e-ISSN: 2460-870X | p-ISSN: 2085-1367

Prediksi Retweet Pada Twitter Menggunakan Metode Decision Tree

Prediction of Retweets on Twitter Using the Decision Tree Method

Jondri
Telkom University, Indonesia
jondri@telkomuniversity.ac.id

ABSTRAK

Twitter merupakan media sosial dimana penggunanya dapat memposting suatu informasi berupa tweet. Tweet yang diposting oleh pengguna dapat dibagikan ulang oleh pengguna lain melalui retweet, tweet yang memiliki banyak retweet tersebar lebih cepat dan luas dibandingkan yang tidak mendapat retweet. Tetapi tidak semua tweet mendapatkan retweet karena terdapat fitur-fitur yang mempengaruhi apakah suatu tweet akan mendapat retweet atau tidak. Dalam penelitian ini fitur yang digunakan adalah fitur berbasis pengguna, berbasis waktu dan berbasis konten, serta menggunakan algoritma pembelajaran mesin Decision Tree (DT) jenis classification and regression tree (CART) untuk memprediksi kelas retweet. Masalah yang dihadapi pada penelitian ini adalah data yang tidak seimbang, lalu mengatasinya dengan oversampling dan undersampling. Setelah mengatasi data yang tidak seimbang performansi dari model meningkat terutama saat melakukan undersampling untuk max_depth = 4 menghasilkan nilai akurasi dan f1 85%. Selain itu diperoleh juga bahwa fitur yang paling berpengaruh dalam menentukan apakah suatu tweet akan mendapat retweet atau tidak adalah fitur pengikut_pengguna, usia_akun_pengguna, total_tweet_disukai_pengguna, fitur cek_mention dan panjang_tweet.

Kata Kunci : retweet, tweet, DT, CART, oversampling, undersampling

ABSTRACT

Twitter is a social media where users can post information in the form of tweets. Tweets posted by users can be re-shared by other users via retweets, tweets that have multiple retweets spread more quickly and widely than those that don't get retweets. But not all tweets get retweets because there are features that affect whether a tweet will get retweeted or not. In this study, the features used are user-based, time-based and content-based, as well as using a classification and regression tree (CART) decision tree machine learning algorithm to predict the retweet class. The problem faced in this research is unbalanced data, then overcome it by oversampling and undersampling. After overcoming unbalanced data, the performance of the model increased, especially when undersampling for max_depth = 4 resulted in 85% accuracy and f1. In addition, it was also found that the most influential features in determining whether a tweet will receive a retweet or not are user_follower, user_account age, user_total_tweet_liked_user, check_mention and tweet_length.

Keywords : retweet, tweet, DT, CART, oversampling, undersampling.

Info Artikel :
Disubmit: 27 September 2021 *Direview: 08 Mei 2022* *Diterima : 19 Mei 2022*

1. PENDAHULUAN

Twitter merupakan layanan yang memiliki kemudahan untuk saling berbagi informasi secara *real-time*. Di dalam Twitter terdapat fitur *retweet* yang merupakan bentuk penyebaran informasi dan sarana untuk berpartisipasi dalam sebuah percakapan yang meyebar, dimana seseorang dapat memposting ulang *tweet* yang ditulis oleh pengguna lain saat menemukan *tweet* yang menarik untuk dibagikan kepada pengikutnya [1]. Penyebaran *tweet* tidak hanya menyampaikan suatu informasi ke pengguna lain saja, tetapi bisa dengan memvalidasi dan mengomentari isi *tweet* [2]. Penyebaran informasi di Twitter tidak merata, dari sekian banyak informasi yang dibagikan di jejaring social tidak semuanya dapat menyebar secara luas [1]. Artinya, dari banyaknya *tweet* yang dibagikan tersebut ada yang mendapat *retweet* dan ada yang tidak mendapat *retweet*.

DOI: https://www.doi.org/10.22303/csrid.14.2.2022.113-124

Beberapa penelitian terdahulu terkait prediksi *retweet* atau biasa disebut difusi informasi pada jaringan sosial Twitter sudah cukup banyak dilakukan. Dari penelitian sebelumnya memberikan arahan bahwa suatu *tweet* dapat di *retweet* oleh pengguna lainnya karena terdapat fitur-fitur yang mempengaruhinya.

Telah diteliti perilaku pengguna terhadap *retweet* pada data Twitter menggunakan 3 metode yaitu J48 (yang merupakan implementasi Java pada algoritma C4.5), Support Vector Machine(SVM), dan *Logistic regression.* Dalam mengidentifikasi faktor-faktor yang berkaitan dengan perilaku *retweet* individu, peneliti melakukan perbandingan "tinggalkan satu fitur" pada 22 fitur yang digunakan dalam model. Setiap kali menghapus satu fitur itu dapat mempengaruhi metrik F1 dari model yang dibangun. Fitur berbasis sosial (pengguna) merupakan fitur terpenting untuk memprediksi *retweet* dibandingkan fitur lainnya. Dan J48 menghasilkan *recall* = 0,84, *precision* = 0,825, dan F1 *metric* = 0,832 yang menjadikan J48 lebih baik dibandingkan SVM dan *logistic regression.* Sedangkan SVM performansinya lebih baik dibandingkan *logistic regression* [4].

