View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by ejournal.nusamandiri.ac.id (STMIK Nusa Mandiri)

KOMPARASI ALGORITMA DENGAN PENDEKATAN RANDOM UNDERSAMPLING UNTUK MENANGANI KETIDAKSEIMBANGAN KELAS PADA PREDIKSI CACAT *SOFTWARE*

Ginabila[1]; Ahmad Fauzi[2]

Ilmu Komputer
STMIK Nusa Mandiri
www.nusamandiri.ac.id
[1]14002151@nusamandiri.ac.id, [2]fauzi.aau@nusamandiri.ac.id

Ciptaan disebarluaskan di bawah Lisensi Creative Commons Atribusi-NonKomersial 4.0 Internasional.

Abstract— *Testing is a process that becomes a standard in producing quality software. In predictions of software defects, prediction errors are very bad. Incorrect and inappropriate data sets result in inaccurate prediction results will be affect the software itself. This study aims to overcome the problem of class imbalance with the software defect prediction data set, through the Random Undersampling (RUS) data level approach by taking several algorithms namely Naive Bayes (NB), J48 and Random Forest (RF) which aims to compare the accuracy level highest so that maximum results are obtained in the process of predicting software defects. From the results of this study it can be found that to overcome class imbalances using the Random Undersampling level data approach to predict software defects, the highest level of accuracy is obtained by the Random Forest algorithm with an accuracy rate of 71.932%.*

Intisari— Pengujian saat ini merupakan proses yang menjadi standar dalam menghasilkan *software* berkulitas. Dalam prediksi cacat *software,* kesalahan prediksi merupakan hal yang sangat buruk. Set data yang salah dan tidak sesuai mengakibatkan kurang akuratnya hasil prediksi dan akan berpengaruh terhadap *software* itu sendiri. Penelitian ini bertujuan untuk mengatasi masalah ketidakseimbangan kelas terhadap set data prediksi cacat *software*, melalui pendekatan level data Random Undersampling (RUS) dengan mengambil beberapa algoritma yaitu Naive Bayes (NB), J48 dan Random Forest (RF) yang bertujuan untuk membandingkan mana tingkat akurasi yang paling tinggi sehingga didapatkan hasil yang maksimal dalam proses memprediksi cacat *software*. Dari hasil penelitian ini dapat ditemukan bahwa untuk mengatasi ketidakseimbangan kelas dengan menggunakan pendekatan level data *Random Undersampling* untuk memprediksi kecacatan pada *software*, tingkat akurasi yang paling tinggi didapatkan oleh algoritma Random Forest dengan tingkat akurasi sebesar 71,932%.

Kata Kunci*: Naive Bayes, J48, Random Forest, Random Undersampling, Prediksi Cacat Software.*

PENDAHULUAN

Prediksi cacat *software* merupakan salah satu cara yang efektif untuk meminimalkan biaya dan usaha dalam membangun atau mengembangkan *software*. Salah satu ciri *software* berkualitas adalah yang paling minimal memiliki kecenderungan cacat (Akbar & Rochimah, 2017).

Atribut kode statis atau biasa disebut *software metric* diekstraksi dari *software* untuk memprediksi modul yang rusak. Prediksi yang akurat pada cacat modul perangkat lunak sebagai bagian dari upaya untuk mengurangi meningkatnya biaya pengembangan dan pemeliharaan perangkat lunak (Putri & Frieyadie, 2017). Prediktor atau modul yang terdeteksi memiliki kecenderungan cacat diharapkan dapat membantu meningkatkan kualitas *software* untuk dirilis pada periode berikutnya (Shuo Wang & Xin Yao, 2013). Berbagai *software metrics* yang berbeda ditemukan setelah proses ekstraksi *software* dan digunakan untuk memprediksi kecenderungan cacat. Untuk mempermudah dan membuat proses prediksi kecacatan *software* lebih praktis, dipilih set *metrics* yang paling penting dari semua *software metrics* yang ada (Okutan & Yildiz, 2014).

Banyaknya dataset untuk memprediksi kecacatan *software* yang tersedia dalam repositori terbuka dan dapat diakses hanya melalui internet

memudahkan para peneliti untuk menemukan data yang dibutuhkan untuk penelitiannya. Tetapi dengan data yang telah tersedia pada repositori sering ditemukan kasus dimana satu kelas dalam data tersebut memiliki jumlah yang sangat besar perbandingannya dari kelas yang lain sehingga algoritma data mining tidak dapat menghasilkan model pengklasifikasi yang optimal untuk memprediksi modul yang cacat.

