
0,5	*Data Test*	0,973

Tabel 2 menampilkan nilai akurasi dari *data training* dan *data testing* untuk setiap *test size* yang berbeda. *Test size* yang memiliki nilai akurasi paling tinggi pada model adalah *test size* 0,4.



Gambar 5. Grafik Nilai *Test Size*

Gambar 5 menampilkan grafik nilai akurasi dari *Data Train* cenderung lebih stabil untuk setiap *test size* berbeda, sedangkan nilai akurasi *Data Test* mengalami penurunan untuk beberapa *test size* berbeda.

2. *Solver*

Pada pengujian ini, penulis memberikan *input* parameter yang berbeda pada *solver Logistic Regression*, yaitu *newton-cg*, *sag*, *saga*, *lbfgs*, dan *liblinear*. *Test size* yang digunakan untuk *dataset splitting* menggunakan hasil eksperimen *test size* sebelumnya yaitu 0.4. Berikut adalah hasil eksperimen pada *solver*:

Tabel 3. Hasil Pengujian *Solver*

Solver	Data	Nilai Akurasi
newton-cg	*Data Train*	1
newton-cg	*Data Test*	0,975
lbfgs	*Data Train*	1
lbfgs	*Data Test*	0,978
sag	*Data Train*	1
sag	*Data Test*	0,975
saga	*Data Train*	1
saga	*Data Test*	0,975
liblinear	*Data Train*	1
liblinear	*Data Test*	0,978

Tabel 3 menampilkan nilai akurasi untuk masing-masing *solver* pada *data training* dan *data testing*. Nilai akurasi yang dihasilkan tidak memiliki perbedaan yang berbeda satu sama lain antar *solver*. Disini nilai akurasi tertinggi *data training* dan *data testing* didapat pada *solver lbfgs*, dan *liblinear*.

B. Analisis Pengujian

1. Ekstraksi Teks menggunakan *TF-IDF*

Langkah awal sebelum pembangunan model adalah dengan melakukan pembobotan pada data teks menggunakan *TF-IDF*. *TF-IDF* adalah algoritma yang umum digunakan untuk seleksi fitur yang berupa teks [14]. *TF* memiliki arti sebagai seberapa banyak sebuah kata muncul dalam suatu dokumen, sedangkan *IDF* seberapa jarang suatu kata muncul dalam dokumen [15]. Berikut adalah formulanya untuk *TF* dan *IDF*:

$$TF(t,d) = \frac{f_{t,d}}{(jumlah\ total\ kata\ dalam\ dokumen)} \quad (3)$$

$f_{t,d}$ = frekuensi kemunculan kata pada dokumen

$$IDF(t) = log\frac{1+n}{1+df(t)} + 1 \quad (4)$$

n = Jumlah dokumen
$df(t)$ = Jumlah dokumen yang mengandung kata tertentu

Pada penelitian ini menggunakan bantuan *TfidfVectorizer* pada *Jupyter Notebook* untuk mengimplementasikan *TFIDF* pada *dataset* yang ada. Hasil dari ekstraksi data teks adalah sebagai berikut ada pada Gambar 6.

```
(0, 2313)     0.6219833648003625
(0, 1429)     0.659510277335598
(0, 3861)     0.42211714961648017
(1, 4249)     0.27066478750901896
(1, 2277)     0.1286935232695319
(1, 1942)     0.12245265280946672
(1, 1138)     0.2552636419070825
(1, 1099)     0.27066478750901896
(1, 4100)     0.23586049314903626
(1, 1786)     0.12914842569733312
(1, 2473)     0.21800792289650264
(1, 2255)     0.27066478750901896
(1, 3067)     0.15156649276580716
(1, 2880)     0.22893520960082436
(1, 3712)     0.12692651170708147
(1, 2050)     0.18135038788204824
(1, 1149)     0.22307996579024286
(1, 3856)     0.1941309152408416
(1, 3956)     0.20591180862585595
(1, 3554)     0.18720563169262974
(1, 1135)     0.2026067772945662
(1, 929)      0.185151947835686 11
(1, 2280)     0.4461599315804857
(2, 2863)     0.34979602517473424
(2, 3383)     0.6995920503494685
:     :
```