Fitur yang memiliki pengaruh penting pada perilaku *retweet* dibagi kedalam empat kategori, yaitu fitur pengguna, fitur teks, fitur temporal, dan fitur metadata. Data yang digunakan merupakan *tweet* berbahasa inggris. Model prediksi menggunakan Regresi Logistik untuk memprediksi hasil kelas data berdasarkan satu fitur atau lebih. Regresi Logistik dapat mengukur hubungan antara variabel independen dan dependen kategorikal, menggunakan skor probabilitas sebagai nilai prediksi dari variabel dependen, karena metode ini tidak hanya digunakan untuk menyelesaikan masalah klasifikasi tetapi bisa juga untuk menyelesaikan masalah regresi. Hasil dari penelitian menunjukan model prediksi analisis regresi memiliki akurasi prediksi dan efektifitas yang lebih baik [5].

Model *machine learning Random Forest* (RF), *Naive Bayes* (NB), dan *Support Vector Machine* (SVM) dan fitur berbasis pengguna, waktu, dan konten telah diteliti untuk memprediksi *retweet*. Hasil yang didapatkan metode RF meningkatkan 5% f-measure dibandingkan dengan dua metode lainnya yaitu NB dan SVM. Selain itu, peneliti juga menunjukkan bahwa beberapa fitur yang diperkenalkan sangat penting untuk memprediksi retweetabilitas seperti jumlah pengikut dan jumlah komunitas yang dimiliki pengguna [6].

Pada penelitian sebelumnya terdapat kaitan mengenai fitur-fitur yang digunakan, hasil penelitian dari studi terkait fitur berbasis pengguna atau sosial mempengaruhi retweetabilitas seperti *followers*, *following*, usia akun, verifikasi akun dan jumlah grup. Selain itu, fitur berbasis konten juga berpengaruh seperti *mention*, *hashtags*, dan *URL*. Kemudian dari penelitian [6] yang mengembangkan dari penelitian [1] dengan menambah fitur berbasis waktu untuk memeriksa apakah *tweet* diposting pada siang hari, malam hari, diakhir pekan atau selama liburan juga sangat berkorelasi dengan kemampuan *retweet*.

Fitur konten (URL, *hashtags* dan *mention*) dan fitur kontekstual (jumlah *tweet*, jumlah *following*, jumlah *followers*, jumlah *tweet favorit* dan usia akun) telah diteliti untuk mengidentifikasi faktor yang sangat terkait dengan tingkat *retweet*. Dari penelitian ditemukan bahwa dari fitur konten, URL dan *hashtags* memiliki hubungan yang kuat dengan *retweetability*. Pada fitur kontekstual jumlah *following*, jumlah *followers*, dan usia akun mempengaruhi retweetabilitas. Sementara jumlah *tweet* sebelumnya tidak dapat memprediksi retweetabilitas *tweet* pengguna [1].

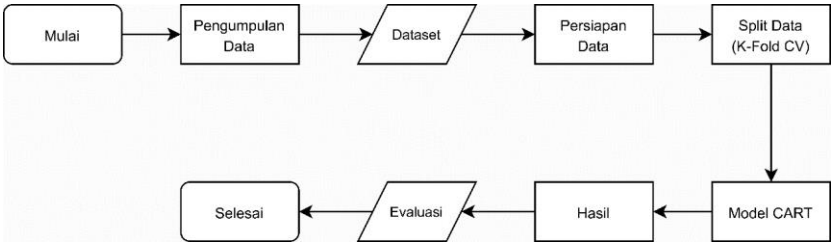
Penelitian ini bertujuan untuk merancang suatu model menggunakan algoritma pohon keputusan jenis CART untuk memprediksi apakah suatu *tweet* masuk ke dalam kelas *retweet* atau tidak menggunakan fitur berbasis pengguna, berbasis waktu dan berbasis konten, serta mengatasi ketidakseimbangan data dengan *oversampling* dan *undersampling*. Metode CART bisa digunakan pada himpunan data yang jumlahnya besar dan variabel yang banyak untuk menentukan aturan-aturan yang kompleks. Penelitian tentang analisis algoritma CART pernah dilakukan untuk memprediksi *influenza* pada pasien perawatan primer, hasilnya algoritma CART memiliki sensitivitas yang baik dan *Negative Prediction Value* (NPV) yang tinggi, tetapi *Positive Prediction Value* (PPV) yang rendah untuk mengidentifikasi *influenza* pada pasien rawat jalan ≥ 5 tahun.

e-ISSN: 2460-870X | p-ISSN: 2085-1367

Sehingga, bagus untuk mengidentifikasi kelompok yang tidak memerlukan tes atau antivirus dan memiliki kinerja prediktif yang baik untuk *influenza* [3].

2. METODE PENELITIAN

Dibawah ini merupakan *flowchart* dari metodologi penelitian yang dilakukan, dapat dilihat pada Gambar 1:



Gambar 1. Flowchart metodologi penelitian

A. Pengumpulan Data

Dataset dikumpulkan melalui Twitter menggunakan Twitter API dengan *query search* "covid" adalah data satu hari yang lalu (tanggal 08 Juni 2021) yang diambil dalam satu hari yang sama untuk menghindari bias, karena *tweet* yang ditulis dalam waktu yang berbeda dalam sehari memiliki kemungkinan berbeda untuk di *retweet* [4]. *Tweet* tersebut diambil pada tanggal 09 Juni 2021 pukul 17:12 sebanyak 10.000 data. Tabel 1 merupakan 10 buah sampel dari *dataset* yang sudah dikumpulkan.

