INSTITUT
TEKNOLOGI
HARAPAN
BANGSA

Veritas vos liberabit

PENANGANAN DATASET *MULTI CLASS* TIDAK SEIMBANG DENGAN MODIFIKASI SMOTE DAN KLASIFIKASI MENGGUNAKAN *SUPPORT VECTOR MACHINE* PADA LAPORAN TIKET INSIDEN LAYANAN IT

TUGAS AKHIR

Agape Arimatea
1117008

PROGRAM STUDI INFORMATIKA
INSTITUT TEKNOLOGI HARAPAN BANGSA
BANDUNG
2020

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang

Seiring perkembangan teknologi yang begitu cepat di dunia ini, penggunaan teknologi juga semakin meningkat. Meningkatnya penggunaan teknologi oleh masyarakat luas ini berkontribusi atas meningkatnya jumlah data yang tersebar luas di internet. Seiring perkembangan TI, lingkup penerapan sistem berbasis TI turut meluas sehingga volume insiden TI meningkat drastis. Hal ini memberikan kesempatan untuk para peneliti untuk menggunakan data tersebut untuk mendapatkan keuntungan atau berkontribusi untuk perkembangan teknologi. Akan tetapi, ada beberapa permasalahan dengan data yang diambil, seperti data dengan rasio keseimbangan yang memiliki perbedaan secara ekstrim, yang merupakan hal yang sering ditemui dari data yang tersebar [1]. Data yang tidak seimbang (*Class Imbalance*) merupakan situasi ketika jumlah data dari kelas minoritas (*positive class*) jauh dibawah atau jauh lebih kecil dibandingkan dengan jumlah data dari kelas mayoritas (*negative class*). Contoh kasus seperti pada data dari sebuah perusahaan, dimana data laporan insiden kecil akan jauh lebih banyak dibandingkan dengan laporan insiden yang dapat mempengaruhi perusahaan secara besar.

Machine Learning dibuat untuk mengurangi error dan meningkatkan akurasi, akan tetapi distribusi jumlah data antar kelas tidak diperhitungkan [2]. *Machine Learning* akan menghasilkan prediksi yang lebih akurat terhadap kelas yang memiliki jumlah data yang lebih besar, dan sebaliknya. Sehingga, *class imbalance* sangat mempengaruhi akurasi dari prediksi yang dibuat oleh *Machine Learning*. Oleh karena itu, *class imbalance* perlu diatasi supaya hasil yang didapatkan lebih akurat. Ada dua pendekatan yang dapat dilakukan untuk mengatasi *class imbalance*, metode melalui *algorithm-level* dan metode melalui *data-level* [3]. Pendekatan *algorithm-level* lebih melibatkan modifikasi dalam algoritma seperti *ensemble learning* dan *cost-sensitive learning* untuk mengatasi data tidak seimbang. Sementara itu, pendekatan *data-level* berfokus pada *preprocessing* terhadap data yang tidak seimbang. Metode pendekatan *data-level* yang biasanya digunakan adalah metode *sampling*.

Metode *sampling* adalah teknik yang digunakan untuk memilih, memanipulasi dan

menganalisa sebuah *subset* representatif dari titik-titik data untuk mengidentifikasi pola dan *trend* dari sebuah dataset yang besar. Metode *sampling* yang sering digunakan dalam penelitian adalah metode SMOTE [4]. Metode SMOTE adalah teknik statistik yang berguna untuk menambahkan jumlah data atau kasus dalam *dataset* secara seimbang. *Instance* baru dibuat dari data kelas minoritas yang dimasukkan. *Instance* baru ini bukan hasil sembarang replikasi, akan tetapi algoritma dari metode ini akan mengambil sampel dari fitur atau pola dari setiap kelas target dan tetangga terdekatnya untuk menghasilkan data baru dengan fitur atau pola yang serupa. SMOTE mengambil seluruh kumpulan data sebagai masukan, akan tetapi hanya meningkatkan persentase kelas minoritas [5]. Meskipun demikian, SMOTE dapat menimbulkan masalah generalisasi yang berlebihan ketika area kelas minoritas dianggap area kelas mayoritas dikarenakan pembuatan *instance* yang salah [9].

Penelitian oleh Zhu et al. melakukan modifikasi tentang SMOTE dengan menggunakan k-NN sehingga menghasilkan algoritma baru bernama *k-NN-based Synthetic Minority Oversampling to combat Multiclass imbalance problems* (SMOM) [10]. Pada SMOM, tiap *neighbor direction*, akan diberikan *weight* untuk mrepresentasikan seberapa besar kemungkinan *neighbor* itu digunakan untuk membuat *instance* baru. Dengan demikian, SMOM dapat menghindari generalisasi yang berlebihan.

Selain itu, ada juga penelitian yang diadakan oleh Hartono et al menggunakan penggabungan dari Biased Support Vector Machine (BSVM) dengan Weighted-SMOTE. BSVM akan memberikan *sensitivity* yang lebih baik dibandingkan dengan SVM biasa. BSVM bekerja dengan cara menyediakan *cost functions* yang berbeda pada kelas minoritas dan kelas mayoritas, lalu masing-masing akan dikelompokkan menjadi *Support Vector Sets* dan *Non Support Vector Sets*.

Pada penelitian yang dilakukan oleh Hensman dan Masko ditemukan bahwa distribusi dari data latih memiliki dampak yang signifikan terhadap hasil dari *Neural Network*[13]. Adanya *class imbalance* dapat mempengaruhi *classifier Neural Network* dalam melakukan konvergensi untuk mendapatkan hasil yang baik[14].

Pada penelitian ini akan digunakan SMOTE pada proses preprocessing training dan *classifier Support Vector Machine* dan *Neural Network* untuk menghasilkan nilai *recall* yang lebih baik sehingga menghasilkan model yang lebih baik.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas penulis merumuskan masalah sebagai berikut:

1. Bagaimana peningkatan akurasi pada *classifier SVM* dan *Neural Network* dengan menggunakan metode SMOTE yang sudah dimodifikasi?
2. Bagaimana pengaruh metode SMOTE yang sudah dimodifikasi pada nilai *recall* pada *classifier SVM* dan *Neural Network*?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah di atas, tujuan penelitian Tugas Akhir ini adalah sebagai berikut:

1. Meningkatkan akurasi pada *classifier* dengan menggunakan metode SMOTE yang sudah dimodifikasi.
2. Meningkatkan hasil prediksi dari dataset *multi class* pada laporan tiket insiden.

1.4 Batasan Masalah

Dalam penelitian ini, peneliti akan membatasi masalah yang akan diteliti antara lain:

1. Dataset yang digunakan merupakan dataset berbahasa Inggris dengan jumlah dataset tidak melebihi 50000 baris.
2. *Classifier* yang digunakan adalah *Support Vector Machine* dan *Neural Network*.

1.5 Kontribusi Penelitian

Kontribusi yang diberikan dari penelitian ini adalah:

1. Melakukan analisis pengaruh teknik-teknik *oversampling* terhadap akurasi dari klasifikasi kelas tiket insiden.

1.6 Metode Penelitian

Metode penelitian yang dilakukan dalam penelitian ini adalah sebagai berikut:

1. Studi Literatur

   Penulisan ini dimulai dengan studi kepustakaan yaitu mengumpulkan bahan-bahan referensi baik dari buku, artikel, *paper*, jurnal, makalah mengenai *class imbalance*.
2. Data sampling

   Data sampling yang akan digunakan berupa data bersifat tulisan yang akan

diambil dari penyedia data terbuka di internet mengenai *class imbalance* pada tiket insiden.

3. Analisis Masalah
   Pada tahap ini dilakukan analisis permasalahan yang ada, batasan yang dimiliki dan kebutuhan yang diperlukan.
4. Perancangan dan Implementasi Algoritme
   Pada tahap ini dilakukan perancangan pada algoritme yang akan dipakai untuk menyelesaikan masalah berdasarkan metode yang telah dipilih.
5. Pengujian
   Pada tahap ini dilakukan pengujian terhadap aplikasi yang telah dibangun.
6. Dokumentasi
   Pada tahap ini dilakukan pendokumentasian hasil analisis dan implementasi secara tertulis dalam bentuk laporan skripsi.

1.7 Sistematika Pembahasan

Pada penelitian ini peneliti menyusun berdasarkan sistematika penulisan sebagai berikut:

BAB I PENDAHULUAN

Pendahuluan yang berisi latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, kontribusi penelitian, serta metode penelitian.

BAB II LANDASAN TEORI

Landasan Teori yang berisi penjelasan dasar teori yang mendukung penelitian ini.

BAB III ANALISIS DAN PERANCANGAN

Analisis dan Perancangan yang berisi analisis berupa algoritme yang digunakan.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Implementasi dan Pengujian yang berisi implementasi pengujian dengan berbagai data testing beserta hasilnya.

BAB V KESIMPULAN DAN SARAN

Penutup yang berisi kesimpulan dari penelitian dan saran untuk

penelitian lebih lanjut di masa mendatang.

BAB II

LANDASAN TEORI

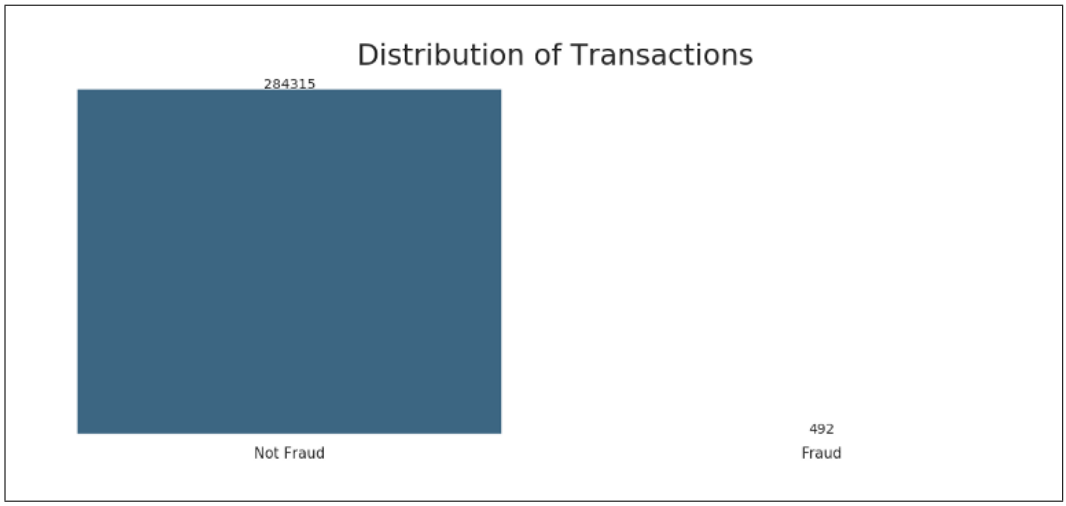
Bab ini menjelaskan teori-teori yang berkaitan mengenai teori penunjang dan jurnal terkait yang digunakan dalam proses penelitian tugas akhir ini.

2.1 Tinjauan Pustaka

Penelitian ini menggunakan beberapa teori terkait yang diperlukan dalam pengerjaan yang dilakukan. Penjelasan mengenai teori-teori tersebut akan dijelaskan sebagai berikut.

2.1.1 Class Imbalance

Class imbalance merupakan kondisi dimana jumlah data dari tiap kelas tidak konstan atau proporsi tiap kelas tidak sama [6]. Apabila terdapat dua kelas, maka data yang seimbang akan memiliki rasio sekitar 50% antar kelasnya. Untuk sebagian besar teknik *machine learning*, rasio ketidakseimbangan dalam jumlah kecil bukan merupakan masalah. Tetapi apa bila rasionya berat terlalu timpang sebelah, contoh 9:1, maka hasil yang didapatkan dari data tersebut menjadi tidak efektif dan memerlukan modifikasi.



