TESIS – SS14 2501

PERBANDINGAN METODE *ENSEMBLE RANDOM FOREST* DENGAN *SMOTE-BOOSTING* DAN *SMOTE-BAGGING* PADA KLASIFIKASI DATA MINING UNTUK KELAS IMBALANCE

(Studi Kasus : Data Beasiswa Bidikmisi Tahun 2017 di Jawa Timur)

Sinta Septi Pangastuti
NRP. 06211650010038

DOSEN PEMBIMBING
Dr. Kartika Fithriasari, M.Si
Prof. Drs. Nur Iriawan, MIKomp., Ph.D.

PROGRAM MAGISTER
DEPARTEMEN STATISTIKA
FAKULTAS MATEMATIKA, KOMPUTASI, DAN SAINS DATA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018

THESIS – SS14 2501

A COMPARISON OF THE ENSEMBLE RANDOM FOREST METHODS WITH SMOTE-BOOSTING AND SMOTE-BAGGING ON DATA MINING CLASSIFICATION FOR IMBALANCE CLASS

(Case Study : Data of Bidikmisi Scholarship Year 2017 in East Java)

Sinta Septi Pangastuti
NRP. 06211650010038

SUPERVISOR
Dr. Kartika Fithriasari, M.Si
Prof. Drs. Nur Iriawan, MIKomp., Ph.D.

MAGISTER PROGRAMME
DEPARTMENT OF STATISTICS
FACULTY OF MATHEMATICS, COMPUTING, AND DATA SCIENCE
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018

PERBANDINGAN METODE *ENSEMBLE RANDOM FOREST* DENGAN *SMOTE-BOOSTING* DAN *SMOTE-BAGGING* PADA KLASIFIKASI DATA MINING UNTUK KELAS IMBALANCE
(Studi Kasus : Data Beasiswa Bidikmisi Tahun 2017 di Jawa Timur)

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Master Sains (M.Si)
di
Institut Teknologi Sepuluh Nopember

oleh :
SINTA SEPTI PANGASTUTI
NRP. 06211650010038

Tanggal Ujian : 11 Juli 2018
Periode Wisuda : September 2018

Disetujui oleh:

1. Dr. Kartika Fithriasari, M.Si **(Pembimbing I)**
   NIP: 19691212 199303 2 002

2. Prof. Drs. Nur Iriawan, MIKomp., Ph.D. **(Pembimbing II)**
   NIP: 19621015 198803 1 002

3. Irhamah, M.Si., Ph.D. **(Penguji)**
   NIP: 19780406 200112 2 002

4. Dr. Dra. Ismaini Zain, M.Si **(Penguji)**
   NIP: 19600525 198803 2 001

Dekan
Fakultas Matematika, Komputasi dan Sains Data
Institut Teknologi Sepuluh Nopember

Prof. Dr. Basuki Widodo, M.Sc
NIP: 19650605 198903 1 002

PERBANDINGAN METODE *ENSEMBLE RANDOM FOREST* DENGAN *SMOTE-BOOSTING* DAN *SMOTE-BAGGING* PADA KLASIFIKASI DATA MINING UNTUK KELAS IMBALANCE
(Studi Kasus : Data Beasiswa Bidikmisi Tahun 2017 di Jawa Timur)

Nama Mahasiswa	**: Sinta Septi Pangastuti**
NRP	**: 06211650010038**
Pembimbing	**: Dr. Kartika Fithriasari, M.Si**
Co-Pembimbing	**: Prof. Drs. Nur Iriawan, MIKomp., Ph.D.**