Tabel 1. Dataset Penelitian

No	1	2	3	4	5	6	7	8	9	10
Usia Akun Pengguna	8,7	11,4	1,2	0,7	7	0,6	12,3	4	0	12,3
Pengikut Pengguna	147	1171	11	1993	675	76	8156812	129	3	8156485
Diikuti Pengguna	284	136	26	1678	568	12	25	160	3	25
Total Tweet Disukai Pengguna	0	6029	5250	31719	9778	2	125	39854	30	125
Total Tweet Pengguna	37436	4425	646	19963	13942	3661	1886987	21844	104	1887002
Cek Verifikasi Akun Pengguna	0	0	0	0	0	0	0	0	0	0
Waktu Posting Pagi	0	0	0	0	0	0	1	1	1	1
Waktu Posting Siang	0	0	0	1	1	1	0	0	0	0
Waktu Posting Sore	0	1	1	0	0	0	0	0	0	0
Waktu Posting Malam	1	0	0	0	0	0	0	0	0	0
Posting di Hari Kerja	1	1	1	1	1	1	1	1	1	1
Posting di Hari Libur	0	0	0	0	0	0	0	0	0	0
Panjang Tweet	138	139	140	140	126	68	124	139	140	100

DOI: https://www.doi.org/10.22303/csrid.14.2.2022.113-124

Cek Tweet Huruf Kapital	0	0	0	0	0	0	0	0	0	0
Cek Tagar	0	0	0	0	0	0	0	0	0	0
Cek Mention	0	1	1	0	1	0	0	0	0	0
Retweet	0	0	1	1	0	1	0	0	1	1

a) Fitur

Dalam *machine learning*, fitur adalah suatu input variabel yang nantinya dapat mempengaruhi hasil prediksi. Dari *dataset* yang dikumpulkan lalu dipilih beberapa fitur yang akan digunakan dalam penelitian, diantaranya fitur berbasis pengguna, waktu, dan konten yang sudah digunakan pada penelitian [1,6]. Tetapi untuk fitur berbasis waktu penulis mendefinisikan batasan waktu yang berbeda dari penelitian sebelumnya [6], serta fitur verifikasi_akun_pengguna [4]. Untuk selebihnya dijelaskan secara detail pada Tabel 2.

Tabel 2. Fitur-fitur yang dipergunakan

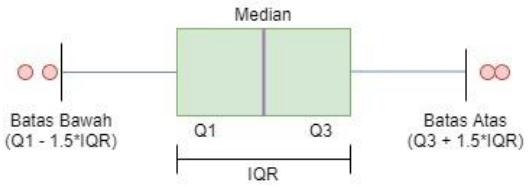
No	Jenis Fitur	Nama Fitur	Tipe	Deskripsi
1	Berbasis Pengguna	usia_akun_pengguna	Float	Merupakan umur akun pengguna, selisih dari tahun ini dengan tahun dibuat.
2		pengikut_pengguna	Int	Jumlah akun yang mengikuti pengguna.
3		diikuti_pengguna	Int	Jumlah akun yang diikuti oleh pengguna.
4		total_tweet_disukai_pengguna	Int	Jumlah *tweet* yang disukai pengguna.
5		total_tweet_pengguna	Int	Jumlah *tweet* yang diposting oleh pengguna.
6		cek_verifikasi_akun_pengguna	Boolean	Mengecek apakah akun sudah terverifikasi.
7	Berbasis Waktu	waktu_posting_pagi	Boolean	*Tweet* yang diposting pada jam 04.00 s/d jam 09.59 WIB.
8		waktu_posting_siang	Boolean	*Tweet* yang diposting pada jam 10.00 s/d jam 13.59 WIB.
9		waktu_posting_sore	Boolean	*Tweet* yang diposting pada jam 14.00 s/d jam 18.59 WIB.
10		waktu_posting_malam	Boolean	*Tweet* yang diposting pada jam 19.00 s/d jam 3.59 WIB.
11		posting_di_hari_kerja	Boolean	*Tweet* yang diposting pada hari Senin s/d hari Jum'at.
12		posting_di_hari_libur	Boolean	*Tweet* yang diposting pada hari Sabtu dan hari Minggu.
13	Berbasis Konten	panjang_tweet	Int	Jumlah karakter pada *tweet* yang diposting pengguna.
14		cek_tweet_huruf_kapital	Boolean	Mengecek apakah *tweet* yang diposting merupakan huruf kapital.
15		cek_tagar	Boolean	Mengecek apakah *tweet* yang diposting mengandung tagar.
16		cek_mention	Boolean	Mengecek apakah *tweet* yang diposting menyebut *username* pengguna lain.
17	Label (Kelas)	*Retweet*	Boolean	Menentukan suatu *tweet* masuk kedalam kelas *Retweet* atau Tidak. (Jika *tweet* memiliki minimal 1 *Retweet*, maka *tweet* tersebut masuk kedalam kelas *Retweet*).

e-ISSN: 2460-870X | p-ISSN: 2085-1367

B. Persiapan Data

Pada *dataset* yang digunakan perlu dilakukan proses persiapan data sebelum data tersebut diimplementasikan pada model, pada tahap ini dilakukan beberapa proses persiapan data yaitu:

- Mengecek *missing value* dengan hasil tidak terdapat *missing value* pada *dataset* yang sudah dikumpulkan.
- Mengatasi nilai duplikasi, terdapat 1.519 data duplikasi dari 10.000 data, lalu peneliti melakukan *drop* terhadap nilai duplikasi, sehingga terdapat 8.481 data yang tersisa.
- *Outlier* yaitu kemunculan data yang nilainya langka, nilai yang muncul bisa sangat kecil atau sangat besar sehingga jauh dengan nilai lain yang memiliki kelompok. *Outlier* merupakan hal penting karena dapat berpengaruh terhadap hasil metode yang digunakan [7]. *Boxplot* adalah cara standar untuk menampilkan distribusi data yang didalamnya terdapat nilai "minimum", kuartil pertama (Q1), median, kuartil ketiga (Q3) dan "maksimum". *Boxplot* juga dapat memberikan informasi tentang *outlier* dan nilainya. Dibawah ini merupakan visualisasi *boxplot* dapat dilihat pada Gambar 2.