Keseimbangan kelas atau *class imbalanced* merupakan hal yang sangat penting dalam rangka menghasilkan data training atau *training set* yang baik. Hampir semua algoritma klasifikasi menunjukkan performa yang sangat buruk ketika bekerja pada data dengan kelas yang sangat tidak seimbang atau imbalance. Pada hakekatnya data real adalah tidak seimbang (Siringoringo, 2017).

Pada penelitian sebelumnya ketidakseimbanagn kelas diatasi dengan berbagai pendekatan level data yaitu *Random Undersampling, Random Oversampling* dan FSMOTE dengan algoritma Naive Bayes. Hasil dari penelitian tersebut pendekatan level data FSMOTE dengan algoritma Naive Bayes menghasilkan tingkat akurasi yang paling tinggi untuk mengatasi ketidakseimbangan kelas data (Aries & Wahono, 2015).

Oleh karena itu penelitian ini bertujuan untuk mengatasi masalah ketidakseimbangan kelas melalui pendekatan level data *Random Undersampling* (RUS) dengan mengambil beberapa algoritma untuk membandingkan mana tingkat akurasi yang paling tinggi untuk dikombinasikan dengan pendekatan level data yang diambil.

BAHAN DAN METODE

Dalam penelitian ini dilakukan komparasi tiga algoritma pengklasifikasi yaitu Naive Bayes (NB), J48 dan Random Forest (RF). Komparasi ketiga algoritma ini bertujuan untuk mengetahui nilai akurasi mana yang paling tinggi dari penggunaan pendekatan level data *Random Undersampling* untuk mengatasi ketidak seimbangan kelas dari data yang diambil dari suatu repositori.

Naive Bayes merupakan metode yang tidak memiliki aturan, Naïve Bayes menggunakan cabang matematika yang dikenal dengan teori probabilitas untuk mencari peluang terbesar dari kemungkinan klasifikasi, dengan cara melihat frekuensi setiap klasifikasi pada data training. naïve bayes merupakan metode klasifikasi populer dan termasuk dalam sepuluh algoritma terbaik dalam data mining (Frastian, Hendrian, & Valentino, 2018). Prediksi Bayes didasarkan pada formula teorema Bayes dengan formula umum sebagai berikut :

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \quad \text{..... (1)}$$

Keterangan:
H = Mempresentasikan sebuah kelas
X = Mempresentasikan sebuah atribut
P(H|X) = *Posterior Probability*
P (X|H) = Kemungkinan munculnya prediktor
P(H) = *Prior Probability*

J48 adalah salah satu jenis *classifier* pada metode klasifikasi dalam data mining dan bagian dari C4.5 decision tree yang sederhana. C4.5 membangun sebuah pohon keputusan berdasarkan pada seperangkat input data yang berlabel. Pohon keputusan adalah model prediksi menggunakan struktur pohon atau struktur berhirarki. Konsep dari pohon keputusan adalah mengubah data menjadi pohon keputusan dan aturan-aturan keputusan (Diwandari & Setiawan, 2015). Ukuran information gain digunakan untuk memilih atribut uji pada setiap node di dalam tree. Ukuran ini digunakan untuk memilih atribut atau node pada pohon. Atribut dengan nilai information gain tertinggi akan terpilih sebagai parent bagi node selanjutnya. Sebelum menghitung gain harus dihitung terlebih dahulu nilai entropy-nya. Entropy adalah suatu parameter untuk mengukur heterogenitas (keberagaman) dari suatu kumpulan data sampel. Apabila sampel data semakin heterogen maka nilai dari entropy-nya semakin besar (Andri, Kunang, & Murniati, 2013). Formula dari entropy adalah:

$$Entropy\ (S) = \sum_i^c - pi\ log2\ pi \quad \text{..... (2)}$$

Keterangan:
c = Jumlah kelas klasifikasi
pi = Jumlah sampel untuk kelas i

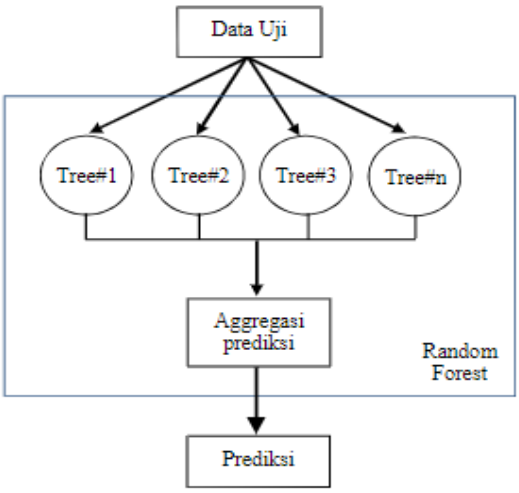
Setelah nilai entropy diperoleh maka langkah selanjutnya adalah menghitung gain untuk mengukur efektifitas suatu atribut dalam mengklasifikasi data. Gain dihitung dengan menggunakan rumus:

$$Gain(S,A) = Entropy(s) - \sum_{v \in values(A)} \frac{Sv}{s} * EntropySv) \quad \text{.. (3)}$$

Keterangan:
v = Nilai yang mungkin untuk atribut A
Values(A) = Himpunan nilai yang mungkin untuk atribut A
Sv = Jumlah sampel data untuk nilai v
S = Jumlah seluruh sampel data

*Entropy(*Sv) = *Entropy* untuk sampel yang memiliki nilai v.