Gambar 6. Representasi Contoh Hasil Perhitungan

Gambar 6 menjelaskan bahwa untuk kolom pertama adalah indeks dari kalimat, untuk kolom kedua adalah indeks *stop words* dari kalimat tersebut, sedangkan kolom ketiga mendeklarasikan bobot nilai *IDF*. Nilai pada kolom ketiga ini yang akan menentukan sebuah pesan itu *spam* (1) atau tidak (0). Jika sebuah kalimat yang mengandung kata-kata yang memiliki nilai bobot *IDF* mendekati 1, maka pesan tersebut dilabeli *spam* (1), begitu juga sebaliknya.

Setelah dilakukan tahapan *preprocessing* terhadap teks, yaitu dengan membuat semua teks berhuruf kecil dan juga memenggal kalimat yang diperlukan. Tabel 4 adalah hasil teks *preprocessing* tersebut.

Tabel 4. Hasil *Preprocessing*

Teks	Label
4.5gb/30 hari hanya rp 55 ribu spesial buat anda yang terpilih.,aktifkan sekarang juga di *550*907# buruan..!,skb	1
anda akan mengaktifkan paket bbm gratis berlaku 30 hari dari pertama kali aktivasi.,ketik flash (spasi) ya untuk melanjutkan	1
anda sedang menikmati paket reguler dgn sisa kuota 209715 kb.dapatkan beragam paket internet hemat tri di *123*3#	1
anyonghaseyo!,ngaku kpopers sejati??,ayo lengkapi koleksi video kpopmu di http://kpop.xl.co.id.,skrg harga video lebih murah loh!,info*123*2200#	1

2. Implementasi Algoritma *Logistic Regression*

Setelah melakukan ekstraksi teks untuk *dataset*, selanjutnya mengimplementasikan *logistic regression* dengan rumus yang sudah dicantumkan pada halaman sebelumnya. Pada Gambar 7 menampilkan hasil prediksi *SMS* dari proses kalkulasi yang dilakukan di *Jupyter Notebook*.

```
PRED: 1 - SMS : INFO RESMI Tri Care SELAMAT Nomor Anda terpilih mendapatkan Hadiah 1 Unit MOBIL Dri Tri Care Dengan PIN Pemenan
g: br2Sh99 info: www.gebeyar-3care.tk

PRED: 0 - SMS : Ceritanya biasa aja. Tp tetep bikin nangis

PRED: 1 - SMS : Aktifkan iRing Coboy Jr - Terhebat. Tekan *808*7#. Info: 100&111 ada hits terbaru dari NOAH - Jika Engkau. Akti
fkan iRing nya di HP kamu. ketik MG NOAH02 kirim ke 808 Info: 100&111 Berkah iRing Rp 1000 dr Yuni Shara - Akhirnya. Aktifkan i
Ring nya, Tkn *808*1*2*3# lalu Ok/Call. Raih THR PuluhanJt!. Berhenti: Unreg ke 808

PRED: 0 - SMS : Assalamualaikum akang teteh, jangan lupa ya hari ini ada carrier day jam 10:30, exhibition fpmipa b :) (love) H
ayuu teh ditungguuu

PRED: 0 - SMS : Rasa mau sidang bulan depaaan mel. Jahaaat pisaaan
```

Gambar 7. Hasil Uji Prediksi

C. Evaluasi Performa Model

Scoring yang digunakan untuk mengukur performa model *machine learning* yang digunakan adalah *accuracy*. *Accuracy* adalah metrik untuk mengevaluasi model klasifikasi. *Accuracy* merupakan perhitungan pecahan dari jumlah prediksi yang benar pada model dengan total prediksi yang ada. *Accuracy* dapat digunakan untuk *dataset* yang memiliki *class* seimbang [16].