Gambar 2. Visualisasi *boxplot*

Inter Quartile Range (IQR) sangat berguna karena sering digunakan untuk mencari *outlier*. Data yang kurang dari batas bawah atau melebihi batas atas merupakan *outlier*. Berikut ini merupakan rumus batas bawah dan batas atas

$$\text{Batas\_Bawah} = \text{Q1} - 1.5 * \text{IQR} \quad (1)$$

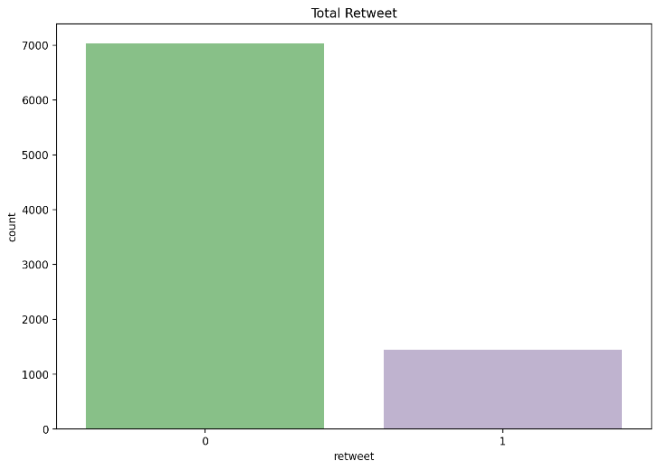
$$\text{Batas\_Atas} = \text{Q3} + 1.5 * \text{IQR} \quad (2)$$

Lalu didalam penelitian ini mengecek *outlier* pada beberapa fitur yang bertipe numerik, diantaranya usia_akun_pengguna, pengikut_pengguna, diikuti_pengguna, total_tweet_disukai_pengguna, total_tweet_pengguna, dan panjang_tweet. Untuk mengatasi *outlier* dibahas kembali pada bagian Skenario Pengujian 2.

- *Imbalence Data* merupakan kumpulan data klasifikasi dengan proporsi kelas tidak seimbang. Kelas yang terbentuk dari sebagian besar kumpulan data disebut kelas mayoritas. Sedangkan kelas yang memiliki proporsi lebih kecil adalah kelas minoritas. Dalam penulisan ini terdapat dua cara yang dilakukan dalam mengatasi *imbalence data*:
    - *Undersampling* mengurangi proporsi kelas mayoritas hingga jumlahnya sama dengan kelas minoritas. Ketika dua titik milik kelas yang berbeda sangat dekat satu sama lain dalam distribusi, algoritma ini menghilangkan titik data dari kelas mayoritas sehingga mencoba untuk menyeimbangkan distribusi. Metode "NearMiss-1" memilih sampel dari kelas mayoritas yang dekat dengan beberapa sampel kelas minoritas. Dengan cara ini, sampel dari kelas mayoritas dipilih ketika jarak rata-ratanya ke tiga sampel dari kelas minoritas terdekat adalah yang terkecil [8].
    - *Oversampling* menduplikasi proporsi kelas minoritas mengikuti proporsi kelas mayoritas untuk mendapatkan distribusi kelas 1:1. *Synthetic Minority Over-sampling Technique* (SMOTE) memanfaatkan algoritma *K-Nearest Neighbor* (KNN) untuk membuat data sintetik dimulai dengan memilih data acak dari kelas minoritas, lalu menghitung tetangga kelas terdekat untuk setiap sampel minoritas berdasarkan tingkat *oversampling* yang diperlukan, lalu memilih secara acak dari sampel tetangga ini. Contoh sintetik kemudian dibangkitkan pada titik acak sepanjang segmen garis yang menghubungkan contoh minoritas dengan tetangga yang dipilih ini [8].

DOI: https://www.doi.org/10.22303/csrid.14.2.2022.113-124

Pada penelitian ini mengecek *imbalance data* pada kelas 0 (tidak *retweet*) dan 1 (*retweet*). Visualisasi *imbalance* data dapat dilihat pada Gambar 3. Untuk mengatasi *imbalance data* dibahas kembali pada bagian Skenario Pengujian 3 dan Skenario Pengujian 4.