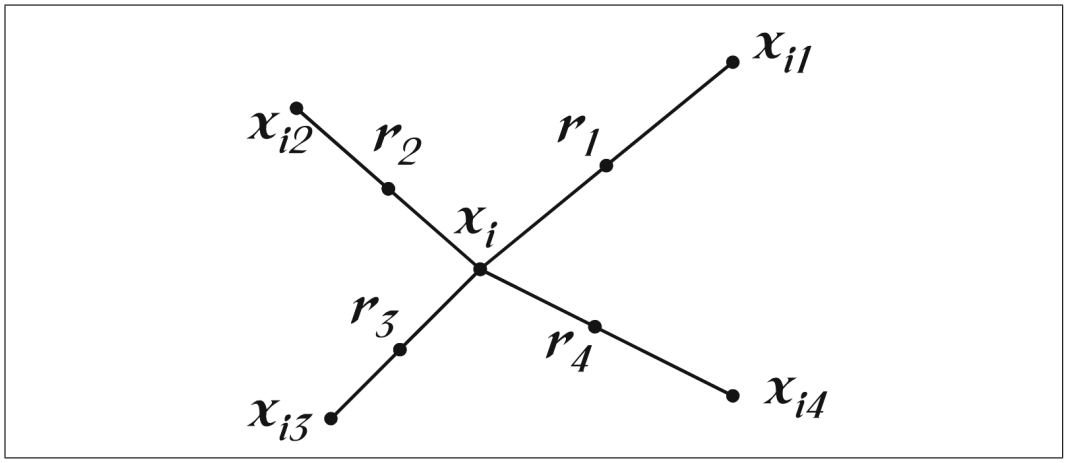
Gambar 2.1 Ilustrasi mengenai *class imbalance*

Dalam mengatasi *class imbalance*, dapat dilakukan pendekatan metode sampling. Metode *sampling* dapat dibagi menjadi dua, yaitu *undersampling* dan *oversampling*. Teknik *undersampling* akan membuang sebagian jumlah dari kelas mayoritas, Teknik *undersampling* digunakan ketika jumlah data yang dimiliki

cukup banyak. Sementara itu, Teknik *oversampling* akan menduplikasi sebagian jumlah dari kelas minoritas dan biasanya digunakan ketika jumlah data yang dimiliki tidak terlalu besar [8][9]. Teknik *oversampling* lebih sering digunakan karena teknik *undersampling* dapat mengurangi atau menghilangkan sebagian data yang penting. Metode *undersampling* biasanya digunakan apabila jumlah data yang dikumpulkan melebihi jumlah data yang ideal, sehingga proses pembelajaran dari *machine learning* dapat lebih efektif [9].

2.1.2 *Synthetic Minority Over-Sampling* (SMOTE)

Algoritme *Synthetic Minority Over-Sampling* (SMOTE) merupakan pendekatan *oversampling* untuk menyeimbangkan data latih. Daripada melakukan penggandaan pada *minority class instances*, ide utama dari SMOTE adalah melakukan *synthetic sampling* pada *minority class instances*. *Instance* yang terbentuk ini terbentuk dari interpolasi antara beberapa *positive instance* yang posisinya berdekatan. Prosedur SMOTE difokuskan untuk *feature space* daripadi *data space*. [12]



Gambar 2.2 Ilustrasi bagaimana *synthetic data points* dibuat oleh algoritme SMOTE

Contoh sederhana dari proses *oversampling* diilustrasikan pada Gambar 2.1. Dimana X_i *positive instance* dipilih sebagai basis untuk membuat *data points* sintetis yang baru. Berdasarkan *distance metric*, beberapa NN dari kelas yang sama (titik X_i1 sampai titik X_i4) dipilih dari data latih. Pada akhirnya, dilakukan interpolasi acak untuk mendapatkan *instance* baru dari r_1 sampai r_4. *r*merupakan nilai sintetis baru yang terbentuk dari proses replikasi yang ada.

Prosedur SMOTE pada dasarnya bekerja sebagai berikut. Pertama, jumlah total *oversampling* N (nilai integer) diatur, yang biasanya didefinisikan untuk

mendapatkan perkiraan distribusi kelas 1:1. Kemudian, proses iterasi dilakukan, terdiri dari beberapa langkah. Pertama, *instance* kelas positif dipilih secara acak dari data latih. Selanjutnya, kNN-nya (5 secara *default*) didapatkan. Pada akhirnya, N dari contoh K ini akan dipilih secara acak untuk menghitung *instance* baru dengan interpolasi. Untuk melakukannya, perbedaan antara vektor fitur (sampel) yang dipertimbangkan dan masing-masing tetangga diambil. Perbedaan ini dikalikan dengan angka acak yang ditarik antara 0 dan 1, kemudian ditambahkan ke vektor fitur sebelumnya. Ini menyebabkan pemilihan secara acak di titik sepanjang "*line segment*" antara fitur. Dalam kasus atribut nominal, salah satu dari dua nilai dipilih secara acak. Seluruh proses tadi dirangkum dalam Algoritma 3.

Algorithm 3 SMOTE algorithm

```
1: function SMOTE(T, N, k)
   Input: T; N; k                ▷ #minority class examples, Amount of oversampling, #NNs
   Output: (N/100) * T synthetic minority class samples
   Variables: Sample[][]: array for original minority class samples;
   newindex: keeps a count of number of synthetic samples generated, initialized to 0;
   Synthetic[][]: array for synthetic samples
2:     if N < 100 then
3:         Randomize the T minority class samples
4:         T = (N/100)*T
5:         N = 100
6:     end if
7:     N = (int)N/100   ▷ The amount of SMOTE is assumed to be in integral multiples of 100.
8:     for i = 1 to T do
9:         Compute KNN for i, and save the indices in the nnarray
10:        POPULATE(N, i, nnarray)
11:    end for
12: end function
```

Algorithm 4 Function to generate synthetic samples

```
1: function POPULATE(N, i, nnarray)
   Input: N; i; nnarray          ▷ #instances to create, original sample index, array of NNs
   Output: N new synthetic samples in Synthetic array
2:     while N ≠ 0 do
3:         nn = random(1,k)
4:         for attr = 1 to numattrs do          ▷ numattrs = Number of attributes
5:             Compute: dif = Sample[nnarray[nn]][attr] − Sample[i][attr]
6:             Compute: gap = random(0, 1)
7:             Synthetic[newindex][attr] = Sample[i][attr] + gap · dif
8:         end for
9:         newindex++
10:        N−−
11:    end while
12: end function
```

Gambar 2.3 Algoritme SMOTE

$$(f_1', f_2') = (f_{11}, f_{12}) + random(0-1) * (f_{21} - f_{11}, f_{22} - f_{11}) \quad (2.1)$$

Di mana :
f_1' : Titik *data point* sumbu X sintetis baru.
f_2' : Titik *data point* sumbu Y sintetis baru.
f_{11} : Titik *data point* sumbu X data awal.
f_{12} : Titik *data point* sumbu Y data awal.
f_{21} : Titik *data point* sumbu X data tetangga terdekat.
f_{22} : Titik *data point* sumbu Y data tetangga terdekat.

2.1.3 *Term Frequency-Inverse Document Frequency*

Term Frequency-Inverse Document Frequency(TF-IDF) merupakan salah satu metode pembobotan yang umum digunakan dalam proses kategorisasi teks. Dalam metode ini akan dihitung nilai *Term Frequency*(TF) dan *Inverse Document Frequency* pada setiap kata atau token didalam sebuah dokumen. Nilai bobot kata / *term* menentukan tingkat kepentingan kata pada data. Pada TF-IDF, semakin besar frekuensi kemunculan kata pada sebuah dokumen maka bobotnya akan semakin tinggi, tetapi jika kata tersebut sering muncul pada dokumen kata lain maka bobot dari kata tersebut akan berkurang.

Inverse Document Frequency dapat dihitung menggunakan rumus berikut.

$$IDF_t = log(N/df) \quad (2.2)$$

Di mana :
IDF_t : Hitungan metrik dari banyaknya dokumen yang mengandung kata tersebut.
N : Total dokumen yang ada.
df : Total kemunculan kata pada dokumen-dokumen.

Term Frequency-Inverse Document Frequency dapat dihitung menggunakan rumus berikut.

$$W_{dt} = tf_{dt} * IDF_t \quad (2.3)$$

Di mana :
W_{df} : Bobot dari kata tersebut.
tf_{dt} : Total kemunculan kata pada .
df : Total kemunculan kata pada dokumen-dokumen.

2.1.4 *Artificial Neural Network*

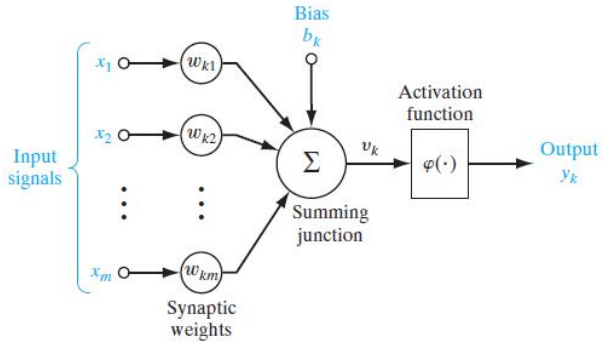
Artificial Neural Network, atau biasa disebut dengan *neural network*, merupakan metode yang terinspirasi dari sistem saraf dan otak manusia dan diimplementasikan pada komputer digital. Otak bersifat sangat kompleks, non linear, dan dapat menghitung secara paralel, sehingga dapat mengatur cara kerja dari struktur sistem saraf seperti *neuron* yang dapat melakukan perhitungan tertentu dengan waktu yang jauh lebih cepat daripada waktu perhitungan komputer digital [11]. Pada definisi paling umum, *neural network* merupakan mesin yang dibentuk untuk membuat model yang sama dengan pada otak dalam melakukan suatu aktivitas [11]. Metode pembelajaran pada *neural network* umumnya tergolong atas *supervised* dan *unsupervised*. Bila nilai keluaran telah diketahui maka termasuk metode *supervised* karena *neural network* akan berusaha untuk mendapatkan nilai tersebut, digunakan untuk kasus *classification* dan *regression*. Sementara metode *unsupervised* tanpa menggunakan nilai keluaran sebagai tujuan, digunakan untuk kasus *clustering* dan *association*.

Neuron merupakan unit untuk memproses informasi yang paling dasar dalam *neural network*. Secara umum, terdapat tiga buah elemen dasar dalam *neuron*:

1. Sekumpulan sinapsis yang masing-masing memiliki bobot atau *weight*. Sinyal masukan x_j pada sinapsis j yang terhubung dengan neuron k akan dikalikan dengan bobot sinapsis w_{kj}.

2. Sebuah proses penjumlahan untuk menjumlahkan sinyal masukan yang telah diproses dengan bobotnya pada masing-masing sinapsis neuron, operasi ini disebut *linear combiner*.

3. Fungsi aktivasi untuk membatasi rentang dari keluaran *neuron*. Biasanya, jarak output dari neuron dibuat antara 0 hingga 1 atau -1 hingga 1.



Gambar 2.4 *Neuron* [11]

Gambar 2.4 terlihat bentuk model *neuron* dan dapat dilengkapi dengan beberapa persamaan dasar berikut (persamaan 2 . 4, 2 . 5, dan 2 . 6):

$$u_k = \sum_{j=1}^{m} w_{kj} x j \tag{2 . 4}$$

$$v_k = u_k + b_k \tag{2 . 5}$$

$$y_k = \varphi(v_k) \tag{2 . 6}$$

Di mana :
u_k : hasil *linear combiner* dari sinyal masukan.
x : sinyal masukan.
w_k : bobot sinapsis pada neuron k.
m : jumlah masukan.
b_k : nilai bias untuk menyesuaikan efek *affine transformation* pada keluaran u_k.
v_k : *induced local field*.
y_k : sinyal keluaran dari neuron.
φ : fungsi aktivasi.