Random forest merupakan pengembangan dari Algoritma C4.5 (*decision tree*) dengan menggunakan beberapa *decision tree*, dimana setiap decision tree telah dilakukan training data menggunakan sampel individu dan setiap atribut dipecah pada *tree* yang dipilih antara atribut subset yang bersifat acak. Dan dalam perkembangannya, sejalan dengan bertambahnya dataset, maka *tree* pun ikut berkembang (Frastian et al., 2018). Proses prediksi Random Forest ditunjukkan oleh gambar 1 berikut:



Sumber : (Putra, Wibawa, & Purnomo, 2016)

Gambar 1. Proses Prediksi Random Forest

Random forest yang dihasilkan memiliki banyak tree dan setiap tree akan tumbuh dengan cara yang sama. Tree dengan variabel x akan ditempatkan pada jarak yang jauh dengan tree variabel y. Sejalan dengan bertambahnya dataset maka tree pun ikut berkembang. Penempatan tree yang saling berjauhan akan memudahkan dalam deteksi jenis tree. Tree yang berada di sekitar tree x maka tree tersebut merupakan perkembangan dari tree x sedangkan tree yang berada disekitar tree y maka tree tersebut merupakan perkembangan dari tree y. Pembangunan tree akan berhenti ketika data sudah homogen atau jika batas jumlah data minimum sudah terlewati.

Penelitian ini dilakukan dengan beberapa tahap:

1. Pengumpulan Data

   Penulis menggunakan data *software* defect yang telah tersedia dari PROMISE repositori, diambil dari situs www.tunedit.org. Data tersebut berisi 22 atribut *software metrics* dan terdiri dari 2 kelas yaitu, 1783 kelas false dan 326 kelas true. Atribut-Atribut yang terdapat dalam data tersebut adalah:

Tabel 1. Atribut Dalam Data

No	Atribut
1	loc
2	v(g)
3	ev(g)
4	iv(g)
5	n
6	v
7	l
8	d
9	i
10	e
11	b
12	t
13	lOCode
14	lOComment
15	lOBlank
16	locCodeAndComment
17	uniq_Op
18	uniq_Opnd
19	total_Op
20	total_Opnd
21	branchCount
22	defects

Sumber: (PROMISE, 2010)

2. Pengolahan Data

   Dalam proses pengolahan data, terdapat 2 tahap yang dilakukan oleh peneliti yaitu:

a. Proses Pengolahan Data Asli

   Dalam tahap ini peneliti mengkomparasikan 3 algoritma pengklasifikasi yang diambil dalam penelitian dengan data awal yang didapatkan dari PROMISE repositori tanpa digunakan pendekatan level data *Random Undersampling*.

b. Proses Pengolahan Ketidakseimbangan Kelas

   Pada tahap ini 3 algoritma pengklasifikasi digunakan dengan pendekatan level data *Random Undersampling* untuk mengetahui tingkat akurasi masing-masing algoritma setelah masalah ketidakseimbangan kelasnya diatasi.

Nilai pada masing-masing kelas dan rata-rata dari kedua kelas yang dibutuhkan untuk mengetahui keakuratan prediksi cacat *software* akan dihitung dan didapatkan tingkat akurasinya:

- TP Rate (True Positive Rate)

  $$TPR = \frac{TP}{TP+FN} \quad \text{.....(4)}$$

  Keterangan:
  TP (*True Positive*) = Data positif yang terdeteksi benar
  FN (*False Negative*) = Data Positif yang terdeteksi negatif

- FP Rate (False Positive Rate)

  $$FPR = \frac{FP}{FP+TN} \quad \text{..... (5)}$$

  Keterangan:
  FP (*False Positive*) = Data negatif yang terdeteksi positif

TN (*True Negative*) = Data negatif yang terdeteksi benar

- Precision

  Precision = $\frac{TP}{TP+FP}$ ..... (6)

  Keterangan:
  TP (*True Positive*) = Data positif yang terdeteksi benar
  FP (*False Positive*) = Data negatif yang terdeteksi positif

- Recall

  Recall = $\frac{TP}{TP+FN}$ ..... (7)

  Keterangan:
  TP (*True Positive*) = Data positif yang terdeteksi benar
  FN (*False Negative*) = Data Positif yang terdeteksi negatif

- F-Measure

  F-Measure = 2 . $\frac{Precision\ .\ Recall}{Precision+Recall}$ ..... (8)

  Keterangan:
  Precision = Tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem
  Recall = Tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi.