Gambar 3. I*mbalance* data, 0 kelas tweet yang tidak diretweet, 1 kelas data tweet yang diretweet

- Normalisasi merupakan teknik penskalaan dengan menggeser nilai dan mengubah skalanya sehingga nilainya berkisar antara 0 dan 1. Salah satu teknik penskalaan dengan normalisasi yaitu menggunakan min-max, X_{max} dan X_{min} adalah nilai maksimum dan minimum dari fitur. Pada penelitian ini dilakukan *scalling* pada fitur-fitur yang bertipe numerik menggunakan rumus normalisasi dengan penskalaan *min-max* yang dapat dilihat pada Rumus (3):

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (3)$$

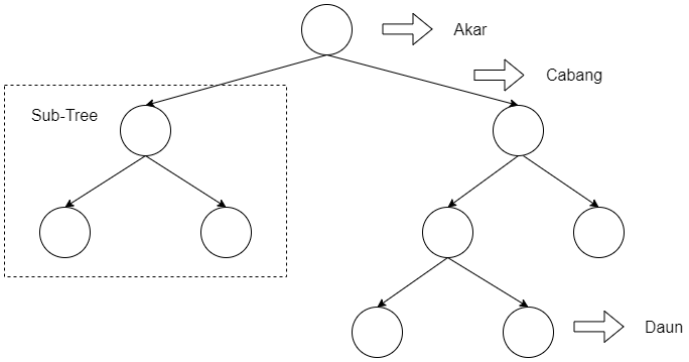
C. K-Fold Cross Validation

Pada k-*fold cross validation*, *dataset* dipartisi menjadi k *subset* yang terpisah dengan ukuran yang kira-kira sama. *Fold* mengacu pada jumlah himpunan bagian yang dihasilkan. Model dilatih menggunakan k-1 himpunan bagian, yang bersama-sama mewakili *train set*. Kemudian, model diterapkan ke *subset* yang tersisa yang merupakan set validasi, dan kinerjanya diukur. Prosedur ini diulang sampai masing-masing k *subset* telah berfungsi sebagai set validasi. Performansi cross validation dihitung dari performansi pada setiap k yang dirata-ratakan[9]. Pada penelitian ini menggunakan K-*Fold Cross Validation* untuk melakukan *split data* dengan jumlah *fold* 10, perbandingan datanya yaitu 9:1 untuk setiap *fold* yang digunakan untuk data *train* sebesar 90% dan 10% untuk data *test* pada data sebanyak 8481.

D. Model Pohon Keputusan

Algoritma Pohon Keputusan merupakan salah satu teknik *supervised learning* yang populer untuk klasifikasi. Pada prinsipnya, pohon keputusan digunakan untuk memprediksi data kedalam suatu kelas berdasarkan variabel prediktor. Fleksibilitas teknik ini menarik, karena memberikan visualisasi yang sugestif ('pohon' yang secara sintetis merangkum klasifikasi) [10]. Pada pohon keputusan terdapat beberapa bagian penting yaitu *node* yang paling atas disebut *root node*, simpul internal mewakili sebuah fitur (atau atribut), cabang mewakili aturan keputusan, dan simpul daun mewakili hasilnya. Gambar 4 ini merupakan contoh pohon keputusan.

e-ISSN: 2460-870X | p-ISSN: 2085-1367



Gambar 4. Pohon keputusan

Metode CART dapat diterapkan baik untuk fitur kategorikal maupun kontinu. Pembuatan *Classification Tree* (CT) dimulai dengan simpul induk yang berisi seluruh kumpulan *dataset* atau biasa disebut *root node*, kemudian pada *root node* dilakukan pemisahan biner rekursif berdasarkan evaluasi setiap atribut, untuk membuat partisi sampel yang lebih homogen atau murni daripada simpul induk untuk setiap *node* berikutnya sampai dengan simpul akhir (*node* daun) yang merupakan hasil prediksi nilai atau kelas [11]. CART menggunakan indeks Gini untuk memilih fitur yang optimal dan menentukan titik segmentasi biner yang optimal. Dalam masalah klasifikasi, asumsikan ada K kelas dan P_k merupakan probabilitas titik sampel yang termasuk kedalam kelas k. Maka Indeks Gini dari distribusi probabilitas tersebut didefinisikan pada Rumus 4 [11]:

$$GINI(P) = 1 - \sum_{k=1}^{K} P_k^2 \quad (4)$$

Jika himpunan sampel D terbagi menjadi D_1 dan D_2 menurut fitur i, maka di bawah kondisi fitur i, indeks gini dari himpunan D didefinisikan pada Rumus 5 (idealnya, 0) [11]:

$$GINI(D,i) = \frac{|D_1|}{|D|} GINI(D_1) + \frac{|D_2|}{|D|} GINI(D_2) \quad (5)$$

Metode yang digunakan untuk mengklasifikasikan apakah suatu *tweet* masuk kedalam kelas *retweet* atau tidak *retweet*, menggunakan algoritma pohon keputusan jenis CART dengan mengatur nilai dari parameter model yaitu *max_depth* yang dimulai dari kedalaman 3 sampai dengan kedalaman 7 untuk menghindari terjadinya *overfitting* dan menggunakan indeks gini untuk kriteria *splitting*.