ABSTRAK

Teknik data mining dalam bidang pendidikan mulai berkembang, seiring dengan berkembangnya teknologi dan besarnya data yang dapat disimpan dalam sistem penyimpanan *database* pendidikan. Metode klasifikasi digunakan untuk mengelompokkan siswa kedalam kelas yang teridentifikasi. Namun dalam teknik klasifikasi kondisi *imbalance class* sering terjadi dan menjadi masalah, karena mesin klasifikasi akan condong memprediksi ke kelas mayoritas (kelas negatif) dibandingkan kelas minoritas (kelas positif). Hampir semua *classifier* termasuk *random forest* mengasumsikan sebuah pembagian yang rata antar kelas-kelas pengamatan. *Random forest* umumnya menunjukkan peningkatan kinerja yang besar dibandingkan CART dan C4.5 dan menghasilkan tingkat kesalahan generalisasi yang lebih baik dibandingkan dengan AdaBoost, dan lebih *robust* terhadap *noise*. Namun, seperti kebanyakan metode klasifikasi lainnya, *random forest* juga bisa menghasilkan hasil kurang optimal pada dataset yang *imbalance*. Alternatif lain dalam meningkatkan akurasi kelas *imbalance* adalah dengan menggunakan metode *ensemble*. Salah satu metode yang popular digunakan akhir-akhir ini yaitu SMOTE-Boosting dan SMOTE-Bagging yang mengkombinasikan algoritma pada level data yaitu SMOTE dengan metode *ensemble*. Data Bidikmisi yang digunakan dalam penelitian mempunyai 10 variabel nominal (X_1-X_{10}) dan 1 variabel rasio (X_{11}). Berdasarkan kriteria performansi *g-mean* dan AUC dari kelas (Y) yang *imbalance* menunjukkan bahwa algoritma *ensemble* SMOTE-Bagging (*g-mean*=33,13% dan AUC=52,12%) dan SMOTE-Boosting (*g-mean*=30,22% dan AUC=50,76%) menunjukkan ketepatan klasifikasi yang cenderung lebih baik dibandingkan metode AdaBoost.M2 (*g-mean*=9,03% dan AUC=50,26%). Selisih antara kedua metode algoritma SMOTE-Boosting dan SMOTE-Bagging sangat kecil. Bisa dikatakan bahwa kedua metode tersebut cukup berhasil mengambil keuntungan dari dua algoritma *boosting* dan *bagging* dengan SMOTE. Ketika *boosting* dan *bagging* mempengaruhi akurasi dari *random forest* dengan berfokus pada semua kelas data, algoritma SMOTE merubah nilai performansi dari *random forest* hanya pada kelas minoritas.

Kata Kunci: *AUC, Bagging, Beasiswa Bidikmisi, Boosting, Data mining, G-Mean, Imbalance Data, Random Forest, SMOTE*

(halaman ini sengaja dikosongkan)

A COMPARISON OF THE ENSEMBLE RANDOM FOREST METHODS WITH SMOTE-BOOSTING AND SMOTE-BAGGING ON DATA MINING CLASSIFICATION FOR IMBALANCE CLASS
(Case Study : Data of Bidikmisi Scholarship Year 2017 in East Java)

Name	**: Sinta Septi Pangastuti**
Student Identity Number	**: 06211650010038**
Supervisor	**: Dr. Kartika Fithriasari, M.Si**
Co Supervisor	**: Prof. Drs. Nur Iriawan, MIKomp., Ph.D.**

ABSTRACT

Data mining techniques in the field of education began to grow, along with the development of technology and the amount of data that can be stored in the database storage system of education. Classification methods are used to group students into identified classes. However, in classification techniques the imbalance class condition often occurs and becomes a problem. In the imbalanced classification, the training data set as one majority class could be far surpassed the training dataset as the minority class. This became a problem because classification will tend to predict the data come from the majority class (negative class) compared to the minority class (positive class). Almost all classifiers including random forest assume an equitable division between observation classes. Random forest generally shows a large performance increase compared to CART and C4.5 and results in a better generalization error rate compared to AdaBoost, and is more robust to noise. However, like most other classification methods, random forest can also produce less than optimal results on the imbalanced dataset. Another alternative in improving the accuracy of the imbalance class is by using the ensemble method. One popular method used recently is SMOTE-Boosting and SMOTE-Bagging that combine algorithms at data level ie SMOTE with ensemble method. Bidikmisi data used in the study have 10 nominal variables (X_1-X_{10}) and 1 ratio variable (X_{11}). Based on the performance criteria of g-mean and AUC of class (Y) the imbalance shows that the ensemble algorithm SMOTE-Bagging (g-mean = 33.13% and AUC = 52.12%) and SMOTE-Boosting (g-mean = 30, 22% and AUC = 50.76%) showed better classification accuracy than the AdaBoost.M2 method (g-mean = 9.03% and AUC = 50.26%). The difference between the two SMOTE-Boosting and SMOTE-Bagging algorithms is very small. It can be said that both methods are quite successful to take advantage of two boosting and bagging algorithms with SMOTE. When boosting and bagging affect the accuracy of random forest by focusing on all data classes, the SMOTE algorithm alters the performance values of random forest only in minority classes.

Keywords : *AUC, Bagging, Bidikmisi Scholarship, Boosting, Data Mining, g-Mean, Imbalanced Data, Random Forest, SMOTE*

(halaman ini sengaja dikosongkan)

KATA PENGANTAR

Syukur Alhamdulillah penulis panjatkan kehadirat Allah SWT yang maha menguasai segala ilmu dan alam. Atas rahmat, ridho dan hidayah-Nya sehingga pengerjaan serta penulisan Tesis dengan judul "***Perbandingan Metode Ensemble Random Forest* dengan *SMOTE-Boosting* dan SMOTE-Bagging pada *Klasifikasi Data Mining untuk Kelas Imbalance (Studi Kasus : Data Beasiswa Bidikmisi Tahun 2017 di Jawa Timur)***" dapat terselesaikan dengan baik dan lancar.