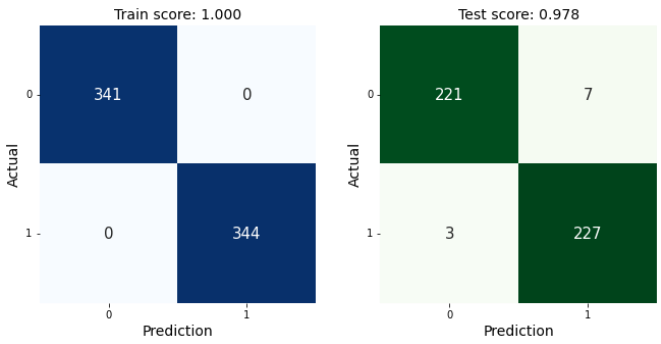
$$Accuracy = \frac{Jumlah\ prediksi\ benar}{Jumlah\ semua\ prediksi} \quad (5)$$

Untuk klasifikasi biner, *accuracy* bisa dihitung dalam hal positif dan negatif, seperti berikut ini:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (6)$$

Keterangan:
TP = *True Positive*
TN = *True Negative*
FP = *False Positive*
FN = *False Negative*

Berikut adalah hasil evaluasi model menggunakan *Accuracy* yang divisualisasikan menggunakn *Confusion Matrix*:

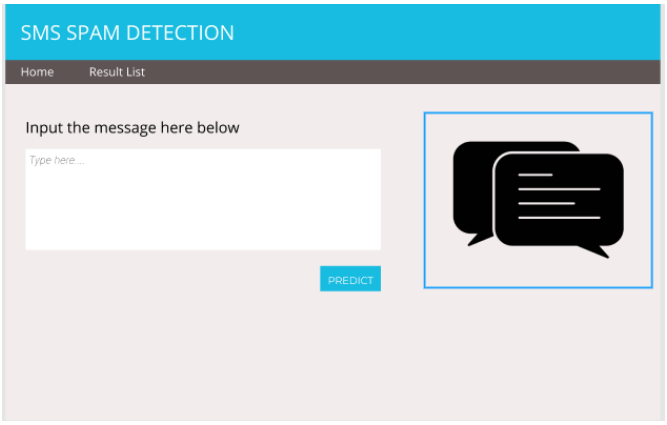


Gambar 8. *Confusion Matrix*

Gambar 8 menampilkan *confusion matrix* untuk *testing score* didapati terdapat 221 pesan yang berlabel tidak *spam*, dan mesin dengan benar menebak tidak *spam*. Berikutnya, terdapat 227 pesan berlabel *spam*, dan mesin dengan benar menebak *spam*. Terdapat beberapa prediksi salah yang dilakukan mesin, seperti terdapat tujuh pesan berlabel tidak *spam* namun mesin menebak tidak *spam*.

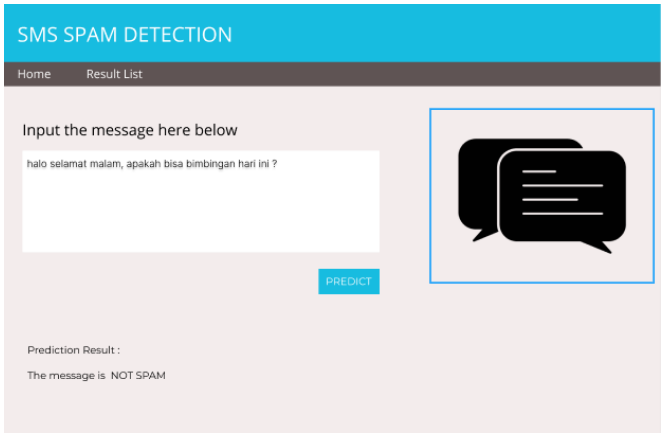
D. Tampilan *Website*

1. Halaman Utama



Gambar 9. Halaman Utama *Website*

Gambar 9 menampilkan halaman utama *website* yang memiliki dua menu pada *navigation bar* yaitu menu *Home* dan *Result List*. Pada menu *Home* menampilkan fitur utama yaitu *input* pesan pada *textarea* dan juga ada tombol *PREDICT*. Nantinya ketika *user* klik tombol tersebut, akan muncul hasil prediksi apakah pesan yang diinputkan adalah *spam* atau tidak *spam*.