E. Skenario Pengujian

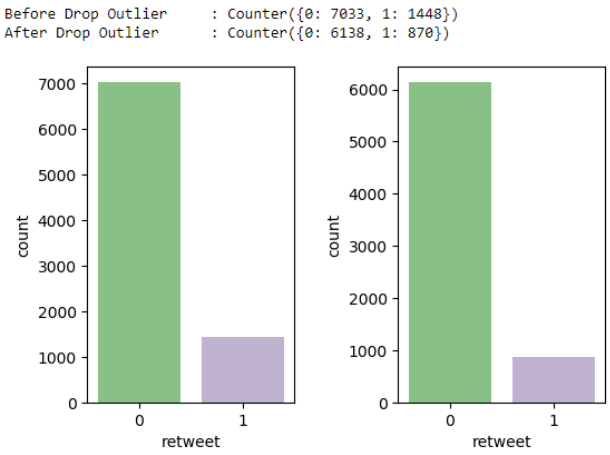
- Skenario Pengujian 1

  Pada skenario pengujian 1, hanya melakukan *drop* duplikasi data dan *scaling* data menggunakan *min-max scaller*, setelah itu data langsung dimodelkan.
- Skenario Pengujian 2

  Pada skenario pengujian 2, mengatasi *outlier* dengan menggunakan IQR (*Interquartile Range*). Dengan membatasi Q1 = 0,10 dan Q3 = 0,90 data diluar batas Q1 dan Q3 di *drop*. Terdapat 1.473 data yang di *drop* sehingga data yang tersisa 7.008.

  Gambar 5 merupakan hasil imbalance data setelah dilakukan *drop outlier*, sebelum melakukan *drop outlier* proporsi data kelas 0 sebanyak 7.033 (83%) dan kelas 1 sebanyak 1.448 (17%), setelah dilakukan *drop outlier* proporsi data kelas 0 menjadi 6.138 (88%) dan kelas 1 menjadi 870 (12%).

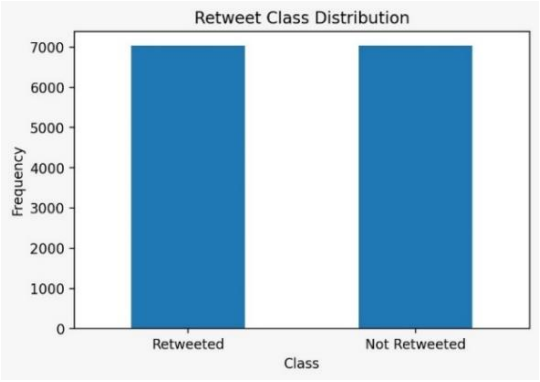
DOI: https://www.doi.org/10.22303/csrid.14.2.2022.113-124



Gambar 5. Distribusi kelas setelah *drop outlier*

- Skenario Pengujian 3

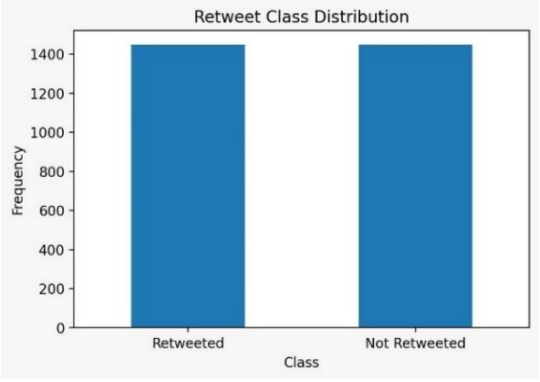
  Pada skenario pengujian 3, mengatasi *imbalance* data dengan *oversampling* menggunakan *method* SMOTE dari *imblearn*. Karena kelas 0 (tidak *retweet)* dan kelas 1 (*retweet*) tidak seimbang maka dilakukan *oversampling* dengan menduplikasi kelas minor agar seimbang dengan kelas major. Jumlah data yang digunakan menjadi 14.066 data, dengan perbandingan 1:1 yaitu 7033 untuk kelas major dan 7033 untuk kelas minor. Pada Gambar 6 merupakan hasil visualisasi distribusi kelas setelah dilakukan *oversampling*.



Gambar 6. Visualisasi distribusi kelas dengan *oversampling*

- Skenario Pengujian 4

  Pada skenario pengujian 4, mengatasi *imbalance* data dengan *undersampling* menggunakan metode *NearMiss* dari i*mblear*n dengan cara mengurangi data kelas major agar seimbang dengan kelas minor. Jumlah data yang digunakan menjadi 2896 data, dengan perbandingan 1:1 yaitu 1448 untuk kelas major dan 1448 untuk kelas minor. Pada Gambar 7 merupakan hasil visualisasi distribusi kelas setelah dilakukan *undersampling*.



Gambar 7. Visualisasi distribusi kelas dengan *undersampling*

e-ISSN: 2460-870X | p-ISSN: 2085-1367

F. Evaluasi

Evaluasi dari model menggunakan f1-score dan akurasi, dimana rumusnya adalah

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

$$\text{F1 Score} = 2\,.\,\frac{\text{Recall}*\text{Precission}}{\text{Recall+Precission}} \quad (7)$$

dimana

$$\text{Presisi} = \frac{\text{TP}}{\text{TP+FP}} \quad (8)$$

$$Recall = \frac{\text{TP}}{\text{TP+FN}} \quad (9)$$

Dimana TP = True Positive, TN = True negative, FP = False Positive, FN = False Negative

3. HASIL DAN PEMBAHASAN

A. Hasil Pengujian

Tabel 3 merupakan hasil dari pengujian scenario 1 yang hanya dilakukan *drop* duplikasi data dan *scaling*, menghasilkan skor akurasi *train* dan akurasi *test* yang tinggi. Karena datanya tidak seimbang pengukurannya menggunakan f1 dan yang paling besar skor-nya hanya 45% tercapai pada *max_depth* = 7.