Penulisan Tesis ini adalah salah satu syarat yang harus dipenuhi dalam memperoleh gelar Magister sesuai dengan kurikulum Departemen Statistika FMKSD-ITS Surabaya. Dalam penyelesaian Tesis serta laporan ini penulis tidak terlepas dari bantuan serta dukungan dari berbagai pihak. Oleh karena itu penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. My support system yaitu Ibuku, Nenek, Kakek dan keluarga besar penulis serta keluarga baru yaitu Mas Ridho, Umak, Bapak dan Adik-Adik atas segala doa, dukungan materi, motivasi, kepercayaan dan rasa kasih sayang.
2. Bapak Dr. Suhartono, M.Sc. selaku Ketua Departemen Statistika ITS yang telah banyak memberikan inspirasi kepada mahasiswa untuk senantiasa berkarya.
3. Ibu Dr. Kartika Fithriasari, M.Si dan Prof. Drs. Nur iriawan, MIKomp., Ph.D. selaku dosen pembimbing yang dengan sabar memberikan bimbingan, arahan, dan masukan selama pengerjaan Tesis.
4. Ibu Irhamah, M.Si., Ph.D. dan Ibu Dr. Drs. Ismaini Zain, M.Si selaku dosen penguji yang telah memberikan banyak tambahan ilmu selama proses perbaikan laporan Tesis.
5. Bapak Dr. rer.pol. Heri Kuswanto, M.Si selaku Ketua Program Studi Pascasarjana Statistika ITS yang memberikan motivasi dalam pendidikan.
6. Seluruh dosen pengajar serta karyawan di departemen Statistika ITS, yang telah memberikan bantuan dan ilmunya sebagai bekal dalam pengerjaan Tesis.

7. Teman-teman S2 Statistika ITS angkatan 2016, khususnya teman seperjuangan Mbak Laila, Mbak Nita, Farida, Uni Nendy, Febrian dkk yang telah sama-sama membantu dalam penyelesaian laporan.
8. Pihak-pihak lain yang telah mendukung dan membantu dalam penyusunan Tesis ini yang tidak mungkin penulis sebutkan satu per satu. Terima kasih.

Penulis menyadari bahwa penyusunan Tesis ini masih jauh dari sempurna, maka kritik dan saran yang membangun akan senantiasa penulis harapkan demi kesempurnaan di masa mendatang. Semoga laporan ini dapat memberikan sumbangan yang bermanfaat bagi semua pihak.

Surabaya, Juli 2018

Penulis

DAFTAR ISI

Halaman

(*Halaman ini sengaja dikosongkan*)

DAFTAR TABEL

Halaman

(halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dewasa ini, seiring pertumbuhan populasi, teknologi, dan globalisasi, system penyimpanan *database* pendidikan mulai menjadi salah satu bagian kemajuan teknologi. Sejumlah besar data pendidikan sekarang bisa disimpan, dianalisis dan dibagikan dengan mudah. Dalam studi mengenai data pendidikan dengan jumlah dataset yang besar, data mining merupakan metode yang sering digunakan untuk mengetahui hubungan tersembunyi antar variabel (Han dan Kamber, 2006). Salah satu metode data mining yang popular adalah klasifikasi, dimana klasifikasi merupakan pendekatan *supervised* yang mengelompokkan siswa ke dalam kelas yang telah diketahui. Teknik klasifikasi mempunyai tujuan menemukan suatu fungsi keputusan yang secara akurat memprediksi kelas dari data *testing* yang berasal dari fungsi distribusi yang sama dengan data untuk *training*. Oleh karena itu terdapat dua kondisi himpunan kelas data, yaitu *balance* dan *imbalance* kelas data. *Imbalance class* terjadi ketika satu kelas melebihi jumlah kelas lainnya. Kelas data banyak disebut kelas mayoritas (kelas negatif) sedangkan kelas data sedikit disebut kelas minoritas (kelas positif). Dalam kondisi seperti itu sebagian besar *classifier* bias terhadap kelas mayor, karena mesin klasifikasi akan condong memprediksi ke kelas mayor dan mengabaikan kelas minor (Japkowicz dan Stephen, 2002). Sebagai ilustrasi, misalkan sebuah data dengan *imbalance ratio* 1:100 (untuk masing-masing contoh kelas positif, terdapat 100 contoh kelas negative). Sebuah *classifier* yang digunakan untuk melihat akurasi akan mencapai nilai 99% dengan mengabaikan kelas positif, dimana semua data dianggap kelas negative. Belakangan ini permasalahan kelas *imbalance* telah dikenali sebagai masalah yang krusial dalam *machine learning* dan *data mining*. Salah satu contoh masalah kelas *imbalance* yaitu pada masalah klasifikasi yang menggunakan data pendidikan (Chau dan Phung, 2015).