- MCC

  MCC = $\frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$ .. (9)

  Keterangan:
  TP (*True Positive*) = Data positif yang terdeteksi benar
  FP (*False Positive*) = Data negatif yang terdeteksi positif
  TN (*True Negative*) = Data negatif yang terdeteksi benar
  FN (*False Negative*) = Data Positif yang terdeteksi negatif

- ROC Area

  Kurva ROC adalah grafik dua dimensi dimana nilai TPR (True Positive Rate) mewakili sumbu Y dan nilai FPR (False Positive Rate) mewakili sumbu X (Tharwat, 2018). Semakin dekat garis dengan angka 1 maka akan semakin baik. Pada bagian hasil dan pembahasan juga akan ditampilkan grafik ROC area dari masing masing kelas.

- PRC Area

  Precision Recal Curve biasanya digunakan jika ditemui kasus dimana data memiliki kelas yang tidak seimbang. Sesuai dengan namanya Kurva ini dibuat berdasarkan nilai yang telah didapatkan pada perhitungan dengan confusion matrix, yaitu antara Precision dan Recall.

- Accuracy

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

  Keterangan:
  TP (*True Positive*) = Data positif yang terdeteksi benar
  FP (*False Positive*) = Data negatif yang terdeteksi positif
  TN (*True Negative*) = Data negatif yang terdeteksi benar
  FN (*False Negative*) = Data Positif yang terdeteksi negatif

Perhitungan-perhitungan yang telah disebutkan sebelumnya akan dihitung melalui WEKA. Waikato Environment for Knowledge Analysis (WEKA) merupakan perangkat lunak pembelajaran mesin yang populer yang ditulis dalam bahasa pemrograman java. WEKA berisikan kumpulan algoritma beserta visualisasinya untuk analisis data dan pemodelan prediktif. Algoritma-algoritma pembelajaran mesin pada WEKA dapat dimanfaatkan untuk pemecahan masalah dibidang data mining. WEKA memiliki implementasi semua teknik pembelajaran untuk klasifikasi dan regresi , yaitu decision trees, rules set, pengklasifikasian teorema bayes, Support Vector Machines (SVM), logistik dan linier, multi layers perceptrons dan metode nearest neighbour. (Frank, Hall, Trigg, Holmes, & Witten, 2004)

HASIL DAN PEMBAHASAN

1. Hasil Perhitungan Menggunakan Algoritma Naive Bayes, J48 dan Random Forest Sebelum Menggunakan RUS

Dari 2109 data *software* defect yang ada, dengan menggunakan algoritma pengklasifikasi Naive Bayes, J48 dan Random Forest didapatkan hasil perhitungan Tabel 2 berikut:

Tabel 2. Hasil Perhitungan Naive Bayes

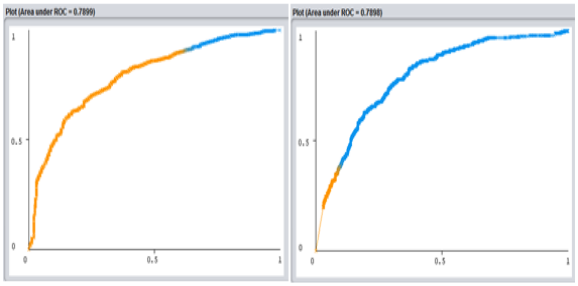
	Naive Bayes		
	Class False	**Class True**	**Rate**
TP Rate	0,905	0,377	0,824
FP Rate	0,623	0,095	0,541
Precision	0,888	0,421	0,816
Recall	0,905	0,377	0,824
F-Measure	0,897	0,398	0,820

	Naive Bayes		
	Class False	Class True	Rate
MCC	0,296	0,296	0,296
ROC Area	0,790	0,790	0,790
PRC Area	0,945	0,396	0,860

Sumber: (Ginabila & Fauzi, 2019)

Dalam pengujian kedua kelas menggunakan algoritma Naive bayes, rata-rata sebesar 0,824 dari seluruh data positif telah terdeteksi benar pada kelasnya masing-masing. Pada *Class False* data negatif yang terdeteksi positif sebesar 0,623 sedangkan pada *class true* hanya sebesar 0,095. Untuk perhitungan lain yang lebih jelas dapat dilihat pada tabel 2.