2.1.5 *Support Vector Machine*

Support Vector Machine (SVM) merupakan metode *supervised learning* yang bersifat biner dalam TC [7] untuk memetakan vektor masukan ke dalam sebuah ruang dimensi tinggi dan menghasilkan luaran berupa sebuah *hyperplane* [8]. Dengan data latihan, algoritme tersebut menghasilkan hyperplane yang optimal yang membagi data. Terdapat beberapa pendekatan untuk menyelesaikan masalah berkelas banyak, antara lain *one-against-all* dan *one-against-one*. *One-against-all* membangun classifier biner sebanyak jumlah kelas. Setiap classifier membedakan kelas tertentu dari kelas lainnya. Kelas yang diprediksi didapatkan berdasarkan

luaran classifier tertinggi. Sementara itu, one-against-one mencakup n kelas dan membuat $\frac{n(n-1)}{2}$ classifier, yang berarti pembuatan satu classifier untuk sepasang kelas. Luaran yang diprediksi didapatkan berdasarkan pemilihan dari classifier [9]. Dengan demikian, SVM dinilai sangat akurat serta cepat dilatih dan dievaluasi [10].

2.2 Pustaka

Berikut adalah penjelasan dari *library* yang digunakan di dalam penelitian.

2.2.1 Pandas

Library yang digunakan untuk mengambil data adalah *Pandas*. *Library* ini merupakan *library open-source* yang umum digunakan untuk mengambil data *spreadsheet*.

Tabel 2.1 Tabel *Library* Pandas

No	***Function***	**Deskripsi**
1	pandas.read_csv(*filepath, encoding*)	*Function* ini digunakan untuk membaca *file* dengan ekstensi .csv.

2.2.2 Numpy

Numpy merupakan sebuah *library open-source* yang biasa digunakan untuk melakukan komputasi ilmiah di dalam *Python*.

Tabel 2.2 Tabel *Library* Numpy

No	***Function***	**Deskripsi**
1	numpy.unique(*column_header*)	*Function* ini digunakan untuk mencari nilai yang unik dari sebuah kolom data.
2	numpy.mean(*array*)	*Function* ini digunakan untuk menghitung nilai rata-rata pada *array* atau *axis* tertentu.

2.2.3 Pickle

Pickle merupakan sebuah *library* yang digunakan untuk melakukan serialisasi dan de-serialisasi sebuah objek Python sehingga objek Python dapat disimpan pada *disk*.

Tabel 2.3 Tabel *Library* Pickle

No	***Function***	**Deskripsi**
1	pickle.dump(*object*, open(*filepath*,*mode*))	*Function* ini digunakan untuk menyimpan objek atau model Python yang sudah dibuat kedalam bentuk file.

Tabel 2.3 Tabel Pickle (Lanjutan)

No	*Function*	Deskripsi
2	pickle.load(open(*filepath*, *mode*))	*Function* ini digunakan untuk memasukkan objek Python yang berbentuk file agar bisa digunakan dalam program.

2.2.4 *Natural Language Toolkit*(NLTK)

NLTK merupakan sebuah *library* yang terdiri dari kumpulan *library* yang biasanya digunakan dalam pemrosesan teks, seperti *stemming* dan *tokenization*.

2.2.4.1 *SnowballStemmer*

SnowballStemmer merupakan salah satu *library* yang digunakan untuk melakukan *stemming* yang ada didalam *library* NLTK.

Tabel 2.4 Tabel *Library* SnowballStemmer

No	*Function*	Deskripsi
1	SnowballStemmer(*language*, ignore_stopword=**boolean**)	*Function* ini digunakan untuk melakukan *stemming* dan juga bisa melakukan *stopword removal* pada data yang dimasukkan.

2.2.5 Scikit-learn

Scikit-learn merupakan *library machine learning* yang bersifat *open source* yang dapat melakukan baik *supervised* maupun *unsupervised learning*. Di dalam scikit-learn terdapat banyak *library-library* lain yang berfungsi untuk *model fitting*, *data preprocessing*, *model selection and evaluation* dan kegunaan lainnya.

2.2.5.1 CountVectorizer

Library ini digunakan untuk membuat matriks yang terdiri dari token yang dibentuk dari kumpulan dokumen teks.

Tabel 2.5 Tabel *Library* CountVectorizer

No	*Function*	Deskripsi
1	CountVectorizer(). fit_transform	*Function* ini digunakan membentuk data menjadi *document-term matrix*.

2.2.5.2 TfidfTransformer

Library ini digunakan untuk membuat *count matrix* menjadi matriks TF-IDF yang telah dinormalisasi.

Tabel 2.6 Tabel *Library* TfidfTransformer

No	*Function*	Deskripsi
1	TfidfTransformer(). fit_transform	*Function* ini digunakan mempelajari *IDF vector* lalu me-*transform* data tersebut.

2.2.5.3 *Train Test Split*

Library ini digunakan untuk membagi *dataset* menjadi data latih dan data *testing*.

Tabel 2.7 Tabel *Library* train_test_split

No	*Function*	Deskripsi
1	train_test_split(data, label, test_size=**integer**)	*Function* ini digunakan mempbagi data latih dan data *testing* dengan rasio data *testing* sebesar parameter test_size.

2.2.5.4 *Support Vector Classifier*(SVC)

Tabel 2.8 Tabel *Library* SVC

No	*Function*	Deskripsi
1	SVC(kernel=”linear”)	*Function* ini digunakan membuat model SVM dengan kernel linear dengan menggunakan data latih yang diberikan.

2.2.6 *Pipeline*

Library ini digunakan untuk menjalankan rangkaian proses yang dimasukkan kedalam paramternya.

Tabel 2.9 Tabel *Library* Pipeline

No	*Function*	Deskripsi
1	Pipeline([sequence-1, .., sequence-N])	*Function* ini digunakan menjalankan rangkaian proses yang ada di dalamnya.
2	Pipeline.fit()	*Function* ini digunakan untuk membuat model yang telah dihasilkan dari rangkaian proses sebelumnya.

2.3 Tinjauan Studi

Pada bagian ini akan dijelaskan mengenai perbandingan dari berbagai penelitian yang terkait dengan penanganan *imbalanced class* pada tiket insiden.

2.3.1 *State of the Art*

Terdapat beberapa metode lain yang memiliki ruang lingkup yang mirip dengan penelitian ini khususnya mengenai deteksi dan pelacakan manusia. Tabel 2.10 *State of the Art* akan menjelaskan perbedaan-perbedaan metode yang telah dipelajari oleh penulis dari jurnal.

Tabel 2.10 *State of the Art*

Penelitian	**Metode**	**Hasil**
Zhu., et al.;"Synthetic Minority oversampling technique for multiclass imbalance problems". Pattern Recognition (2017)	*k-NN-based synthetic minority oversampling algorithm*(SMOM)	Sebelum dilakukan data sampling oleh SMOM nilai recall yang dihasilkan sebesar 54%. Lalu, setelah dilakukan data sampling terjadi peningkatan terhadap nilai recall menjadi 66.9%.
Yap, et al. ; "Handling Imbalanced Dataset Using SVM and k-NN Approach". AIP Conference Proceedings 1750, 020023 (2016)	1. SVM. 2. k-NN. 3. *Logistic Regression*.	Pada penelitian ini didapatkan nilai awal akurasi 96.2% dan *sensitivity* 21.9%. Lalu hasil yang didapatkan setelah dilakukan *sampling data* adalah akurasi 79.4% dan *sensitivity* 74.8%.
Hartono., et al.; "Biased support vector machine and weighted-SMOTE in handling class imbalance problem". International Journal of Advances in Intelligent Informatics Vol. 4, No. 1, March 2018, pp. 21-27 (2018)	1. *Biased Support Vector Machine*. 2. *Weighted-SMOTE Algorithm*.	Dengan menggunakan algoritma yang baru, nilai akurasi yang didapatkan sebesar 88% dan sensitivity / recall sebesar 87%. Dengan rata-rata rasio keseimbangan data sebelum diproses = 63.2 :35.9 dan rata-rata rasio keseimbangan data setelah diproses = 50.9 : 48.2

2.4 Tinjauan Objek

Pada bagian ini akan dijelaskan objek yang terkait dengan pengangan *imbalanced class* pada tiket insiden.

2.4.1 Insiden IT

Berdasarkan pengertian dari Kamus Besar Bahasa Indonesia, insiden adalah peristiwa (khususnya yang kurang penting dalam hubungannya dengan peristiwa lainnya yang lebih besar). Insiden dapat diartikan sebagai sesuatu yang rusak atau sesuatu yang tidak bekerja dengan semestinya.

Kasus insiden yang dapat terjadi diperusahaan ada yang ringan seperti untuk mengganti kredensial akun atau juga hal yang lebih berat seperti putusnya sebuah layanan dari *website* perusahaan.

2.4.2 Tiket Insiden IT

Jika sebuah gangguan pada sistem terdeteksi, baik oleh sistem, maupun oleh pengguna sistem, sebagai sebuah insiden. Dilakukan identifikasi insiden yang dilaporkan melalui berbagai jalur. Lalu, insiden tersebut dicatat sebagai sebuah tiket insiden. Jadi, tiket insiden adalah sebuah sarana untuk pelaporan terjadinya suatu insiden.

service unavailable	sent monday accessible hi we unable access thanks
for us not working	hello advise there some issues with open for users form th floor sent re for working hell
not responding	sent wednesday down hello down we cannot work can you please have look thanks soft
performance issues and	sent thursday july re incident warning for hi sorry seems there was some interruption co
connection down th floor	connection was down th floor for few seconds
inaccessible	sent license server problem dear looks like there issue with license server you registere
urgent cannot open files on	sent wednesday urgent cannot open files importance high hello when try file get below
wireless network down	sent thursday wireless issue we currently having outage sent
not accessible	sent tuesday accessible dear accessible today could you please investigate made test tha
certificate expired	multiple users have reported certificate issues
oracle errors when logging	error hello trying connect but working could you please investigate issue thank you best
oracle system is down	down
survey taking errors app outage for us accounts	sent thursday march status investigating survey taking errors app outage for accounts sur

Gambar 2.5 Tiket Insiden

BAB III

ANALISIS DATA

3.1 Analisis Masalah

Pada bab 1 telah dijelaskan bahwa *class imbalance* merupakan sebuah masalah yang cukup penting bagi dunia *machine learning*. Pada penelitian ini, penulis akan membuat suatu program yang bertujuan untuk melakukan klasifikasi kelas prioritas dari sebuah dataset tiket insiden pada perusahaan IT.

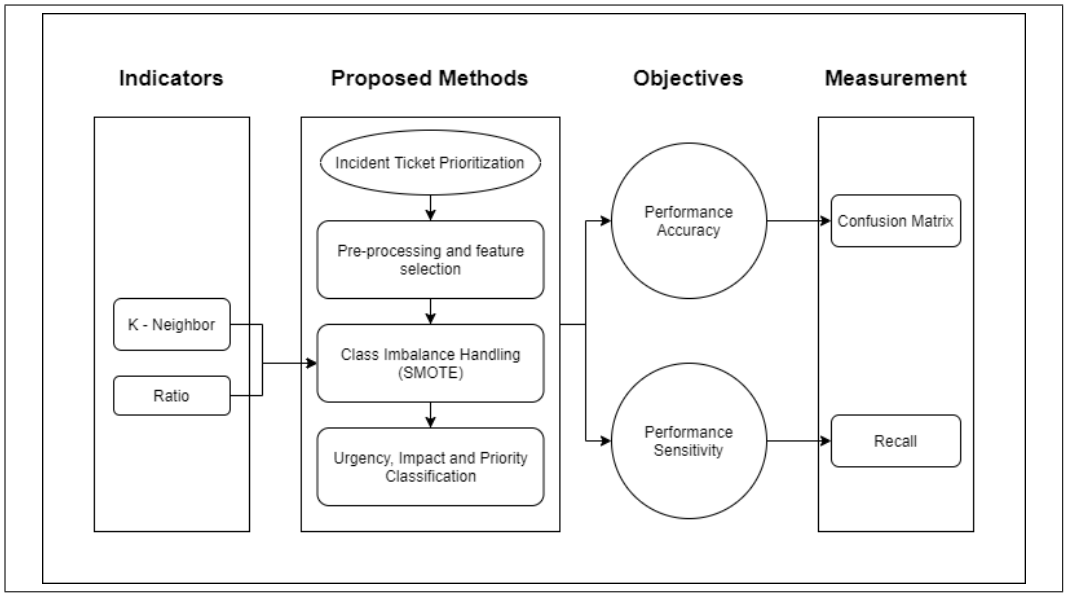
Dataset yang digunakan dalam penelitian ini diambil dari sebuah layanan di perusahaan IT. Data yang diambil ini merupakan data historis yang telah diambil dari tahun 2019 hingga sekarang. Dataset ini terdiri dari 5 atribut berikut *Summary*, *Description*, *Impact*, *Urgency* dan *Priority*.