Hampir semua *classifier* termasuk *random forest* mengasumsikan sebuah pembagian yang rata antar kelas-kelas pengamatan. *Random forest* umumnya

menunjukkan peningkatan kinerja yang besar dibandingkan CART dan C4.5. Hal ini menghasilkan tingkat kesalahan generalisasi yang lebih baik dibandingkan dengan AdaBoost, dan lebih *robust* terhadap *noise*. Namun, seperti kebanyakan metode klasifikasi lainnya, *random forest* juga bisa menghasilkan hasil kurang optimal pada dataset yang *imbalance*.

Melihat pentingnya permasalahan kelas *imbalance*, berbagai macam teknik telah dikembangkan untuk mengatasinya. Teknik tersebut bisa dikategorikan ke dalam tiga pendekatan, berdasarkan bagaimana cara mereka mengatasi masalah *imbalance data*. Pendekatan pada level algoritma (*internal*) dengan membuat atau memodifikasi sebuah algoritma, untuk memperhitungkan signifikansi dari kelas positif. Pendekatan algoritma mencakup metode *cost-sensitive* dan pendekatan berbasis pengenalan, pembelajaran berdasarkan kernel, seperti *support vector macine* (SVM) dan *radial basis function* (Nitesh *et al*, 2004). Menerapkan algoritma saja bukanlah ide yang bagus karena ukuran data dan rasio *imbalance class* tinggi sehingga munculah teknik baru yaitu kombinasi metode sampling dengan aloritma yang digunakan (Liu *et al*, 2010). Pendekatan level data (*external*) menambahkan langkah *preprocessing*, dimana distribusi data diseimbangkan kembali untuk mengurangi efek distribusi kelas mayoritas dalam proses *learning*. Prinsip kerja pendekatan berbasis *sampling* dalam penanganan kelas *imbalance* yaitu memodifikasi distribusi data *training* sehingga kedua kelas data baik kelas positif (minor) dan kelas negatif (mayor) dipresentasikan dengan baik (seimbang) di dalam data *training*. Secara umum pendekatan berbasis *sampling* dibagi menjadi dua yaitu metode *oversampling* dan *undersampling*. Metode *undersampling* menyeimbangkan data dengan cara menghapus beberapa pengamatan pada kelas mayor hingga keseimbangan data *training* yang diinginkan tercapai. Namun kelemahan metode ini yaitu kemungkinan hilangnya informasi yang berasosiasi dengan penghapusan pengamatan dari data *training* (Batista, 2004). Pendekatan berbasis *sampling* yang kedua yaitu *oversampling*. Metode ini bekerja untuk menyeimbangkan data *training* dengan cara meningkatkan jumlah data pada kelas minor. Salah satu metode *oversampling* yang paling efektif yaitu SMOTE (*Synthetic Minority Oversampling Technique*).

Salah satu alternatif lain dalam meningkatkan akurasi kelas *imbalance* adalah dengan menggunakan metode *ensemble*. Metode *ensemble* pada prinsipnya mengkombinasikan sekumpulan *classifier* yang dilatih dengan tujuan untuk membuat model klasifikasi (*classifier*) campuran yang terimprovisasi sehingga membuat *classifier ensemble* yang terbentuk lebih akurat dari pada *classifier* asalnya dalam melakukan suatu pengklasifikasian (Han dkk, 2012). *Boosting* (Freud dan Schapire, 1997) dan *Bagging* (Breiman, 1996) adalah metode yang paling popular digunakan. *Boosting* dan *Bagging* adalah salah satu metode *ensemble* yang berbasis variasi data *ensemble*, yang terdiri dari memanipulasi data *training* sedemikian rupa sehingga masing-masing *classifier* dilatih dengan data *training* yang berbeda. Metode *Bagging* didasarkan pada gagasan membuat berbagai sampel dari data *training*. Untuk variasi dari data *training* akan dihasilkan model klasifikasi tertentu, kemudian hasilnya akan diberikan sebagai kombinasi atau gabungan model.