Berikut adalah grafik ROC dari masing masing kelas dengan algoritma Naive bayes sebelum digunakannya pendekatan level data:



Sumber: (Ginabila & Fauzi, 2019)
Gambar 2. ROC Class False & Class True Algoritma Naive Bayes

Kurva ROC diatas menunjukkan bahwa ROC berada pada true positive sebesar 0,790 pada *class false* dan *class true* dengan menggunakan algoritma Naive Bayes.

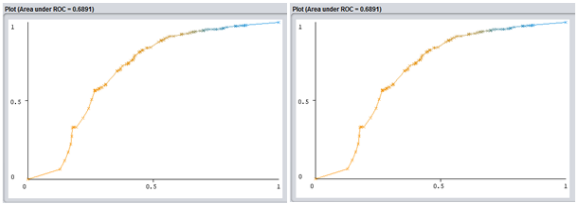
Tabel 3. Hasil Perhitungan J48

	J48		
	Class False	Class True	Rate
TP Rate	0,939	0,331	0,845
FP Rate	0,669	0,061	0,575
Precision	0,885	0,500	0,825
Recall	0,939	0,331	0,845
F-Measure	0,911	0,399	0,832
MCC	0,323	0,323	0,323
ROC Area	0,698	0,689	0,689
PRC Area	0,885	0,364	0,804

Sumber: (Ginabila & Fauzi, 2019)

Dalam pengujian kedua kelas menggunakan algoritma J48, rata-rata sebesar 0,845 dari seluruh data positif telah terdeteksi benar pada kelasnya masing-masing. Pada *Class False* data negatif yang terdeteksi positif sebesar 0,669 sedangkan pada *class true* hanya sebesar 0,061. Untuk perhitungan lain lebih jelasnya dapat dilihat pada tabel 3.

Berikut adalah grafik ROC dari masing masing kelas dengan algoritma J48 sebelum digunakannya pendekatan level data:



Sumber: (Ginabila & Fauzi, 2019)
Gambar 3. ROC Class False & Class True Algoritma Naive Bayes

Kurva ROC diatas menunjukkan bahwa ROC berada pada true positive sebesar 0,698 pada *class false* dan 0,689 pada *class true* dengan menggunakan algoritma J48.

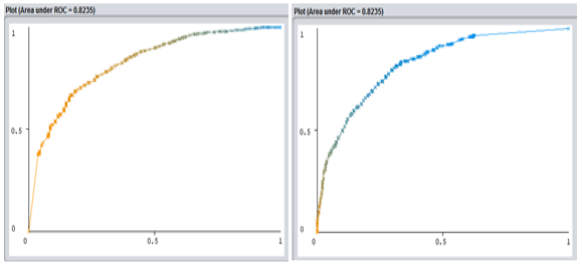
Tabel 4. Hasil Perhitungan Random Forest

	RandomForest		
	Class False	Class True	Rate
TP Rate	0,965	0,328	0,867
FP Rate	0,672	0,035	0,573
Precision	0,887	0,633	0,848
Recall	0,965	0,328	0,867
F-Measure	0,925	0,432	0,848
MCC	0,391	0,391	0,391
ROC Area	0,824	0,823	0,823
PRC Area	0,956	0,498	0,886

Sumber: (Ginabila & Fauzi, 2019)

Dalam pengujian kedua kelas menggunakan algoritma Random Forest, rata-rata sebesar 0,867 dari seluruh data positif telah terdeteksi benar pada kelasnya masing-masing. Pada *Class False* data negatif yang terdeteksi positif sebesar 0,669 sedangkan pada *class true* hanya sebesar 0,061. Untuk perhitungan lain lebih jelasnya dapat dilihat pada tabel 4.

Berikut adalah grafik ROC dari masing masing kelas dengan algoritma Random Forest sebelum digunakannya pendekatan level data:



Gambar 4. ROC Class False & Class True Algoritma Random Forest

Kurva ROC Gambar 4 diatas menunjukkan bahwa ROC berada pada true positive sebesar 0,824 pada *class false* dan 0,823 pada *class true* dengan menggunakan algoritma Random Forest.