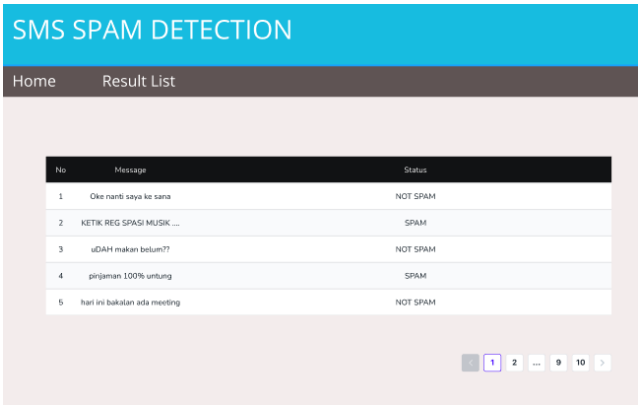


Gambar 10. Halaman Utama *Website* dengan *Prediction Result*

Gambar 10 menampilkan halaman utama *website* saat *user* klik tombol *submit* untuk memprediksi inputan pesan teks. Dibagian bawah tombol, akan tampil keterangan berupa hasil klasifikasi pesan tersebut.

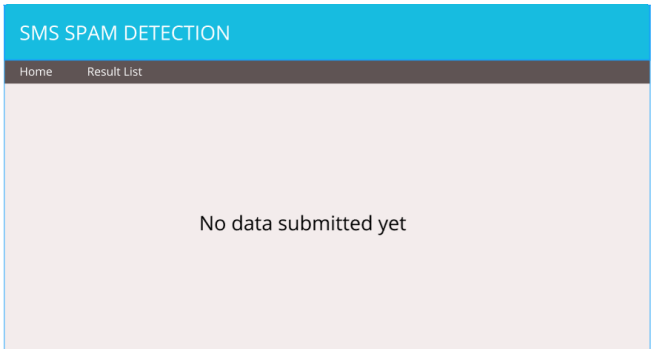
2. Halaman *Result List*

Gambar 11 menampilkan halaman *Result List* sebelum ada pesan yang diprediksi oleh sistem.



Gambar 11. Halaman *Result List*

Gambar 11 menampilkan halaman *Result List* yang memuat hasil riwayat inputan pesan dari *user* beserta dengan hasil dari prediksinya yang ditempatkan pada kolom status. Data tersebut ditampilkan dalam tabel.



Gambar 12. Halaman *Result* Sebelum Ada Pesan

Gambar 12 menampilkan halaman *Result List* dalam kondisi belum ada pesan yang diprediksi. Halaman akan menampilkan pesan diatas. Halaman ini nantinya akan menampilkan tabel jika ada data pesan yang sudah diprediksi.

IV. KESIMPULAN

Berdasarkan dari hasil penelitian yang telah dilakukan mengenai klasifikasi *SMS Spam* menggunakan *machine learning*, berikut adalah beberapa kesimpulannya:

1) Cara *machine learning* mendeteksi pesan mana yang *spam* dan yang tidak *spam* adalah dengan menganalisa pola dari sms tersebut, apakah pola tersebut mirip dengan pola *SMS* yang ada di *dataset* yang sebelumnya telah diberi label 0 (*spam*) dan 1 (tidak *spam*). Pola tersebut dihasilkan dari hasil perhitungan pembobotan kata menggunakan *TFIDF*.
2) Nilai akurasi model *machine learning* yang didapat dari hasil pengujian *test size* dan *solver* pada pembahasan sebelumnya menggunakan algoritma *Logistic Regression* dengan *solver lbfgs* adalah 1 untuk *data training*, dan 0,975 untuk *data testing*.
3) Nilai parameter *solver* yang berbeda untuk *test size* 0,4, secara signifikan tidak berpengaruh dalam meningkatkan nilai akurasi model.