Tabel 3. Hasil pengujian skenario 1

Max Depth	***Accuracy Test***	***Accuracy Train***	**F1**
3	0.84	0.85	0.32
4	0.85	0.85	0.44
5	0.85	0.86	0.41
6	0.84	0.87	0.41
7	0.85	0.88	0.45

Pada Tabel 4 merupakan hasil dari skenario pengujian 2, saat *outlier* diatasi dapat meningkatkan skor akurasi *train* dan akurasi *test* dari pengujian 1. Tetapi pada nilai f1 mengalami penurunan dari skenario pengujian 1 karena setelah dilakukan *drop outlier* ketidakseimbangan data semakin jauh antara kelas 0 dan 1. Pada pengujian 2, karena datanya tidak seimbang pengukurannya menggunakan f1, nilai f1 paling besar yang diperoleh 32% pada *max_depth* = 7.

Tabel 4. Hasil pengujian skenario 2

Max Depth	***Accuracy Test***	***Accuracy Train***	**F1**
3	0.88	0.88	0.28
4	0.88	0.88	0.24
5	0.88	0.89	0.26
6	0.88	0.90	0.29
7	0.88	0.91	0.32

Pada Tabel 5 merupakan hasil dari pengujian skenario 3, saat mengatasi ketidakseimbangan kelas "*retweet"* dan "tidak *retweet*" menggunakan *oversampling* dengan metode SMOTE, skor akurasi *train* dan akurasi *test* mengalami penurunan dari kedua pengujian sebelumnya. Tetapi untuk nilai f1 mengalami kenaikan karena datanya sudah seimbang. Pada hasil pengujian 3 jika dibandingkan dengan *max_depth* lainnya, *max_depth* = 7 merupakan yang terbaik karena memiliki akurasi paling tinggi 80% juga f1 paling tinggi 79%.

DOI: https://www.doi.org/10.22303/csrid.14.2.2022.113-124

Tabel 5. Hasil Pengujian Skenario 3

Max Depth	***Accuracy Test***	***Accuracy Train***	**F1**
3	0.76	0.76	0.74
4	0.77	0.77	0.76
5	0.78	0.79	0.77
6	0.79	0.80	0.78
7	0.80	0.82	0.79

Pada Tabel 6 merupakan hasil dari pengujian skenario 4, saat mengatasi ketidakseimbangan kelas "*retweet"* dan "tidak *retweet*" menggunakan *undersampling* dengan method *NearMiss*, skor akurasi *train* dan akurasi *test* mengalami kenaikan dari skenario pengujian 3 meskipun tidak lebih tinggi dari skenario pengujian 2. Tetapi untuk f1 hasil nya paling baik dibandingkan dengan ketiga pengujian sebelumnya. Pada hasil pengujian 4, *max_depth* terbaik ada pada *max_depth* = 4. Karena, akurasi *test* dan akurasi *train*-nya memiliki selisih skor paling sedikit yaitu 1%. Selain itu, pada *max_depth* = 4 juga memiliki nilai f1yang paling tinggi.

Tabel 6. Hasil Pengujian Skenario 4

Max Depth	***Accuracy Test***	***Accuracy Train***	**F1**
3	0.84	0.85	0.84
4	0.85	0.86	0.85
5	0.85	0.87	0.84
6	0.85	0.88	0.84
7	0.84	0.89	0.83

B. Pembahasan

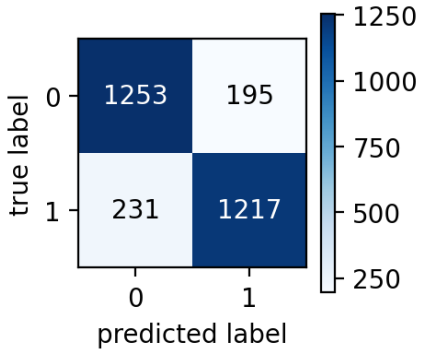
Pada Skenario pengujian 2 saat *outlier* diatasi dibandingkan dengan skenario pengujian 1 yang hanya dilakukan *drop* duplikasi data dan *scaling* akurasinya meningkat karena *outlier* mempengaruhi kinerja model, sehingga saat *outlier* diatasi akurasinya meningkat. Tetapi nilai f1 turun dari pengujian 1, sehingga skenario pengujian 1 dan skenario pengujian 2 bukan skenario pengujian terbaik karena memiliki f1 score yang rendah. Akurasi yang bagus saja tidak cukup untuk mengukur performansi model karena dapat menjebak saat terjadi ketidakseimbangan data pada kelas, jadi perlu dilihat f1 *score* nya saat data tidak seimbang. Kemudian disini penulis mengatasi ketidakseimbangan data pada skenario pengujian 3 dengan melakukan *oversampling* dan pada skenario 4 melakukan *undersampling*. Dapat dilihat pada Tabel 5 dan Tabel 6 bahwa f1 *score* mengalami peningkatan dari kedua skenario pengujian yang sudah dilakukan sebelumnya. Saat data tidak seimbang metrik yang digunakan sebagai pengukuran adalah f1, tetapi saat data sudah seimbang maka pengukuran dapat menggunakan akurasi. Jika dibandingkan antara pengujian 3 dan pengujian 4, baik nilai akurasi maupun f1 hasilnya lebih tinggi pengujian 4. Oleh karena itu, skenario pengujian 4 dengan *max_depth* = 4 dipilih sebagai model yang performansinya paling baik, pada Tabel 7 merupakan hasil *confusion matrix*-nya. Lalu untuk setiap TP, FP, FN dan TN dijumlahkan terlebih dahulu setiap *fold*-nya, kemudian hasil penjumlahan tersebut divisualisasikan pada Gambar 8 dan pada Gambar 9 merupakan hasil visualisasi pohon keputusan dengan *max_depth* = 4.