Setelah melalui perhitungan yang dilakukan, dapat diketahui perbandingan tingkat

akurasi dari masing masing algoritma sebelum digunakannya pendekatan level data adalah Tabel 5 berikut:

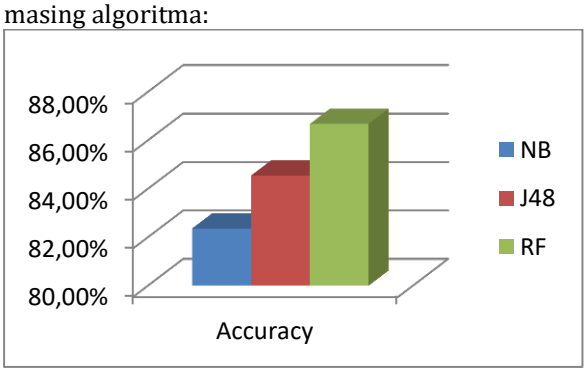
Tabel 5. Hasil Perhitungan Akurasi NB, J48 dan RF

	NB	J48	RF
Correctly Classified Instances	82,361%	84,542%	86,676%
Incorrectly Classified Instances	17,639%	15,458%	13,324%

Sumber: (Ginabila & Fauzi, 2019)

Dapat dilihat dari tabel diatas bahwa sebelum digunakannya pendekatan level data untuk menangani ketidakseimbangan kelas, tingkat akurasi yang didapatkan dari masing-masing algoritma yaitu 82,361% untuk penggunaan algoritma Naive Bayes, 84,542% dengan Algoritma J48 dan 86,676% dengan menggunakan Algoritma Random Forest.

Berikut Gambar 5 adalah grafik perbandingan tingkat akurasi sebelum digunakan pendekatan level data Random Undersampling dari masing-masing algoritma:



Sumber: (Ginabila & Fauzi, 2019)

Gambar 5. Perbandingan Tingkat Akurasi dari Ketiga Algoritma

Dari grafik perbandingan tingkat akurasi diatas dapat dilihat bahwa penggunaan Algoritma Random Forest adalah yang paling tinggi tingkat akurasinya.

2. Hasil Perhitungan Menggunakan Algoritma Naive Bayes, J48 dan Random Forest Setelah Menggunakan RUS

Dari 2109 data dengan kelas yang tidak seimbang, dilakukan pendekatan level data Random Undersampling untuk mengatasi ketidakseimbangan kelas tersebut. Dengan digunakannya pendekatan level data, data yang dihitung akan menjadi seimbang dengan 326 data berada pada *class false* dan 326 berada pada *class true.* Berikut Tabel 6 hasil perhitungan yang didapatkan setelah digunakan pendekatan level data RUS:

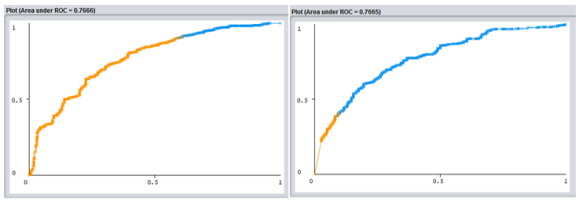
Tabel 6. Hasil Perhitungan RUS + Naive Bayes

	Naive Bayes		
	Class False	Class True	Rate
TP Rate	0,905	0,402	0,653
FP Rate	0,598	0,095	0,347
Precision	0,602	0,809	0,705
Recall	0,905	0,402	0,653
F-Measure	0,723	0,537	0,630
MCC	0,355	0,355	0,355
ROC Area	0,767	0,767	0,767
PRC Area	0,743	0,759	0,751

Sumber: (Ginabila & Fauzi, 2019)

Dalam penggunaan pendekatan level data Random Undersampling dengan algoritma Naive bayes, rata-rata sebesar 0,653 dari seluruh data positif telah terdeteksi benar pada kelasnya masing-masing. Pada *Class False* data negatif yang terdeteksi positif sebesar 0,598 sedangkan pada *class true* hanya sebesar 0,095. Untuk perhitungan lain yang lebih jelas dapat dilihat pada tabel 6.

Berikut Gambar 6 adalah grafik dari ROC menggunakan RUS + Naive Bayes dari masing masing kelas:



Sumber: (Ginabila & Fauzi, 2019)

Gambar 6. RUS + Naive Bayes ROC Class False & Class True

Kurva ROC Gambar 6 diatas menunjukkan bahwa ROC berada pada true positive sebesar 0,767 pada *class false* dan *class true* setelah digunakannya pendekatan level data *Random Undersampling* dengan algoritma Naive Bayes.