Data *input* pada penelitian ini merupakan data teks yang berisi kumpulan tiket-tiket insiden yang terjadi pada sebuah layanan IT sebuah perusahaan IT. Data *input* ini digunakan untuk menguji metode yang diimplementasikan dalam pra-pemrosesan teks dan digunakan untuk menunjukkan kemampuan sistem jika digunakan pada skenario penggunaan nyata.

Data teks yang digunakan sebagai data *input* memiliki 2 buah atribut yang terdiri dari *Summary* dan *Description*. Teks-teks yang dimasukkan sebagai data *input* akan sangat mempengaruhi hasil dari akurasi sistem ini. Dari data *input* tersebut akan dihasilkan sebuah *output* yang beratribut *Priority*.

3.2 Kerangka Pemikiran

Berikut ini adalah kerangka pemikiran dari metode yang diusulkan untuk mengatasi *class imbalance*.



Gambar 3.1 Kerangka Pemikiran

Pada gambar 3.1, tahap pertama yang akan dilakukan adalah melakukan *preprocessing* dan *feature selection* menggunakan TF-IDF pada data *incident ticket prioritization*. Setelah melalui tahap itu, maka akan dilakukan *resampling* pada data latih menggunakan metode *Synthetic Minority Oversampling Technique*(SMOTE).

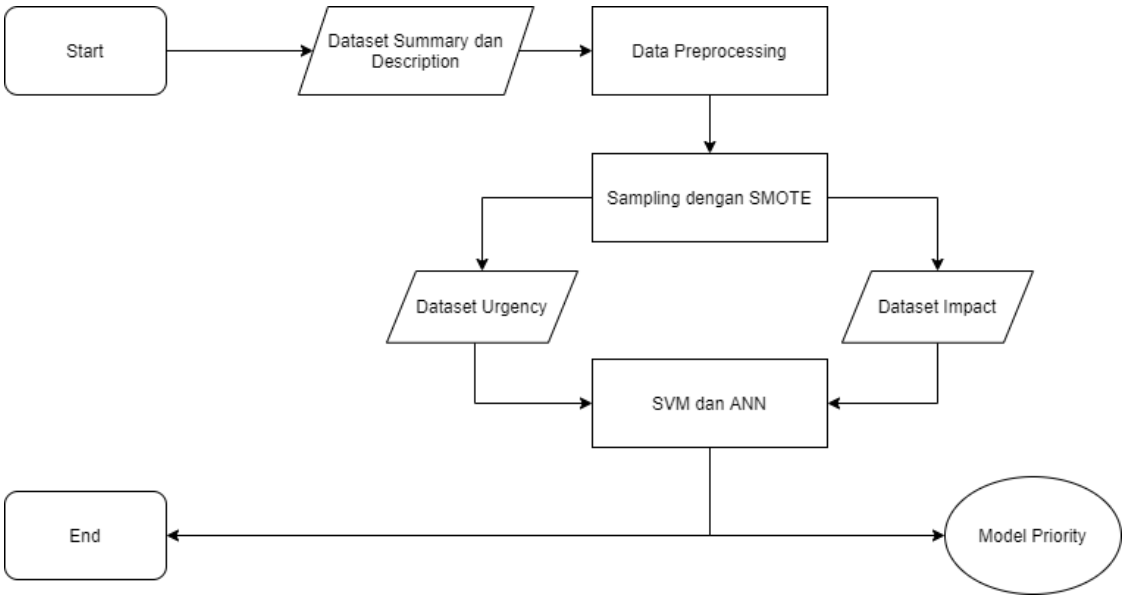
Pada tahap ini terdapat 2 variabel indikator yang dapat mempengaruhi kerja dari *Synthetic Minority Oversampling Technique* (SMOTE) yaitu nilai *k-neighbor* dan *ratio*. Nilai *k-neighbor* disini dipilih untuk menentukan jumlah *nearest neighbor* yang akan diambil atau digunakan dalam algoritma SMOTE. Sementara nilai *ratio* merupakan nilai rasio *oversampling* yang diinginkan pada data kelas minoritas.

Pada tahap klasifikasi, *classifier* SVM dan ANN akan digunakan dalam melakukan klafikasi untuk data *urgency, impact* dan *priority*.

Penelitian untuk bertujuan untuk melihat hasil akurasi dari hasil klasifikasi SVM dan ANN dan mengukur *sensitivity* yang dimiliki oleh model yang dibuat dengan menggunakan nilai *recall*.

3.3 Analisis Urutan Proses Global

Penelitian ini akan menggunakan *Synthetic Minority Oversampling* untuk mengatasi *imbalanced class*. Setelah dilakukan pelatihan, diharapkan arsitektur yang dibangun dapat melakukan estimasi dengan tepat. Berikut adalah urutan proses global.



Gambar 3.2 Flowchart Global

Berikut ini adalah uraian dari *flowchart* pada Gambar 3.2 yang dilakukan dalam penelitian ini:

1. Data latih dimasukkan ke dalam runtutan *preprocessing*.
2. Dilakukan *resampling* pada data latih menggunakan algoritma SMOTE.
3. Kemudian data latih dipisah menjadi data dengan label *urgency* dan *impact*.
4. Hasil dari data latih *urgency* dan *impact* akan disatukan kembali menjadi nilai akhir yaitu nilai *priority*.

3.3.1 Data Belajar

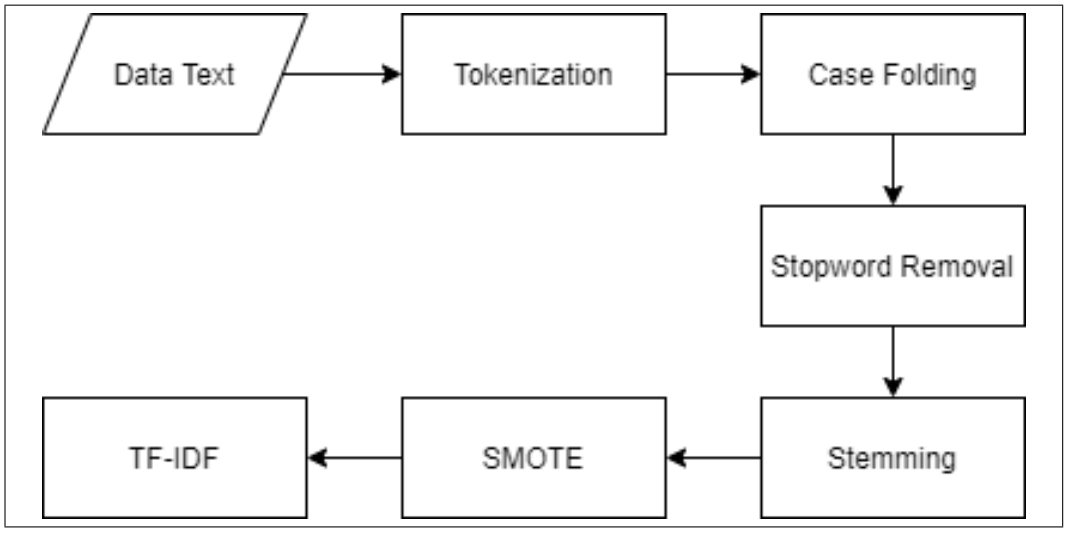
Data belajar pada penelitian ini didapatkan dengan membagi keseluruhan data menjadi 80% untuk data belajar dan 20% data *test*. Data ini tersusun dari 5 atribut yang terdiri dari *Summary*, *Description*, *Impact*, *Urgency* dan *Priority*. Data belajar ini akan digunakan untuk melakukan penanganan data tidak seimbang yang nantinya dapat meningkatkan akurasi dari hasil klasifikasi *Priority* dari masukan yang ada.

3.4 Analisis Kasus

Pada bagian ini dilakukan analisis proses secara bertahap dengan menggunakan perhitungan manual.

3.4.1 *Preprocessing*

Tahapan dimana teks dipersiapkan menjadi data untuk diolah ke tahapan-tahapan selanjutnya. Pada penelitian ini ada beberapa tahapan preprocessing yang dilakukan yaitu proses *tokenization* , proses *case folding*, proses *stopword removal*, proses *stemming*, *data resampling* dengan SMOTE dan pembobotan TF-IDF.



Gambar 3.3 Flowchart Preprocessing

3.4.1.1 *Tokenization*

Didalam tahap ini setiap kata yang terpisah dengan spasi akan dipecah menjadi kata-kata tunggal. Contoh dokumen dapat dilihat pada tabel 3.1.

Tabel 3.1 Contoh *Tokenization*

Sebelum	**Sesudah**

Tabel 3.1 Contoh *Tokenization*

Sebelum	**Sesudah**
hello advise there some issues with open for users form th floor sent re for working hello wireless se thanks sent wednesday for working hello currently we have issue with connecting client for can you please raise ticket if caused by or something side regards	['hello', 'advise', 'there', 'some', 'issues', 'with', 'open', 'for', 'users', 'form', 'th', 'floor', 'sent', 're', 'for', 'working', 'hello', 'wireless', 'se', 'thanks', 'sent', 'wednesday', 'for', 'working', 'hello', 'currently', 'we', 'have', 'issue', 'with', 'connecting', 'client', 'for', 'can', 'you', 'please', 'raise', 'ticket', 'if', 'caused', 'by', 'or', 'something', 'side', 'regards']
sent wednesday down hello down we cannot work can you please have look thanks software design lead en	['sent', 'wednesday', 'down', 'hello', 'down', 'we', 'cannot', 'work', 'can', 'you', 'please', 'have', 'look', 'thanks', 'software', 'design', 'lead', 'en']
sent thursday server down importance high hi guys servers db web responding please restore functionality asap there ukraine working with among others billing for inactivity regards	['sent', 'thursday', 'server', 'down', 'importance', 'high', 'hi', 'guys', 'servers', 'db', 'web', 'responding', 'please', 'restore', 'functionality', 'asap', 'there', 'ukraine', 'working', 'with', 'among', 'others', 'billing', 'for', 'inactivity', 'regards']
stuck hi teams noticed stuck responsive order fix going restart application fix issue there could be some down could you log ticket sent notification users ad ll keep you updated thank you senior engineer	['stuck', 'hi', 'teams', 'noticed', 'stuck', 'responsive', 'order', 'fix', 'going', 'restart', 'application', 'fix', 'issue', 'there', 'could', 'be', 'some', 'down', 'could', 'you', 'log', 'ticket', 'sent', 'notification', 'users', 'ad', 'll', 'keep', 'you', 'updated', 'thank', 'you', 'senior', 'engineer']
down between hi we cannot access any servers between seems be down can you please have look urgent because blocking entire text thanks design lead	['down', 'between', 'hi', 'we', 'cannot', 'access', 'any', 'servers', 'between', 'seems', 'be', 'down', 'can', 'you', 'please', 'have', 'look', 'urgent', 'because', 'blocking', 'entire', 'text', 'thanks', 'design', 'lead']

3.4.1.2 *Case Folding*

Di tahap ini, semua huruf yang ada akan diubah kedalam huruf kecil, angka dan tanda baca juga akan dihapus. Tujuan dari *case folding* adalah menyeragamkan kata-kata yang ada. Contoh dokumen dapat dilihat pada tabel 3.2