Pada prinsipnya *Boosting* membentuk satu *classifier* yang kuat dengan mengkombinasikan sekumpulan *classifier*. *Boosting* mempertahankan sekumpulan bobot pengamatan pada saat *training* pengamatan dan secara adaptif menyesuaikan (*updating*) bobot-bobot ini pada akhir tiap iterasi *boosting*. Bobot-bobot dari pengamatan yang salah terklasifikasikan pada saat *training* akan dinaikkan sementara bobot-bobot pengamatan yang terklasifikasikan dengan benar akan diturunkan nilainya (Li dkk, 2008). Dengan kata lain, *Boosting* memaksa suatu *classifier* untuk memberi perhatian yang lebih pada pengamatan yang salah diklasifikasikan (Garcia dan Lozano, 2007). Namun, karena desainnya yang berorientasi pada akurasi, algoritma metode *ensemble* yang secara langsung diterapkan ke data yang *imbalance* tidak bisa menyelesaikan masalah pengklasifikasi. Dengan mengkombinasikan *ensemble* dengan teknik lain untuk mengatasi masalah *imbalance* data telah dilakukan dan menghasilkan nilai yang positif (Galar dkk, 2011). Salah satu penelitian mengenai *boosting* dengan *base classifier random forest* dilakukan oleh Mishina dan Tsuchiya (2014).

Secara umum algoritma pada level data dan metode *ensemble* lebih fleksibel karena mereka dapat digunakan secara terpisah dari *base classifier*. Salah satu metode yang popular digunakan akhir-akhir ini yaitu SMOTE-Boosting dan

SMOTE-Bagging yang mengkombinasikan algoritma pada level data yaitu SMOTE dengan metode *ensemble*. SMOTE-Boosting memodifikasi algoritma *Adaptive Boosting* (Freud dan Schapire, 1995) dengan menambahkan algoritma SMOTE di tiap iterasi *boosting*-nya. Penelitian yang dilakukan Chawla (2003), menunjukkan bahwa SMOTE-Boosting memperoleh F-value yang tinggi, dibandingkan dengan standar *boosting* ataupun standar SMOTE. Begitu halnya dengan SMOTE-Bagging yang menambahkan algoritma SMOTE di tiap prosedur *resampling*-nya. Tujuan dari adanya SMOTE yaitu untuk menambah probabilitas terpilihnya sampel-sampel yang sulit diklasifikasikan yang berasal dari kelas minor ke dalam data *training* di tiap iterasi sehingga membuat *base classifier* lebih fokus pada pengamatan kelas minor. Hal ini tentunya akan meningkatkan ketepatan klasifikasi pada kelas minoritas. Kemudian SMOTE yang dikombinasikan dengan prosedur *Bagging* memberikan kinerja keseluruhan (G-Mean) mengalami peningkatan (Wang, 2009).

Pemilihan data untuk penelitian ini adalah dataset pendidikan yaitu data beasiswa Bidikmisi. Tujuan utama penerima beasiswa adalah masyarakat dengan kondisi ekonomi menengah kebawah. Namun, ada indikasi masalah dalam pelaksanaan program beasiswa Bidikmisi, yaitu adanya kondisi penerimaan yang tidak dapat diterima. Status penerima data Bidikmisi tipe biner (0 dan 1), dimana faktor utama penerima beasiswa ketika pendapatan orangtua dibagi dengan jumlah tanggungan tidak lebih dari Rp. 750.000., dianggap tidak mampu. Menurut Iriawan dkk (2018) dengan mempertimbangkan kondisi penerima dikategorikan *"benar"* jika siswa diterima Bidikmisi tersebut memenuhi kategori tidak mampu dan sebaliknya, atau penerima dikategorikan *"salah"* jika siswa diterima Bidikmisi dengan kondisi mampu dan sebaliknya. Berdasarkan alasan tersebut diatas, peneliti memilih menggunakan algoritma *boosting* dan *bagging* dalam melakukan klasifikasi data Bidikmisi yang *imbalance*. Dua algoritma *boosting* yang digunakan yakni AdaBoost dan SMOTE-Boosting ditambah satu metode yaitu SMOTE-Bagging. Sebelumnya dilakukan optimasi parameter untuk klasifikasi *random forest* yang kemudian digunakan sebagai *base classifier* model *boosting* dan *bagging*.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang, permasalahan yang akan dibahas adalah akan dicari terlebih dahulu bagaimanakah parameter yang optimal untuk *random forest* yang dijadikan sebagai *base classifier* model. Kemudian bagaimana penerapan dan hasil kinerja metode algoritma SMOTE-Boosting *Random Forest*, SMOTE-Bagging *Random Forest* dan AdaBoost *Random Forest* untuk pengklasifikasian *imbalance* data Beasiswa Bidikmisi di Jawa Timur tahun 2017.

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian ini adalah sebagai berikut.

1. Bagaimana nilai parameter yang optimum untuk algoritma *random forest* yang akan digunakan sebagai *base classifier* dalam pengklasifikasian *imbalance* data Beasiswa Bidikmisi di Jawa Timur tahun 2017.
2. Bagaimana perbandingan kinerja (*performance*) metode metode *ensemble* AdaBoost, SMOTE-Boosting dan SMOTE-Bagging dalam pengklasifikasian *imbalance* data Beasiswa Bidikmisi di Jawa Timur tahun 2017.