Tabel 7. Hasil Perhitungan RUS + J48

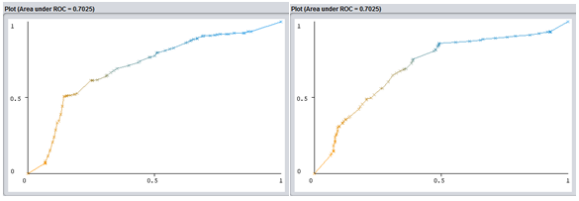
	J48		
	Class False	Class True	Rate
TP Rate	0,644	0,687	0,666
FP Rate	0,313	0,356	0,334
Precision	0,673	0,659	0,666
Recall	0,644	0,687	0,666
F-Measure	0,658	0,673	0,665
MCC	0,332	0,332	0,332
ROC Area	0,703	0,703	0,703
PRC Area	0,650	0,661	0,655

Sumber: (Ginabila & Fauzi, 2019)

Dalam penggunaan pendekatan level data Random Undersampling dengan algoritma J48, rata-rata sebesar 0,666 dari seluruh data positif

telah terdeteksi benar pada kelasnya masing-masing. Pada *Class False* data negatif yang terdeteksi positif sebesar 0,313 sedangkan pada *class true* sebesar 0,356. Untuk perhitungan lain yang lebih jelas dapat dilihat pada tabel 7.

Berikut adalah grafik dari ROC menggunakan RUS + J48 dari masing masing kelas:



Sumber: (Ginabila & Fauzi, 2019)

Gambar 7. RUS + J48 ROC Class False & Class True

Kurva ROC Gambar 7diatas menunjukkan bahwa ROC berada pada true positive sebesar 0,703 pada *class false* dan *class true* setelah digunakannya pendekatan level data *Random Undersampling* dengan algoritma J48.

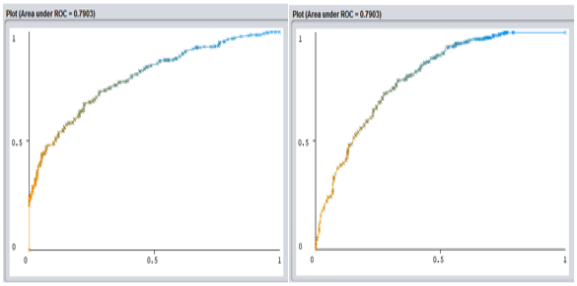
Tabel 8. Hasil Perhitungan RUS + Random Forest

	RandomForest		
	Class False	Class True	Rate
TP Rate	0,721	0,718	0,719
FP Rate	0,282	0,279	0,281
Precision	0,719	0,720	0,719
Recall	0,721	0,718	0,719
F-Measure	0,720	0,719	0,719
MCC	0,439	0,439	0,439
ROC Area	0,790	0,790	0,790
PRC Area	0,816	0,758	0,787

Sumber: (Ginabila & Fauzi, 2019)

Dalam penggunaan pendekatan level data Random Undersampling dengan algoritma Random Forest, rata-rata sebesar 0,719 dari seluruh data positif telah terdeteksi benar pada kelasnya masing-masing. Pada *Class False* data negatif yang terdeteksi positif sebesar 0,282 sedangkan pada *class true* sebesar 0,279. Untuk perhitungan lain yang lebih jelas dapat dilihat pada tabel 8.

Berikut adalah grafik dari ROC menggunakan RUS + Random Forest dari masing masing kelas:



Sumber: (Ginabila & Fauzi, 2019)

Gambar 8. RUS + Random Forest ROC Class False & Class True

Kurva ROC Gambar 8 diatas menunjukkan bahwa ROC berada pada true positive sebesar 0,790 pada *class false* dan *class true* setelah digunakannya pendekatan level data *Random Undersampling* dengan algoritma Random Forest.

Setelah melalui perhitungan yang dilakukan, dapat diketahui nilai akurasi dari masing masing algoritma setelah menggunakan pendekatan level data Random Undersampling adalah sebagai berikut:

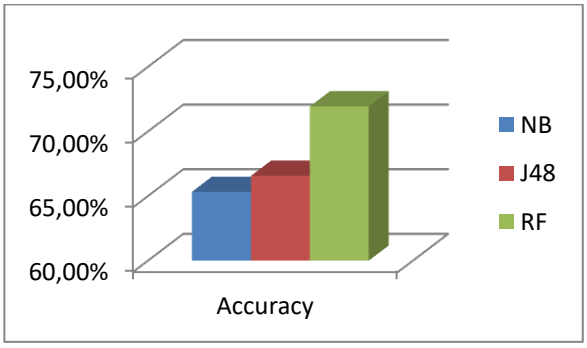
Tabel 9. Hasil Perhitungan Akurasi RUS + NB, J48 dan RF

	NB	J48	RF
Correctly Classified Instances	65,337%	66,564%	71,932%
Incorrectly Classified Instances	34,663%	33,435%	28,067%

Sumber: (Ginabila & Fauzi, 2019)

Dapat dilihat dari tabel 9 diatas bahwa setelah digunakannya pendekatan level data *Random Undersampling* untuk menangani ketidakseimbangan kelas, tingkat akurasi yang didapatkan dari masing-masing algoritma menjadi 65,337% untuk penggunaan algoritma Naive Bayes, 66,564% dengan Algoritma J48 dan 71,932% dengan menggunakan Algoritma Random Forest.