Tabel 3.2 Contoh *Case Folding*

Sebelum	**Sesudah**
hello advise there some issues with open for users form th floor sent re for working hello wireless se thanks sent wednesday for working hello currently we have issue with connecting client for can you please raise ticket if caused by or something side regards	['hello', 'advise', 'there', 'some', 'issues', 'with', 'open', 'for', 'users', 'form', 'th', 'floor', 'sent', 're', 'for', 'working', 'hello', 'wireless', 'se', 'thanks', 'sent', 'wednesday', 'for', 'working', 'hello', 'currently', 'we', 'have', 'issue', 'with', 'connecting', 'client', 'for', 'can', 'you', 'please', 'raise', 'ticket', 'if', 'caused', 'by', 'or', 'something', 'side', 'regards']
sent wednesday down hello down we cannot work can you please have look thanks software design lead en	['sent', 'wednesday', 'down', 'hello', 'down', 'we', 'cannot', 'work', 'can', 'you', 'please', 'have', 'look', 'thanks', 'software', 'design', 'lead', 'en']
sent thursday server down importance high hi guys servers db web responding please restore functionality asap there ukraine working with among others billing for inactivity regards	['sent', 'thursday', 'server', 'down', 'importance', 'high', 'hi', 'guys', 'servers', 'db', 'web', 'responding', 'please', 'restore', 'functionality', 'asap', 'there', 'ukraine', 'working', 'with', 'among', 'others', 'billing', 'for', 'inactivity', 'regards']
stuck hi teams noticed stuck responsive order fix going restart application fix issue there could be some down could you log ticket sent notification users ad ll keep you updated thank you senior engineer	['stuck', 'hi', 'teams', 'noticed', 'stuck', 'responsive', 'order', 'fix', 'going', 'restart', 'application', 'fix', 'issue', 'there', 'could', 'be', 'some', 'down', 'could', 'you', 'log', 'ticket', 'sent', 'notification', 'users', 'ad', 'll', 'keep', 'you', 'updated', 'thank', 'you', 'senior', 'engineer']

Tabel 3.2 Contoh *Case Folding*

Sebelum	**Sesudah**
down between hi we cannot access any servers between seems be down can you please have look urgent because blocking entire text thanks design lead	['down', 'between', 'hi', 'we', 'cannot', 'access', 'any', 'servers', 'between', 'seems', 'be', 'down', 'can', 'you', 'please', 'have', 'look', 'urgent', 'because', 'blocking', 'entire', 'text', 'thanks', 'design', 'lead']

3.4.1.3 *Stopword Removal*

Pada tahap ini, kata-kata yang tidak terkait akan dibuang. Kata-kata tersebut sudah ada didalam kamus *stopword*, kata-kata yang biasa dibuang adalah kata sambung, kata depan, kata ganti, kata penghubung. Contoh dokumen dapat dilihat pada tabel 3.3

Tabel 3.3 Contoh *Stopword Removal*

Sebelum	**Sesudah**
hello advise there some issues with open for users form th floor sent re for working hello wireless se thanks sent wednesday for working hello currently we have issue with connecting client for can you please raise ticket if caused by or something side regards	['hello', 'advise', 'there', 'some', 'issues', 'with', 'open', 'for', 'users', 'form', 'th', 'floor', 'sent', 're', 'for', 'working', 'hello', 'wireless', 'se', 'thanks', 'sent', 'wednesday', 'for', 'working', 'hello', 'currently', 'we', 'have', 'issue', 'with', 'connecting', 'client', 'for', 'can', 'you', 'please', 'raise', 'ticket', 'if', 'caused', 'by', 'or', 'something', 'side', 'regards']
sent wednesday down hello down we cannot work can you please have look thanks software design lead en	['sent', 'wednesday', 'down', 'hello', 'down', 'we', 'cannot', 'work', 'can', 'you', 'please', 'have', 'look', 'thanks', 'software', 'design', 'lead', 'en']

Tabel 3.3 Contoh *Stopword Removal*

Sebelum	**Sesudah**
sent thursday server down importance high hi guys servers db web responding please restore functionality asap there ukraine working with among others billing for inactivity regards	['sent', 'thursday', 'server', 'down', 'importance', 'high', 'hi', 'guys', 'servers', 'db', 'web', 'responding', 'please', 'restore', 'functionality', 'asap', 'there', 'ukraine', 'working', 'with', 'among', 'others', 'billing', 'for', 'inactivity', 'regards']
stuck hi teams noticed stuck responsive order fix going restart application fix issue there could be some down could you log ticket sent notification users ad ll keep you updated thank you senior engineer	['stuck', 'hi', 'teams', 'noticed', 'stuck', 'responsive', 'order', 'fix', 'going', 'restart', 'application', 'fix', 'issue', 'there', 'could', 'be', 'some', 'down', 'could', 'you', 'log', 'ticket', 'sent', 'notification', 'users', 'ad', 'll', 'keep', 'you', 'updated', 'thank', 'you', 'senior', 'engineer']
down between hi we cannot access any servers between seems be down can you please have look urgent because blocking entire text thanks design lead	['down', 'between', 'hi', 'we', 'cannot', 'access', 'any', 'servers', 'between', 'seems', 'be', 'down', 'can', 'you', 'please', 'have', 'look', 'urgent', 'because', 'blocking', 'entire', 'text', 'thanks', 'design', 'lead']

3.4.1.4 *Stemming*

Pada tahap ini kata-kata yang ada akan diubah menjadi bentuk kata dasar. Algoritma *stemming* yang digunakan adalah algoritma Snowball yang tersedia pada library NLTK. Contoh dokumen dapat dilihat pada tabel 3.4.

Tabel 3.4 Contoh *Stemming*

Sebelum	**Sesudah**

Tabel 3.4 Contoh *Stemming*

Sebelum	**Sesudah**
hello advise there some issues with open for users form th floor sent re for working hello wireless se thanks sent wednesday for working hello currently we have issue with connecting client for can you please raise ticket if caused by or something side regards	['hello', 'advise', 'there', 'some', 'issue', 'with', 'open', 'for', 'users', 'form', 'th', 'floor', 'sent', 're', 'for', 'work', 'hello', 'wireless', 'se', 'thank', 'sent', 'wednesday', 'for', 'work', 'hello', 'currently', 'we', 'have', 'issue', 'with', 'connect', 'client', 'for', 'can', 'you', 'please', 'raise', 'ticket', 'if', 'caused', 'by', 'or', 'something', 'side', 'regard']
sent wednesday down hello down we cannot work can you please have look thanks software design lead en	['sent', 'wednesday', 'down', 'hello', 'down', 'we', 'cannot', 'work', 'can', 'you', 'please', 'have', 'look', 'thanks', 'software', 'design', 'lead', 'en']
sent thursday server down importance high hi guys servers db web responding please restore functionality asap there ukraine working with among others billing for inactivity regards	['sent', 'thursday', 'server', 'down', 'importance', 'high', 'hi', 'guys', 'servers', 'db', 'web', 'respond', 'please', 'restore', 'functionality', 'asap', 'there', 'ukraine', 'work', 'with', 'among', 'other', 'bill', 'for', 'inactivity', 'regards']
stuck hi teams noticed stuck responsive order fix going restart application fix issue there could be some down could you log ticket sent notification users ad ll keep you updated thank you senior engineer	['stuck', 'hi', 'teams', 'notice', 'stuck', 'responsive', 'order', 'fix', 'go', 'restart', 'application', 'fix', 'issue', 'there', 'could', 'be', 'some', 'down', 'could', 'you', 'log', 'ticket', 'sent', 'notification', 'users', 'ad', 'll', 'keep', 'you', 'update', 'thank', 'you', 'senior', 'engineer']
down between hi we cannot access any servers between seems be down can you please have look urgent because blocking entire text thanks design lead	['down', 'between', 'hi', 'we', 'cannot', 'access', 'any', 'server', 'between', 'seem', 'be', 'down', 'can', 'you', 'please', 'have', 'look', 'urgent', 'because', 'block', 'entire', 'text', 'thank', 'design', 'lead']

3.4.1.5 *Synthetic Minority Oversampling Technique (SMOTE)*

Pada tahap ini, data yang ada akan di*sampling* menggunakan metode SMOTE. Proses *resampling* data ini dikerjakan dengan menggunakan perhitungan dari persamaan 2 . 1.

	Old Sample		
	Urgency	Impact	Priority
1	1	2	2
2	1	2	2
3	2	2	2
4	2	2	2
5	3	3	3
6	3	3	3
7	3	3	3
8	3	3	3
9	3	3	3
10	3	3	3
11	3	3	3
12	3	3	3
13	3	3	3
14	3	3	3
15	3	3	3
16	3	3	3
17	3	3	3
18	3	3	3
19	3	3	3
20	3	3	3

Gambar 3.4 Contoh data pada *dataset* sebelum dilakukan *resampling* oleh SMOTE

Akan dibuat sebuah list untuk mendata *nearest neighbors* dari masing-masing sampel milik kelas minoritas (Data no 1, 2, 3, 4). Data dapat dilihat pada tabel 3.5

Tabel 3.5 Tabel *nearest neighbors* masing-masing sampel kelas minoritas

	NN - 1	**NN - 2**	**NN - 3**
Data-1	2	3	4
Data-2	1	3	4
Data-3	1	2	4
Data-4	1	2	3

Pada data-1 ambil *NN* secara acak dari *list* untuk menentukan nilai *NN* yang akan digunakan untuk membuat *instance* baru. Proses pembentukan *instance* baru dari data-1 dapat dilihat sebagai berikut:

Dari *list NN* dari data 1 = [2, 3, 4] *NN* yang terpilih secara acak adalah data 3.
Data-1 = (1,2)
Data-3 = (2,2)
$N = 3$

Instance baru yang dibuat:
$f_{11} = 1\ f_{21} = 2\ f_{21} - f_{11} = 1$
$f_{12} = 2\ f_{22} = 2\ f_{22} - f_{12} = 0$

Sample yang baru akan dibuat dari perhitungan diatas:
$(f_1', f_2') = (f_{11}, f_{22}) + rand(0,1) * (f_{21} - f_{11}, f_{22} - f_{12})$
$(f_1', f_2') = (1,2) + 1 * (1,0)$
$(f_1', f_2') = (2,2)$
$N = N - 1 = 2$
rand(0,1) akan menghasilkan angka acak antara 0 dan 1.
Sampel baru yang terbentuk dari hasil perhitungan diatas adalah data dengan nilai
Urgency = 2
Impact = 2
Priority = 2

Langkah diatas akan diulangi sampai nilai $N = 0$,
lalu akan dilakukan perhitungan pada data kelas minoritas lainnya.

Berikut adalah tabel hasil perhitungan dari pembuatan *instance* baru masing-masing data 1, 2, 3, 4.

	New Sample		
	Urgency	Impact	Priority
1	2	2	2
2	1	2	2
3	1	2	2
4	2	2	2
5	1	2	2
6	2	2	2
7	2	2	2
8	2	2	2
9	1	2	2
10	2	2	2
11	1	2	2
12	2	2	2

Gambar 3.5 *Instance* baru yang telah dibuat melalui perhitungan diatas

	Old + New Sample		
	Urgency	Impact	Priority
1	1	2	2
2	1	2	2
3	2	2	2
4	2	2	2
5	3	3	3
6	3	3	3
7	3	3	3
8	3	3	3
9	3	3	3
10	3	3	3
11	3	3	3
12	3	3	3
13	3	3	3
14	3	3	3
15	3	3	3
16	3	3	3
17	3	3	3
18	3	3	3
19	3	3	3
20	3	3	3
21	2	2	2
22	1	2	2
23	1	2	2
24	2	2	2
25	1	2	2
26	2	2	2
27	2	2	2
28	2	2	2
29	1	2	2
30	2	2	2
31	1	2	2
32	2	2	2

Gambar 3.6 Isi dataset setelah dilakukan resampling oleh SMOTE

3.4.1.6 *TF-IDF*

Pada tahap awal kata pada tiap dokumen akan dihitung sehingga didapatkan frekuensi kemunculan dari kata tersebut (*term frequency* / tf). Lalu akan dihitung berapa banyak kemunculan kata tersebut pada dokumen-dokumen yang ada. Tahap itu disebut *document frequency* (df). Setelah nilai tf dan df didapatkan maka akan dilakukan perhitungan *inverse* dari nilai df dengan rumus dari persamaan 3 . 1

$$IDF_t = log(N/df) \tag{3 . 1}$$

Di mana :
IDF_t : Hitungan metrik dari banyaknya dokumen yang mengandung kata tersebut.
N : Total dokumen yang ada.
df : Total kemunculan kata pada dokumen-dokumen.