1.4 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah sebagai berikut.

1. Mengetahui nilai parameter *random forest* yang optimum dalam pengklasifikasian *imbalance* data Beasiswa Bidikmisi di Jawa Timur tahun 2017.
2. Memberikan informasi mengenai penerapan metode *ensemble* dengan teknik *re-sampling* SMOTE guna meningkatkan kinerja metode *classifier random forest* sebagai dasar dalam pengklasifikasian *imbalance* data Beasiswa Bidikmisi di Jawa Timur.
3. Menambah keilmuan statistika di bidang *machine learning* khususnya pada teknik klasifikasi data *imbalance*.

1.5 Batasan Masalah

Ada beberapa poin batasan masalah di dalam penelitian, poin-poin tersebut adalah sebagai berikut:

1. Permasalahan klasifikasi pada penelitian ini adalah permasalahan klasifikasi biner, yaitu terdapat dua kelas klasifikasi (0 dan 1).
2. Metode *random forest* digunakan sebagai *base classifier*.
3. Kombinasi antara metode sampling SMOTE dan *ensemble Boosting* dan *Bagging* digunakan dalam penanganan kelas *imbalance* berdasarkan kelebihan-kelebihan yang telah diuraikan pada latar belakang.
4. Data Bidikmisi yang digunakan merupakan data pada tahun 2013 – 2017, oleh karena itu pemberian judul hanya disebutkan tahun 2017 untuk mempersingkat.

BAB 2
TINJAUAN PUSTAKA

Pada bab ini membahas tentang teori metode-metode yang digunakan dalam penelitian yaitu *Decision Tree* (DT), *Random Forest* (RF), *Synthetic Minority Oversampling Technique* (SMOTE), *Boosting* dan *Bagging*, *k-Fold Cross Validation*, evaluasi kinerja (*performance*) metode dan tinjauan tentang data Beasiswa Bidikmisi di Jawa Timur.

2.1 *Preprocesssing* Data

Tujuan dari *preprocessing* adalah mentransformasikan data input mentah ke dalam format yang sesuai untuk analisis selanjutnya. Langkah-langkah yang terlibat dalam *preprocessing* data meliputi mengabungkan data dari berbagai sumber, membersihkan (*cleaning*) data untuk membuang *noise*, mengisi data *missing value* dan menghilangkan *outlier*, menyeleksi *record* dan fitur yang relevan untuk data mining. Metode yang digunakan untuk *preprocessing* data adalah deteksi *missing value*. *Missing value* merupakan kejadian dimana informasi yang tidak tersedia untuk sebuah subyek (kasus). *Missing* data terjadi karena beberapa sebab antara lain informasi tentang suatu objek tidak diberikan, sulit dicari bahkan memang informasi tersebut tidak ada. *Missing value* menyebabkan adanya sel-sel kosong pada satu atau beberapa variabel. Tahap analisis *Missing Value* ada 2, yang pertama adalah menguji keacakan *Missing Value* yang terjadi dan mengatasinya dengan mengisi *missing* data. Jika persentase data *Missing value* melebihi 30%, maka data boleh dihapus sedangkan jika persentase data *Missing Value* kurang 30%, maka data *missing* diimputasi dengan nilai mean jika data kuantitatif dan modus jika data kualitatif (Hair, 1995).

2.2 Metode *Ensemble*

Metode klasifikasi *ensemble* menggabungkan kumpulan pengklasifikasian untuk menciptakan model komposit tunggal yang memberikan kinerja metode yang lebih baik. Penelitian menunjukkan bahwa prediksi dari model komposit

memberikan hasil yang lebih baik dibandingkan dengan prediksi model tunggal. Penelitian dengan metode *ensemble* menjadi popular semenjak beberapa dekade terakhir. Sejumlah studi eksperimental telah dilakukan dengan *machine learning*, mereka membuktikan bahwa menggabungkan output dari beberapa pengklasifikasian mengurani kesalahan generalisasi (Quinlan, 1996). Selanjutnya akan dijelaskan metode *ensemble* yang akan digunakan dalam penelitian kali ini.

2.2.1 *Synthetic Minority Over-Sampling Technique* (SMOTE)

SMOTE (*Synthetic Minority Oversampling Technique*) merupakan salah satu metode dalam penanggulangan data *imbalance* yang diusulkan oleh Chawla dkk (2002). Ide dasar dari SMOTE yaitu menambah jumlah sampel pada kelas minor agar setara dengan kelas mayor dengan cara membangkitkan data *synthetic* berdasarkan tetangga terdekat *k-nearest neighbour* dimana tetangga terdekat dipilih berdasarkan jarak *euclidean* antara kedua data (Chawla dkk, 2002).