Berikut adalah Gambar 9 grafik perbandingan tingkat akurasi setelah digunakan pendekatan level data Random Undersampling untuk mengatasi ketidakseimbangan kelas dari masing-masing algoritma:



Sumber: (Ginabila & Fauzi, 2019)

Gambar 9. Perbandingan Tingkat Akurasi dari RUS + Ketiga Algoritma

Dari grafik perbandingan tingkat akurasi Gambar 9 diatas setelah ditangani ketidakseimbangan kelasnya dapat dilihat bahwa penggunaan Algoritma Random Forest masih paling tinggi tingkat akurasinya dibandingkan algoritma lainnya.

KESIMPULAN

Hasil dari komparasi algoritma yang diambil oleh peneliti yaitu Naive Bayes, J48 dan Random Forest dapat ditemukan bahwa untuk mengatasi ketidaksimbangan kelas pada dataset kecacatan pada *software*, tingkat akurasi yang didapatkan dengan digunakannya pendekatan level data *Random Undersampling* yaitu sebesar 65,337% dengan menggunakan algoritma Naive Bayes, 66,564% dengan menggunakan Algoritma J48 dan 71,932% dengan menggunakan Algoritma Random Forest. Oleh sebab itu hasil penelitian dapat disimpulkan bahwa model pendekatan level data *Random Undersampling* + algoritma Random Forest adalah yang terbaik untuk digunakan untuk memprediksi cacat pada *software*.

REFERENSI

Akbar, M. S., & Rochimah, S. (2017). Prediksi Cacat Perangkat Lunak Dengan Optimasi Naive Bayes Menggunakan Gain Ratio. *Jurnal Sistem Dan Informatika*, *11*, 147–155.

Andri, Kunang, Y. N., & Murniati, S. (2013). Implementasi Teknik Data Mining Untuk Memprediksi Tingkat Kelulusan Mahasiswa pada Universitas Bina Darma Palembang, *2013*(June 2016), 1–8. https://doi.org/10.13140/RG.2.1.4212.1845

Aries, S., & Wahono, R. S. (2015). Pendekatan Level Data untuk Menangani Ketidakseimbangan Kelas pada Prediksi Cacat Software. *Journal of Software Engineering*, *1*(2), 76–85. https://doi.org/10.1016/S1896-1126(14)00030-3

Diwandari, S., & Setiawan, N. A. (2015). Perbandingan Algoritme J48 dan Nbtree Untuk Klasifikasi Diagnosa Penyakit Pada Soybean. *Seminar Nasional Teknologi Informasi Dan Komunikasi*, *2015*(Sentika), 205–212.

Frank, E., Hall, M., Trigg, L., Holmes, G., & Witten, I. H. (2004). Data Mining in Bioinformatics using Weka. *Bioinformatics*, *20*(15), 2479–2481. https://doi.org/10.1093/bioinformatics/bth 261

Frastian, N., Hendrian, S., & Valentino, V. H. (2018). Komparasi Algoritma Klasifikasi Menentukan Kelulusan Mata Kuliah Pada Universitas. *Faktor Exacta*, *11*(1), 66. https://doi.org/10.30998/faktorexacta.v11i1.1826

Okutan, A., & Yildiz, O. T. (2014). Software Defect Prediction using Bayesian Networks. *Empirical Software Engineering*, *19*(1), 154–181. https://doi.org/10.1007/s10664-012-9218-8

PROMISE. (2010). Data sets for software defect prediction. Retrieved from http://tunedit.org/repo/PROMISE/DefectPrediction

Putra, D. S., Wibawa, A. D., & Purnomo, M. H. (2016). Berjalan Menggunakan Random Forest, *1*(1), 51–56.

Putri, S. A., & Frieyadie. (2017). Combining Integreted Sampling Technique With Feature Selection For Software Defect Prediction. In *2017 5th International Conference on Cyber and IT Service Management (CITSM)* (pp. 1–6). Bali: IEEE. https://doi.org/10.1109/CITSM.2017.8089264

Shuo Wang, & Xin Yao. (2013). Using Class Imbalance Learning for Software Defect Prediction. *IEEE Transactions on Reliability*, *62*(2), 434–443. https://doi.org/10.1109/tr.2013.2259203

Siringoringo, R. (2017). Integrasi Metode Resampling dan K-Nearest Naighbor pada Prediksi Cacat Software Aplikasi Android. *ISD*, *2 No.1*(1), 47–58.

Tharwat, A. (2018). Classification assessment methods. *Applied Computing and Informatics*. https://doi.org/10.1016/j.aci.2018.08.003