Contoh kata ”chang” ditemukan pada 2 dari 5 dokumen yaitu dokumen D2 dan D3. Maka df dari kata ”chang” adalah 2 dan nilai N nya sebesar 5. Perhitungan IDF_t dari kata ”chang” adalah sebagai berikut:

$$IDF_t = log(5/2) = 0.39794$$

Setelah itu, untuk mendapatkan bobot dari kata ”chang” maka perlu dilakukan perhitungan dengan menggunakan persamaan

$$W_{dt} = tf_{dt} * IDF_t \quad (3\ .\ 2)$$

Di mana :
W_{df} : Bobot dari kata tersebut.
tf_{dt} : Total kemunculan kata pada .
df : Total kemunculan kata pada dokumen-dokumen.

Kata ”chang” muncul pada 2 dokumen yaitu dokumen D2 dan D3. Oleh karena itu, perhitungan $TF - IDF$ nya perlu dilakukan sebanyak 2 kali dengan menggunakan persamaan 3 . 2.

$$W_{D1} = 1 * 0.39794 = 0.39794$$

$$W_{D2} = 2 * 0.39794 = 0.79588$$

Sehingga kata ”chang” memiliki bobot 0.39794 pada dokumen D2 dan bobot 0.79588 pada dokumen D3.

Hasil dari perhitungan pembobotan $TF - IDF$ lain dapat dilihat pada Tabel 3.7

No	Word	TF					Df	N/Df	Idf	TF-IDF				
		D1	D2	D3	D4	D5				D1	D2	D3	D4	D5
0	accept				1		1	5	0.69897	0	0	0	0.69897	0
1	also					1	1	5	0.69897	0	0	0	0	0.69897
2	anymor		1				1	5	0.69897	0	0.69897	0	0	0
3	area					1	1	5	0.69897	0	0	0	0	0.69897
4	asap		1				1	5	0.69897	0	0.69897	0	0	0
5	ask					1	1	5	0.69897	0	0	0	0	0.69897
6	assist		1				1	5	0.69897	0	0.69897	0	0	0
7	authent		1				1	5	0.69897	0	0.69897	0	0	0
8	book					1	1	5	0.69897	0	0	0	0	0.69897
9	chang		1	2			2	2.5	0.39794	0	0.39794	0.79588	0	0
10	current			1			1	5	0.69897	0	0	0.69897	0	0
11	declin				1		1	5	0.69897	0	0	0	0.69897	0
12	design	1					1	5	0.69897	0.69897	0	0	0	0
13	doc					1	1	5	0.69897	0	0	0	0	0.69897
14	en	1		1			2	2.5	0.39794	0.39794	0	0.39794	0	0
15	estim				1		1	5	0.69897	0	0	0	0.69897	0
16	februari				1		1	5	0.69897	0	0	0	0.69897	0
17	floor					1	1	5	0.69897	0	0	0	0	0.69897
18	follow					1	1	5	0.69897	0	0	0	0	0.69897
19	Friday			1			1	5	0.69897	0	0	0.69897	0	0
20	guest					1	1	5	0.69897	0	0	0	0	0.69897
21	guy		1				1	5	0.69897	0	0.69897	0	0	0
22	hello	1					1	5	0.69897	0.69897	0	0	0	0
23	help					1	1	5	0.69897	0	0	0	0	0.69897
24	hi		1	1		1	3	1.666667	0.221849	0	0.221849	0.221849	0	0.221849
25	invalid		1				1	5	0.69897	0	0.69897	0	0	0
26	issu			1			1	5	0.69897	0	0	0.69897	0	0
27	kind		1				1	5	0.69897	0	0.69897	0	0	0
28	kindli					1	1	5	0.69897	0	0	0	0	0.69897
29	lead	1					1	5	0.69897	0.69897	0	0	0	0
30	librari					1	1	5	0.69897	0	0	0	0	0.69897
31	lock		1				1	5	0.69897	0	0.69897	0	0	0
32	look	1					1	5	0.69897	0.69897	0	0	0	0
33	make				1		1	5	0.69897	0	0	0	0.69897	0
34	meet					1	1	5	0.69897	0	0	0	0	0.69897
35	monitor			3			1	5	0.69897	0	0	2.09691	0	0
36	music				1		1	5	0.69897	0	0	0	0.69897	0
37	number			1			1	5	0.69897	0	0	0.69897	0	0
38	octob			1			1	5	0.69897	0	0	0.69897	0	0
39	order				1		1	5	0.69897	0	0	0	0.69897	0
40	part				1		1	5	0.69897	0	0	0	0.69897	0
41	password		2			1	2	2.5	0.39794	0	0.79588	0	0	0.39794
42	pleas	1	1				2	2.5	0.39794	0.39794	0.39794	0	0	0
43	pm				1	2	2	2.5	0.39794	0	0	0	0.39794	0.79588
44	problem		2				1	5	0.69897	0	1.39794	0	0	0
45	propos				2		1	5	0.69897	0	0	0	1.39794	0
46	regard		1				1	5	0.69897	0	0.69897	0	0	0
47	relat		1				1	5	0.69897	0	0.69897	0	0	0
48	resolut			1			1	5	0.69897	0	0	0.69897	0	0
49	resourc				1		1	5	0.69897	0	0	0	0.69897	0
50	room					1	1	5	0.69897	0	0	0	0	0.69897
51	say		2				1	5	0.69897	0	1.39794	0	0	0
52	sent	1		1			2	2.5	0.39794	0.39794	0	0.39794	0	0
53	softwar	1					1	5	0.69897	0.69897	0	0	0	0
54	solut				1		1	5	0.69897	0	0	0	0.69897	0
55	specifi			1			1	5	0.69897	0	0	0.69897	0	0
56	storag					1	1	5	0.69897	0	0	0	0	0.69897
57	tester			1			1	5	0.69897	0	0	0.69897	0	0
58	thank	1		1		2	3	1.666667	0.221849	0.221849	0	0.221849	0	0.443697
59	Tuesday					2	1	5	0.69897	0	0	0	0	1.39794
60	view					1	1	5	0.69897	0	0	0	0	0.69897
61	want			1	1		2	2.5	0.39794	0	0	0.39794	0.39794	0
62	wednesday	1					1	5	0.69897	0.69897	0	0	0	0
63	work	1					1	5	0.69897	0.69897	0	0	0	0

Gambar 3.7

3.4.2 Klasifikasi dengan *Support Vector Machine (SVM)*

3.4.2.1 *Training dengan SVM*

Data hasil pembobotan diubah ke bentuk data SVM dengan format [*kelas urutan_bobot 1:bobot 1 urutan_bobot n:bobot n*]. Nilai pada *kelas* adalah +1 atau -1 untuk mnyatakan label awalan, dimana angka +1 merupakan kelas positif dan angka -1 merupakan kelas negatif.

Berikut adalah contoh pengubahan data teks menjadi data vektor dari hasil

pembobotan TF-ID. Dokumen yang dipakai terdiri dari D1, D2, D3, D4 dan D5 yang sudah diberikan label. Dokumen D1 diberi label 1, dokumen lain diberikan label -1.

Tabel 3.6 Format Vektor

D1	0 0 0 ... 0 0.69897 0.69897
Vektor	[1 1:0 2:0 3:0 .. 62:0 63:0.69897 64:0.69897
Kelas	1
D2	0 0 0.69897 ... 0 0 0
Vektor	[-1 1:0 2:0 3:0.69897 ... 62:0 63:0 64:0]
Kelas	2
D3	0 0 0 ... 0.39794 0 0
Vektor	[-1 1:0 2:0 3:0 ... 62:0.39794 63:0 64:0]
Kelas	2
D4	0.69897 0 0 ... 0.39794 0 0
Vektor	[-1 1:0.69897 2:0 3:0 ... 62:0.39794 63:0 64:0]
Kelas	3
D5	0 0.69897 0 ... 0 0 0
Vektor	[-1 1:0 2:0.69897 3:0 ... 62:0 63:0 64:0]
Kelas	3

Pada Gambar 3.8 berikut merupakan gambaran vektor yang disajikan dalam bentuk tabel.

	D1	D2	D3	D4	R5		D1	D2	D3	D4	R5
kata1	0	0	0	0.69897	0	*kata33*	0.69897	0	0	0	0
kata2	0	0	0	0	0.69897	*kata34*	0	0	0	0.69897	0
kata3	0	0.69897	0	0	0	*kata35*	0	0	0	0	0.69897
kata4	0	0	0	0	0.69897	*kata36*	0	0	2.09691	0	0
kata5	0	0.69897	0	0	0	*kata37*	0	0	0	0.69897	0
kata6	0	0	0	0	0.69897	*kata38*	0	0	0.69897	0	0
kata7	0	0.69897	0	0	0	*kata39*	0	0	0.69897	0	0
kata8	0	0.69897	0	0	0	*kata40*	0	0	0	0.69897	0
kata9	0	0	0	0	0.69897	*kata41*	0	0	0	0.69897	0
kata10	0	0.39794	0.79588	0	0	*kata42*	0	0.79588	0	0	0.39794
kata11	0	0	0.69897	0	0	*kata43*	0.39794	0.39794	0	0	0
kata12	0	0	0	0.69897	0	*kata44*	0	0	0	0.39794	0.79588
kata13	0.69897	0	0	0	0	*kata45*	0	1.39794	0	0	0
kata14	0	0	0	0	0.69897	*kata46*	0	0	0	1.39794	0
kata15	0.39794	0	0.39794	0	0	*kata47*	0	0.69897	0	0	0
kata16	0	0	0	0.69897	0	*kata48*	0	0.69897	0	0	0
kata17	0	0	0	0.69897	0	*kata49*	0	0	0.69897	0	0
kata18	0	0	0	0	0.69897	*kata50*	0	0	0	0.69897	0
kata19	0	0	0	0	0.69897	*kata51*	0	0	0	0	0.69897
kata20	0	0	0.69897	0	0	*kata52*	0	1.39794	0	0	0
kata21	0	0	0	0	0.69897	*kata53*	0.39794	0	0.39794	0	0
kata22	0	0.69897	0	0	0	*kata54*	0.69897	0	0	0	0
kata23	0.69897	0	0	0	0	*kata55*	0	0	0	0.69897	0
kata24	0	0	0	0	0.69897	*kata56*	0	0	0.69897	0	0
kata25	0	0.221849	0.221849	0	0.221849	*kata57*	0	0	0	0	0.69897
kata26	0	0.69897	0	0	0	*kata58*	0	0	0.69897	0	0
kata27	0	0	0.69897	0	0	*kata59*	0.221849	0	0.221849	0	0.443697
kata28	0	0.69897	0	0	0	*kata60*	0	0	0	0	1.39794
kata29	0	0	0	0	0.69897	*kata61*	0	0	0	0	0.69897
kata30	0.69897	0	0	0	0	*kata62*	0	0	0.39794	0.39794	0
kata31	0	0	0	0	0.69897	*kata63*	0.69897	0	0	0	0
kata32	0	0.69897	0	0	0	*kata64*	0.69897	0	0	0	0
Y	1	-1	-1	-1	-1	*Y*	1	-1	-1	-1	-1

Gambar 3.8 Kata yang sudah memiliki label

Selanjutnya, pada tahap ini nilai X akan dihitung untung perhitungan kernel. Untuk x1 semua nilai akan diambil dari kolom D1, begitu juga dengan X2 = D2 dan Xn = Dn. Nilai X1 hingga XN dapat dilihat pada Tabel 3.7.