Misalkan diberikan data dengan p variabel yaitu $\mathbf{x}^T = [x_1, x_2, ..., x_p]$ dan $\mathbf{z}^T = [z_1, z_2, ..., z_p]$ maka jarak *euclidean* $d(x,z)$ secara umum sebagai berikut:

$$d(\mathbf{x},\mathbf{z}) = \sqrt{(x_1 - z_1)^2 + (x_2 - z_2)^2 + ... + (x_p - z_p)^2} \quad (2.1)$$

Pembangkitan data *synthetic* dilakukan dengan menggunakan persamaan berikut:

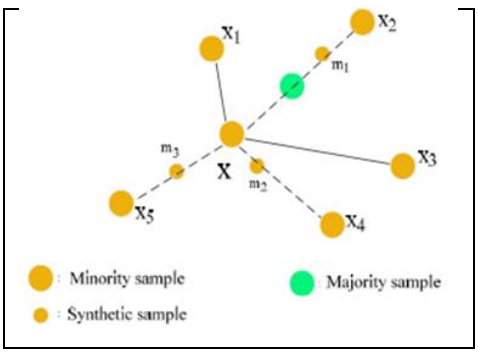
$$\mathbf{x}_{syn} = \mathbf{x}_i + (\mathbf{x}_{knn} - \mathbf{x}_i)\gamma \quad (2.2)$$

Dimana: $\mathbf{x}_{syn}$ adalah data *synthetic*

$\mathbf{x}_i$ adalah data ke-i dari kelas minor

$\mathbf{x}_{knn}$ adalah data dari kelas minor yang memiliki jarak terdekat dari $\mathbf{x}_i$

γ adalah bilangan random antara 0 dan 1



Gambar 2.1 Ilustrasi SMOTE

Nilai X pada Gambar 2.1 menyatakan sebuah sampel dan X_1, X_2, X_3, X_4, dan ,X_5 adalah tetangga terdekat dari sampel X. SMOTE membangkitkan data baru (data *synthetic*) m_1, m_2, dan m_3 melalui sebuah garis diantara X dan masing-masing tetangga terdekatnya.

2.2.2 *Synthethic Minority Over-Sampling Technique – Nominal Continous* (SMOTE-NC)

SMOTE-NC merupakan perluasan atau generalisasi dari metode SMOTE yang digunakan untuk menangani kumpulan data campuran kontinyu dan nominal. Berdasarkan penelitian Chawla dkk (2002) menggunakan data Adult (Blake dan Merz, 1998) dari repository UCI, dataset tersebut mempunyai 6 kontinyu variabel dan 8 nominal variabel. Algoritma SMOTE dan SMOTE-NC digunakan untuk mengevaluasi dataset tersebut. Algoritma SMOTE-NC dijelaskan sebagai berikut:

1. Perhitungan Median: Hitung median standar deviasi semua variabel kontinyu untuk kelas minoritas. Jika variabel nominal berbeda antara sampel dan potensi tetangga terdekatnya, maka median ini termasuk dalam jarak Euclidean. Median digunakan untuk mengesampingkan perbedaan variabel nominal dengan suatu jumlah yang terkait dengan perbedaan khas dalam variabel kontinyu.
2. Perhitungan Tetangga Terdekat: Hitung jarak Euclidean antara variabel vektor yang mana *k* tetangga terdekat yang sedang diidentifikasi (sampel kelas minor) dan variabel vektor lainnya (sampel kelas minoritas) menggunakan ruang variabel kontinyu. Untuk setiap variabel nominal yang berbeda antara variabel vektor yang dipertimbangkan dan calon tetangga terdekatnya, termasuk median standar deviasi yang dihitung sebelumnya, dalam perhitungan jarak Euclidean. Gambar 2.2 menunjukkan contoh perhitungannya.
3. Membuat *Synthetic* Sampel: Variabel kontinyu dari data *synthethic* yang baru untuk kelas minoritas dibuat menggunakan pendekatan SMOTE yang sama seperti penjelasan sebelumnya. Variabel nominal diberikan nilai yang sering terjadi di sebagian besar *k* tetangga terdekat.

Misalkan diberikan sampel untuk menghitung tetangga terdekat

F1 = 1 2 3 A B C

F2 = 4 6 5 A D E

F3 = 3 5 6 A B K

Maka, jarak Euclidean antara F2 dan F1 adalah sebagai berikut:

$$Euclidean = \sqrt{(4-1)^2 + (6-2)^2 + (5-3)^2 + Med^2 + Med^2}$$

Med adalah nilai median dari nilai standar deviasi variabel kontinyu dari kelas minoritas.

Istilah median disebut dua kali untuk variabel nomer 5: B → D, dan nomer 6: C → E yang mana berbeda untuk dua vektor variabel F1 dan F2.