Tabel 7. Confusion Matrix Skenario Pengujian 4 dengan max_depth = 4

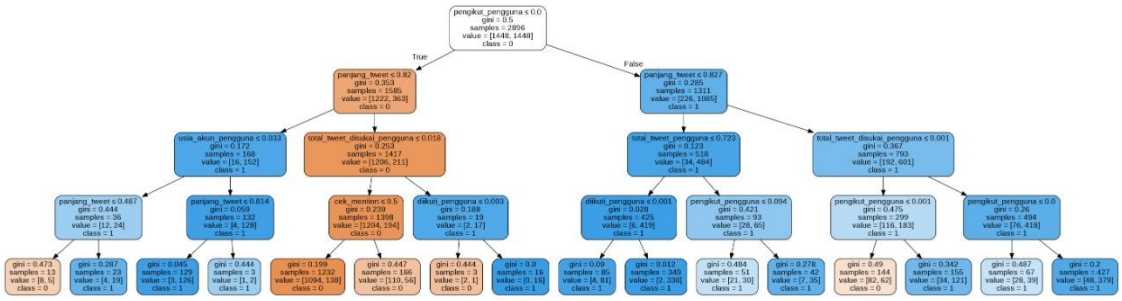
***Fold* ke-**	**TP**	**FP**	**FN**	**TN**
1	137	23	20	110
2	109	17	27	137
3	135	31	17	107
4	128	17	22	123
5	116	19	27	128
6	123	12	22	133
7	129	21	23	116
8	118	21	27	123

e-ISSN: 2460-870X | p-ISSN: 2085-1367

9	132	20	20	117
10	126	14	26	123
Jumlah	1253	195	231	1217



Gambar 8. Visualisasi Confusion Matrix Skenario Pengujian 4 dengan max_depth = 4



Gambar 9 Visualisasi pohon keputusan dari pengujian terbaik

4. KESIMPULAN

Berdasarkan analisis pengujian yang sudah dilakukan dapat disimpulkan bahwa metode *Decision Tree* jenis CART yang dibangun dapat memprediksi kelas *retweet* dan tidak *retweet* terhadap suatu *tweet*, dengan membatasi *max_depth* dari 3 sampai dengan 7, karena jika *max_depth* kurang dari 3 nilai akurasi dan f1 nya rendah dan apabila *max_depth* lebih dari 7 terjadi *overfitting*. Lalu dari skenario pengujian terbaik, fitur yang paling berpengaruh adalah fitur pengikut_pengguna, usia_akun_pengguna, total_tweet_disukai_pengguna dan total_tweet_pengguna yang dikelompokan berdasarkan fitur berbasis pengguna. Selain itu, fitur cek_mention dan panjang_tweet juga berpengaruh terhadap prediksi kelas *retweet* yang dikelompokan berdasarkan fitur berbasis konten. Serta model yang digunakan performansinya lebih baik saat mengatasi *imbalance* data dengan *oversampling* dan *undersampling*. Terutama saat menggunakan *undersampling* dengan method *NearMiss* untuk *max_depth* = 4, menghasilkan nilai akurasi dan f1 sebesar 85%.

Saran untuk kedepannya, fitur yang digunakan dapat dikembangkan menjadi lebih banyak, dan menggunakan lebih dari satu metode *machine learning* agar dapat dibandingkan antar satu metode dengan metode lainnya.

REFERENSI

[1] Suh, B., Hong, L., et al. 2010. "Want to be retweeted? large scale analytics on factor impacting retweet in twitter network", *IEEE International Conference on Social Computing* 177-184

[2] Boyd, D., Golder, S., and Lotan, G. 2010. "Tweet, tweet, retweet: Conversational aspects of retweeting on twitter", *43rd Hawaii international conference on system sciences* 1-10

[3] Zimmerman, R. K., Balasubramani, G. K., et al. 2016. "Classification and Regression Tree (CART) analysis to predict influenza in primary care patients", *BMC infectious diseases* 1-11

[4] Xu, Z., and Yang, Q. 2012. "Analyzing user retweet behavior on twitter", *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* 46-50

[5] Yu, H., Bai, X. F., et al. 2015. "Prediction of users retweet times in social network", *International Journal of Multimedia and Ubiquitous Engineering* 10. 5, 315-322

[6] Hoang, T. B. N., and Mothe, J. 2018. "Predicting information diffusion on twitter–analysis of predictive features", *Journal of computational science*. 28, 257-264

[7] Gorunescu, F. (2011). *Data Mining: Concepts, models and techniques* 12. Springer Science & Business Media.

[8] Fernández, A., García, S., et al. (2018). *Learning from imbalanced data sets*. 10. Berlin: Springer.

[9] Berrar, D. 2019. "Cross-validation", *Encyclopedia of Bioinformatics and Computational Biology* 1. 542–545

[10] Buskirk, T. D. (2018). "Surveying the forests and sampling the trees: An overview of classification and regression trees and random forests with applications in survey research", *Survey Practice* 11. 1-13

[11] Tang, R., and Zhang, X. 2020. "CART Decision Tree Combined with Boruta Feature Selection for Medical Data Classification", *5th IEEE International Conference on Big Data Analytics (ICBDA)* 80-84