Tabel 3.7 Nilai x1, x2, x3, x4, x5

x1	**x2**	**x3**	**x4**	**x5**
[0 0 0 ... 0 0.69897 0.69897]	[0 0 0.69897 ... 0 0 0]	[0 0 0 ... 0.39794 0 0]	[0.69897 0 0 ... 0.39794 0 0]	[0 0.69897 0 ... 0 0 0]

Setelah itu akan dilakukan kernelisasi dengan fungsi kernel linear dengan persamaan 3 . 3 yang hasilnya dapat dilihat pada Tabel 3.9.

$$x_1 x_1^T = x_1 \cdot x_1^T \qquad (3\ .\ 3)$$

Tabel 3.8 Perhitungan nilai x1 dengan Kernel

$\mathbf{x_1}$	$\mathbf{x_1^T}$	$\mathbf{x_1 x_1^T = x_1 * x_1^T}$
[0 0 0 ... 0 0.69897 0.69897]	[0; 0; 0; ... 0; 0.69897; 0.69897]	[3.9441990879425]

Untuk perhitungan nilai $x_i x_j^T$ dapat dilihat pada Tabel 3.9.

Tabel 3.9 Matrix Perhitungan Nilai x dengan Kernel

	$\mathbf{x_1^T}$	$\mathbf{x_2^T}$	$\mathbf{x_3^T}$	$\mathbf{x_4^T}$	$\mathbf{x_5^T}$
$\mathbf{x_1}$	3.9441	0.1583	0.3659	0	0.0984
$\mathbf{x_2}$	0.1583	9.7934	0.3659	0	0.3659
$\mathbf{x_3}$	0.3659	0.3659	9.5124	0.1583	0.1476
$\mathbf{x_4}$	0	0	0.1583	7.1565	0.3167
$\mathbf{x_5}$	0.0984	0.3659	0.1476	0.3167	10.3204

Pada tahap berikutnya dilakukan perhitungan terhadap y. Nilai y tersebut dapat dilihat pada Tabel 3.10

Tabel 3.10 Nilai Label pada Y

$\mathbf{y_1}$	$\mathbf{y_2}$	$\mathbf{y_3}$	$\mathbf{y_4}$	$\mathbf{y_5}$
1	-1	-1	-1	-1

Untuk perhitungan nilai $y_i y_j^T$ dapat dilihat pada Tabel 3.11.

Tabel 3.11 Matrix Perhitungan Nilai y dengan Kernel

	$\mathbf{y_1^T}$	$\mathbf{y_2^T}$	$\mathbf{y_3^T}$	$\mathbf{y_4^T}$	$\mathbf{y_5^T}$
$\mathbf{y_1}$	1	-1	-1	-1	-1
$\mathbf{y_2}$	-1	1	1	1	1
$\mathbf{y_3}$	-1	1	1	1	1
$\mathbf{y_4}$	-1	1	1	1	1
$\mathbf{y_5}$	-1	1	1	1	1

Lalu ubah nilai *support vektor* agar mendapat nilai Ai. Nilai x dapat dihitung dengan menggunakan persamaan 3 . 4 kernel linear berikut.

$$\sum_{i=1,j=1}^{n} x_i x_j^T, (i, j = 1, .., n) \qquad (3 \,.\, 4)$$

Hasil perhitungan $x_i x_j^T$ yang telah dihitung adalah berikut:

$$X_{D1} = x_1 x_1^T + ... + x_1 x_5^T = 3.944 + 0.1583 + 0.3659 + 0 + 0.0984 = 4.5669$$

$$X_{D2} = x_2 x_1^T + ... + x_2 x_5^T = 3.944 + 0.1583 + 0.3659 + 0 + 0.0984 = 10.6836$$

$$X_{D3} = x_3 x_1^T + ... + x_3 x_5^T = 3.944 + 0.1583 + 0.3659 + 0 + 0.0984 = 10.5503$$

$$X_{D4} = x_4 x_1^T + ... + x_4 x_5^T = 3.944 + 0.1583 + 0.3659 + 0 + 0.0984 = 7.6316$$

$$X_{D5} = x_5 x_1^T + ... + x_5 x_5^T = 3.944 + 0.1583 + 0.3659 + 0 + 0.0984 = 11.2492$$

Nilai-nilai x yang didapatkan dapat dilihat pada Tabel 3.12.

Tabel 3.12 Nilai X pada setiap dokumen

Dokumen	**D1**	**D2**	**D3**	**D4**	**D5**
X	4.5669	10.6836	10.5503	7.6316	11.2492

Lalu nilai y dapat dihitung menggunakan persamaan 3 . 5 seperti berikut:

$$\sum_{i=1,j=1}^{n} y_i y_j^T, (i, j = 1, .., n) \qquad (3 \,.\, 5)$$

Dengan perhitungan sebagai berikut:

$$Y_{D1} = y_1 y_1^T + ... + y_1 y_5^T = 1 + -1 + -1 + -1 + -1 = -3$$

$$Y_{D2} = y_2 y_1^T + ... + y_2 y_5^T = -1 + 1 + 1 + 1 + 1 = 3$$

$$Y_{D3} = y_3 y_1^T + ... + y_3 y_5^T = -1 + 1 + 1 + 1 + 1 = 3$$

$$Y_{D4} = y_4 y_1^T + ... + y_4 y_5^T = -1 + 1 + 1 + 1 + 1 = 3$$

$$Y_{D5} = y_5 y_1^T + ... + y_5 y_5^T = -1 + 1 + 1 + 1 + 1 = 3$$

sehingga didapatkan nilai y pada tiap dokumen seperti pada Tabel 3.13

Tabel 3.13 Nilai Y pada setiap dokumen

Dokumen	**D1**	**D2**	**D3**	**D4**	**D5**
Y	-3	3	3	3	3

Tabel 3.14 Support Vector Bias

Dokumen	**D1**	**D2**	**D3**	**D4**	**D5**
Support Vector Bias	$\begin{bmatrix}4.5669\\-3\\1\end{bmatrix}$	$\begin{bmatrix}10.6836\\3\\1\end{bmatrix}$	$\begin{bmatrix}10.5503\\3\\1\end{bmatrix}$	$\begin{bmatrix}7.6316\\3\\1\end{bmatrix}$	$\begin{bmatrix}11.2492\\3\\1\end{bmatrix}$

Setelah nilai *Support Vector*, tahap berikutnya adalah mencari nilai a_i, nilai data dapat dihitung menggunakan persamaan 3 . 6.

$$\sum_{i=1}^{n} a_i S_i^T S_j \tag{3 . 6}$$

Didapatkan perhitungan pada D1 sebagai berikut:

$$\alpha_1 \begin{bmatrix}4.5669\\-3\\1\end{bmatrix}^T * \begin{bmatrix}4.5669\\-3\\1\end{bmatrix} = 30.8567\alpha_1$$

$$\alpha_2 \begin{bmatrix}4.5669\\3\\1\end{bmatrix}^T * \begin{bmatrix}10.6836\\3\\1\end{bmatrix} = 40.7912\alpha_2$$

...

...

$$\alpha_5 \begin{bmatrix}4.5669\\3\\1\end{bmatrix}^T * \begin{bmatrix}11.2492\\3\\1\end{bmatrix} = 40.1823\alpha_5$$

Lalu dilakukan hal yang sama terhadap D2 hingga D5, lalu nilai a_i dihitung menggunakan metode *Gauss-Jordan* sehingga mendapatkan nilai a_i seperti berikut:

$$a_1 = 0.055555556233614460066$$

$$a_2 = 0.032042451793126767227$$

$$a_3 = 0.053151650245201842982$$

$$a_4 = -0.11789837006866176686$$

$$a_5 = -0.022851286802488384642$$

Dari perhitungan a_i yang sudah didapatkan maka dapat kita hitung dengan

menggunakan persamaan 3 . 7 untuk mendapatkan nilai w dan b.

$$\tilde{W} = \sum_{i=1}^{n} a_i S_i^T \qquad (3 \,.\, 7)$$

dari persamaan diatas dapat dibentuk nilai $\tilde{W}$ sebagai berikut:

$$\tilde{W}0.05555 \begin{bmatrix} 4.5669 \\ -3 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.25371 \\ -0,1667 \\ 0,0555 \end{bmatrix}$$

Dari persamaan diatas maka didapatkan hasil :

$w = \begin{bmatrix} 0.25371 \\ -0.1667 \end{bmatrix}$, dan dengan $b = 0.0555$

Dengan demikian didapatkan nilai *hyperplane* untuk klasifikasi sebesar 0.0555.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan menjelaskan mengenai proses implementasi dan pengujian terhadap sistem yang telah dibangun berdasarkan penjelasan pada bab sebelumnya.

4.1 Lingkungan Implementasi

Pada lingkungan implementasi, akan dijelaskan mengenai perangkat yang digunakan dalam proses pembangunan sistem baik dari perangkat keras maupun perangkat lunak yang digunakan.

4.1.1 Spesifikasi Perangkat Keras

Spesifikasi dari perangkat keras yang digunakan dalam pembangunan aplikasi adalah sebagai berikut:

1. *Laptop* Acer Nitro AN515-52
2. *Processor* Intel Core i5-8300H CPU @ 2.3GHz
3. *Hard Disk* kapasitas 1TB
4. RAM 20GB

4.1.2 Lingkungan Perangkat Lunak

Spesifikasi dari perangkat lunak yang digunakan dalam pembangunan aplikasi adalah sebagai berikut:

1. Sistem Operasi Windows 10 64-bit.
2. Jupyter Notebook 6.0.1
3. Anaconda3 2019.10
4. Python 3.7.4 64-bit

4.2 Daftar *Class* dan *Method*

Pada sub bab ini akan dijelaskan mengenai *class* dan *method* yang digunakan pada program.

4.2.1 *Class* StemmedCountVectorizer

Class StemmedCountVectorizer merupakan kelas yang berfungsi untuk melaksanakan proses *preprocessing tokenization, stemming* dan *stopwords removal*. Lalu data tersebut akan divektorisasi. Detail *class* terdapat pada Tabel 4.1 dan 4.2.

Tabel 4.1 Daftar Atribut pada *Class* StemmedCountVectorizer

No	*Nama Atribut*	Tipe Data	Keterangan
1	language	string	Pilihan bahasa korpus yang akan digunakan dalam proses *stemming*.
2	ignore_stopwords	boolean	Pilihan untuk menjalankan proses *stopwords removal*.

Tabel 4.2 Daftar *Method* pada *Class* StemmedCountVectorizer

No	*Method*	Keterangan
1	super(StemmedCountVectorizer, self).build_analyzer()	Menerima masukan data teks. Melakukan proses vektorisasi pada data teks. Mengembalikan data teks yang sudah divektorisasi sebagai keluaran.
2	SnowballStemmer(language, ignore_stopwords=boolean) -	Menerima masukan data teks. Melakukan proses *preprocessing stemming* dan *stopwords removal*. Mengembalikan data teks yang sudah diproses sebagai keluaran.

4.3 Implementasi Perangkat Lunak

Pada bagian ini dijelaskan mengenai tahap implementasi aplikasi penanganan data tidak seimbang dan klasifikasi tiket insiden. Seluruh *class* dan *method* yang digunakan pada aplikasi dijelaskan pada sub bab di bawah.