4) Model *machine learning* yang penulis buat, penulis implementasikan pada aplikasi *website* menggunakan *framework* dari *Python*, yaitu *Flask*.

REFERENSI

[1] S. Kushwaha, S. Bahl, A. K. Bagha, K. S. Parmar, M. Javaid, A. Haleem, and R. P. Singh, "Significant applications of machine learning for COVID-19 pandemic," *Journal of Industrial Integration and Management*, vol. 05, no. 04, pp. 453–479, 2020.

[2] "What is SMS and how does it work?," *Android Authority*, 30-Aug-2021. [Online]. Available: https://www.androidauthority.com/what-is-sms-280988/. [Accessed: 27-Aug-2021].

[3] "SMS SPAM: Security against SMS spam," *T*. [Online]. Available: https://www.t-mobile.com/privacy-center/education-and-resources/sms-spam. [Accessed: 25-Dec-2021].

[4] C. C. Aggarwal, "Machine learning for text," 2018.

[5] A. Dilip Patel and V. N. Pandya, "Web page classification based on context to the content extraction of articles," *2017 2nd International Conference for Convergence in Technology (I2CT)*, 2017.

[6] A. A. Mohammed, R. Basa, A. K. Kuchuru, S. P. Nandigama, and M. Gangolla, "Random Forest Machine Learning technique to predict Heart disease ," *European Journal of Molecular & Clinical Medicine*, vol. 7, no. 4, pp. 2453–2459 , 2020.

[7] M. I. Gunawan, D. Sugiarto, and I. Mardianto, "Peningkatan Kinerja Akurasi prediksi penyakit diabetes mellitus menggunakan metode grid Seacrh Pada algoritma logistic regression," *Jurnal Edukasi dan Penelitian Informatika (JEPIN)*, vol. 6, no. 3, p. 280, 2020.

[8] L. Wu and M. Li, "Applying the CG-logistic regression method to predict the customer churn problem," *2018 5th International Conference on Industrial Economics System and Industrial Security Engineering (IEIS)*, 2018.

[9] K. Polat, "Freezing of gait (fog) detection using logistic regression in parkinson's disease from acceleration signals," *2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT)*, 2019.

[10] L. Liu, "Research on logistic regression algorithm of breast cancer diagnose data by machine learning," *2018 International Conference on Robots & Intelligent System (ICRIS)*, 2018.

[11] M. Saw, T. Saxena, S. Kaithwas, R. Yadav, and N. Lal, "Estimation of prediction for getting heart disease using logistic regression model of machine learning," *2020 International Conference on Computer Communication and Informatics (ICCCI)*, 2020.

[12] M. R. Romadhon and F. Kurniawan, "A comparison of naive bayes methods, logistic regression and KNN for predicting healing of covid-19 patients in Indonesia," *2021 3rd East Indonesia Conference on Computer and Information Technology (EIConCIT)*, 2021.

[13] F. Rahmi, "APLIKASI SMS SPAM FILTERING PADA ANDROID MENGGUNAKAN ALGORITMA NAÏVE BAYES," thesis, 2017.

[14] Z. Zhu, J. Liang, D. Li, H. Yu, and G. Liu, "Hot topic detection based on a refined TF-IDF algorithm," *IEEE Access*, vol. 7, pp. 26996–27007, 2019.

[15] U. Bhattacharjee, P. K. Srijith, and M. S. Desarkar, "Term specific TF-IDF boosting for detection of rumours in social networks," *2019 11th International Conference on Communication Systems & Networks (COMSNETS)*, 2019.

[16] "Classification: Accuracy | machine learning crash course | google developers," *Google*. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy. [Accessed: 01-Dec-2021].