4.3.1 Implementasi *Dataset*

Langkah pertama yang dilakukan adalah membaca *file* data teks yang akan digunakan untuk data *training*. *Dataset* yang digunakan berupa data teks dengan jumlah baris 43107 baris dan 5 kolom. Kolom yang digunakan ada 5 yaitu kolom *title*, *body*, *urgency*, *impact*dan *priority*. Kolom yang digunakan menjadi label ada 3 yaitu kolom *urgency*, *impact* dan *priority*.

title	body	urgency	impact	priority
service unavailable	sent monday accessible hi we unable access thanks	1	1	1
for us not working	hello advise there some issues with open for users form th fl	1	1	1
not responding	sent wednesday down hello down we cannot work can you pl	1	1	1
performance issues and	sent thursday july re incident warning for hi sorry seems ther	1	1	1
connection down th floor	connection was down th floor for few seconds	1	1	1
inaccessible	sent license server problem dear looks like there issue with li	1	1	1
urgent cannot open files on	sent wednesday urgent cannot open files importance high he	1	1	1
wireless network down	sent thursday wireless issue we currently having outage sent	1	1	1
not accessible	sent tuesday accessible dear accessible today could you pleas	1	1	1
certificate expired	multiple users have reported certificate issues	1	1	1
oracle errors when logging	error hello trying connect but working could you please inves	1	1	1

Gambar 4.1 Dataset

	title	body	urgency	impact	priority
count	43107	43107	43107	43107	43107
unique	27346	43107	3	3	3
top	new purchase po	wednesday pm change hi please change questions...	3	3	3
freq	1114	1	32270	32270	32270

Gambar 4.2 Statistik Deskriptif Dataset

4.3.2 Implementasi *Preprocessing*

Pada tahap ini, *dataset* akan diolah dalam proses *processing* dengan tahapan berupa *tokenization*, *case folding*, *stopword removal* dan *stemming*. Tahap ini dilakukan untuk memudahkan data *training* untuk diolah oleh program dan untuk menghilangkan data-data yang tidak diperlukan pada *dataset training*.

4.3.3 Implementation *Synthetic Minority Oversampling Technique*(SMOTE)

Tahap ini merupakan tahap utama dalam penelitian ini dimana *dataset* akan melalui proses *resampling* oleh SMOTE dengan tahapan sebagai berikut.

1. Pilih label dan kelas minoritas yang akan disampling.
2. Cari *nearest neighbor* dari masing-masing data kelas minoritas. Lalu bentuk *array* kumpulan dari *nearest neighbor* masing-masing data kelas minoritas.
3. Lalu, buat *instance* baru dari data kelas minoritas menggunakan algoritma seperti pada Gambar 2.3

Data yang dihasilkan ditahap ini adalah sebagai berikut:

	title	body	urgency	impact	priority
count	69835	69835	69835	69835	69835
unique	27346	43107	3	3	3
top	new purchase po	card problem hi trying submit card for get con...	3.0	3.0	3.0
freq	1114	390	32270	32270	32270

Gambar 4.3 Dataset setelah implementasi oleh SMOTE

4.4 Pengujian

Pada penelitian ini, pengujian yang digunakan adalah pengujian dengan menggunakan *dataset* sebelum dilakukan SMOTE dan *dataset* setelah dilakukan SMOTE. Pengujian dilakukan terhadap 2 *classifier* yang digunakan yaitu *Support Vector Machine*(SVM) dan *Artificial Neural Network*(ANN). Hasil yang dijadikan ukuran dalam pengujian ini adalah pengaruh implementasi SMOTE terhadap nilai *recall* dan akurasi pada *classifier* SVM dan ANN.

4.4.1 Pengujian *Artificial Neural Network*(ANN)

Pada bagian ini, akan dijelaskan perbandingan dari pembuatan model ANN terhadap data latih yang belum diimplmentasi SMOTE dan data latih yang sudah diimplementasi SMOTE.

1. Hasil tes dengan model ANN dengan data latih yang belum diimplementasi SMOTE terhadap kelas *title* dan label *urgency*.

Tabel 4.3 Hasil tes ANN *Title Urgency* sebelum SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.46	**0.46**	0.46	
2	0.41	**0.36**	0.38	
3	1	**0.92**	0.91	
			Accuracy	**0.79**

2. Hasil tes dengan model ANN dengan data latih yang telah diimplementasi SMOTE terhadap kelas *title* dan label *urgency*.

Tabel 4.4 Hasil tes ANN *Title Urgency* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.59	**0.48**	0.53	
2	0.64	**0.67**	0.65	
3	0.87	**0.93**	0.9	
			Accuracy	**0.71**

3. Hasil tes dengan model ANN dengan data latih yang belum diimplementasi SMOTE terhadap kelas *title* dan label *impact*.

Tabel 4.5 Hasil tes ANN *Title Impact* sebelum SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	1	**0.02**	0.04	
2	0.8	**0.62**	0.7	
3	0.88	**0.95**	0.92	
			Accuracy	**0.87**

4. Hasil tes dengan model ANN dengan data latih yang telah diimplementasi SMOTE terhadap kelas *title* dan label *impact*.

Tabel 4.6 Hasil tes ANN *Title impact* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.96	**0.99**	0.97	
2	0.71	**0.57**	0.63	
3	0.88	**0.92**	0.9	
			Accuracy	**0.89**

5. Hasil tes dengan model ANN dengan data latih yang belum diimplementasi SMOTE terhadap kelas *body* dan label *urgency*.

Tabel 4.7 Hasil tes ANN *Body Urgency* sebelum SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.6	**0.57**	0.58	
2	0.58	**0.58**	0.5	
3	0.49	**0.98**	0.98	
			Accuracy	**0.88**

6. Hasil tes dengan model SVM dengan data latih yang telah diimplementasi SMOTE terhadap kelas *body* dan label *urgency*.

Tabel 4.8 Hasil tes SVM *Body Urgency* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.62	**0.53**	0.57	
2	0.65	**0.74**	0.69	
3	0.98	**0.97**	0.98	

			Accuracy	**0.79**

7. Hasil tes dengan model ANN dengan data latih yang belum diimplementasi SMOTE terhadap kelas *body* dan label *impact*.

Tabel 4.9 Hasil tes ANN *Body Urgency* sebelum SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.05	**0.03**	0.03	
2	0.92	**0.96**	0.94	
3	0.99	**0.98**	0.98	
			Accuracy	**0.97**

8. Hasil tes dengan model ANN dengan data latih yang telah diimplementasi SMOTE terhadap kelas *body* dan label *impact*.

Tabel 4.10 Hasil tes ANN *Body Urgency* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.98	**1**	0.99	
2	0.96	**0.94**	0.95	
3	0.99	**0.98**	0.98	
			Accuracy	**0.98**

9. Hasil tes dengan model ANN dengan data latih yang belum diimplementasi SMOTE terhadap label *priority*.

Tabel 4.11 Hasil tes ANN *Body Urgency* sebelum SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0	**0**	0	
2	0.98	**0.99**	0.99	
3	1	**1**	1	
			Accuracy	**0.99**

10. Hasil tes dengan model ANN dengan data latih yang telah diimplementasi SMOTE terhadap label *priority*.

Tabel 4.12 Hasil tes ANN *Body Urgency* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	

1	0.59	**0.65**	0.62	
2	0.54	**0.46**	0.5	
3	0.98	**1**	0.99	
			Accuracy	**0.76**

4.4.2 Pengujian *Support Vector Machine*(SVM)

Pada bagian ini, akan dijelaskan perbandingan dari pembuatan model SVM terhadap data latih yang belum diimplmentasi SMOTE dan data latih yang sudah diimplementasi SMOTE.

1. Hasil tes dengan model SVM dengan data latih yang belum diimplementasi SMOTE terhadap kelas *title* dan label *urgency*.

Tabel 4.13 Hasil tes SVM *Title Urgency* sebelum SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.54	**0.52**	0.53	
2	0.52	**0.2**	0.29	
3	0.87	**0.97**	0.92	
			Accuracy	**0.81**

2. Hasil tes dengan model SVM dengan data latih yang telah diimplementasi SMOTE terhadap kelas *title* dan label *urgency*.

Tabel 4.14 Hasil tes SVM *Title Urgency* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.6	**0.6**	0.6	
2	0.52	**0.52**	0.52	
3	1	**1**	1	
			Accuracy	**0.76**

3. Hasil tes dengan model SVM dengan data latih yang belum diimplementasi SMOTE terhadap kelas *title* dan label *impact*.

Tabel 4.15 Hasil tes SVM *Title Impact* sebelum SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0	**0**	0	
2	0.79	**0.63**	0.7	
3	0.88	**0.95**	0.92	

			Accuracy	**0.87**

4. Hasil tes dengan model SVM dengan data latih yang telah diimplementasi SMOTE terhadap kelas *title* dan label *impact*.

Tabel 4.16 Hasil tes SVM *Title impact* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.96	**0.99**	0.97	
2	0.71	**0.57**	0.63	
3	0.88	**0.92**	0.9	
			Accuracy	**0.89**

5. Hasil tes dengan model SVM dengan data latih yang belum diimplementasi SMOTE terhadap kelas *body* dan label *urgency*.

Tabel 4.17 Hasil tes SVM *Title Urgency* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.6	**0.52**	0.53	
2	0.52	**0.2**	0.29	
3	0.87	**0.97**	0.92	
			Accuracy	**0.81**

6. Hasil tes dengan model SVM dengan data latih yang telah diimplementasi SMOTE terhadap kelas *title* dan label *urgency*.

Tabel 4.18 Hasil tes SVM *Title Urgency* setelah SMOTE

Kelas	*Precision*	*Recall*	*F1-Score*	
1	0.54	**0.52**	0.53	
2	0.52	**0.2**	0.29	
3	0.87	**0.97**	0.92	
			Accuracy	**0.81**

DAFTAR REFERENSI

[1] Lin, Wei-Chao, et al, ”Clustering-based undersampling in class-imbalanced data”, *Information Sciences 409*, Taiwan, 2017, 17-26.

[2] Hartono, Opim Salim Sitompul, Tulus, Erna Budhianti Nababan, “Biased support vector machine and weighted-SMOTE in handling class imbalance problem”, *International Journal of Advances in Intelligent Informatics*, Medan, Sumatera Utara, 2017, pp. 1-4.

[3] Wah, Y. B., Rahman, H. A. A., He, H., Bulgiba, A. (2016, June), ”Handling imbalanced dataset using SVM and k-NN approach”, *In AIP Conference Proceedings (Vol. 1750, No. 1, p. 020023)*, AIP Publishing LLC.

[4] L. Zhang, C. Zhang, R. Gao, R. Yang, and Q. Song, “Using the SMOTE technique and hybrid features to predict the types of ion channel-targeted conotoxins,” J. Theor. Biol., vol. 403, pp. 75–84, 2016, doi: https://doi.org/10.1016/j.jtbi.2016.04.034.

[5] Microsoft Azure Machine Learning Team, ”SMOTE”, https://docs.microsoft.com/en-us/azure/machine-learning/algorithm-module-reference/smote , Diakses 30 September 2020.

[6] M. Bach, A. Werner, J. Zywiec, W. Pluskiewicz, ”The study of under- and over-sampling methods’ utility in analysis of highly imbalanced data on osteoporosis”, Information Sciences Volume 384, April 2017, Pages 174-190.

[7] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, Francisco Herrera. 2018 ”Learning from Imbalanced Data Sets”. Springer.

[8] Margaret Rouse, ”over sampling and under sampling”, https://whatis.techtarget.com/definition/over-sampling-and-under-sampling, Diakses 1 Oktober 2020.

[9] Kurtis Pykes, ”Undersampling and Oversampling A technique for Imbalanced Classification”, https://towardsdatascience.com/oversampling-and-undersampling-5e2bbaf56dcf, Diakses 1 Oktober 2020.

[10] Zhu, Tuanfei, Yaping Lin, and Yonghe Liu. ”Synthetic minority oversampling technique for multiclass imbalance problems.” *Pattern Recognition 72*, Kuala Lumpur, Malaysia, 2017, pp. 327-340.

[11] Haykin, Neural Networks and Learning Machines, 3rd ed. Pearson, 2008.

[12] Fernandez, Alberto, et al, ”Learning from imbalanced data sets”, Berlin:Springer, 2018.

[13] Hensman P, Masko D. The impact of imbalanced training data for convolutional neural networks. Degree Project in Computer Science, KTH Royal Institute of Technology. 2015 May 8.

[14] Johnson, Justin M., and Taghi M. Khoshgoftaar. ”Survey on deep learning with class imbalance.” Journal of Big Data 6.1 (2019): 27.