Gambar 2.2 Contoh Perhitungan Tetangga Terdekat untuk SMOTE-NC

2.2.3 Algoritma Boosting

Boosting merupakan salah satu metode *ensemble* yang digunakan untuk mengimprovisasi performa suatu algoritma *learning* dengan mengkombinasikan kumpulan *classifier* lemah guna membentuk suatu *classifier* akhir yang kuat. *Boosting* mengasumsikan ketersediaan dari suatu algoritma *learning* yang lemah (suatu algoritma *learning* yang menghasilkan model *classifier* lemah atau model yang kurang akurat dalam memprediksi suatu *training*). Ide utama didalam proses *boosting* yaitu memilih sekumpulan data *training* (sampel *training*) dengan beberapa cara untuk kemudian dipelajari oleh suatu *base learner*, dimana *base learner* tersebut dipaksa menarik sesuatu yang baru tentang sampel tersebut setiap kali *base learner* itu dipanggil. Proses ini dapat dicapai dengan memilih sampel *training* yang diharapkan dapat membuat performa dari *base classifier* menjadi sangat buruk bahkan lebih buruk dari pada performa *base classifier* secara reguler. Jika hal ini dapat dicapai, kemudian pada iterasi selanjutnya diharapkan *base learner* dapat menghasilkan suatu *base classifier* baru yang secara signifikan berbeda dari pendahulunya. Hal ini dikarenakan meskipun *base learner* diharapkan sebagai sesuatu algoritma *learning* yang lemah dan medioker, namun *base learner* ini diharapkan pula dapat memberikan *output* suatu *classifier* yang membuat

prediksi *nontivial* (Schapire dan Freund, 2012). Algoritma *boosting* yang digunakan dalam penelitian ini adalah Adaptive Boosting M2, yaitu sebuah algoritma perluasan dari algoritma *boosting* original. Selengkapnya akan dijelaskan pada subbab 2.2.3.

2.2.4 Adaptive Boosting M2 (*AdaBoost*.M2)

AdaBoost.M2 adalah boosting yang digunakan untuk menangani masalah multiclass sebagai *base classifier* nya. Algoritma ini mengambil *input* sekumpulan data *training* $(\mathbf{x}_1, y_1),...,(\mathbf{x}_m, y_m)$, dimana setiap $\mathbf{x}_i \in \mathbf{X}$ dan $y_i \in \mathbf{Y} = \{1,...,k\}$. Adaboost memanggil sebuah *base learner* secara berulang dalam suatu urutan iterasi $t = 1,...,T$. Dalam memilih sampel *training* yang akan digunakan oleh *base learner* pada setiap iterasi *boosting* berdasarkan data *training* yang disediakan, *AdaBoost*.M2 menggunakan sebuah distribusi dari sampel *training*. Distribusi yang digunakan pada iterasi ke-*t* dinyatakan dalam D_t, dan bobot untuk sampel *training* ke-*i* dinyatakan dalam $D_t(i)$. Secara intuitif, bobot ini adalah suatu ukuran dari pentingnya sampel ke-*i* terklasifikasikan secara benar pada suatu iterasi. Pada inisialisasi, semua bobot diatur sama besarnya, kemudian pada setiap iterasi bobot-bobot dari sampel yang terklasifikasikan salah akan dinaikan, dengan kata lain sampel yang sulit diklasifikasikan akan memperoleh bobot yang lebih besar sehingga memaksa *base learner* untuk memberikan perhatian lebih terhadap sampel-sampel tersebut. Kebaikan dari suatu *weak hypothesis* diukur berdasarkan *pseudo-loss* berikut.

$$\varepsilon_t = \frac{1}{2} \sum_{(i,y) \in B} D_t(i,y)\left(1 - h_t(x_i, y_i) + h_t(x_i, y_i)\right) \qquad (2.3)$$

Perhatikan bahwa didalam algoritma AdaBoost.M2, masing-masing base classifier harus meminimalkan *pseudo-error* daripada error rate. Jika nilai *pseudo-loss* dibawah 0.5, yang mana lebih mudah dicapai sebagai dasar *weak classifier*, penurunan eksponensial batas atas pada tingkat kesalahan *training set* dijamin.

Berikut algoritma AdaBoost.M2 secara umum:

Input: $(\mathbf{x}_i, y_i),...,(\mathbf{x}_m, y_m)$ dimana $\mathbf{x}_i \in \mathbf{X}$ dan $y_i \in \mathbf{Y} = \{1,...,k\}$

Diberikan: $B = \{(i, y) : i \in \{1,...,m\}, y \neq y_i\}$

Inisialisasi: $D_1(i, y) = 1/|B|$ for $(i, y) \in B$

Untuk $t = 1, 2, ..., T$

1. Latih *weak learner* menggunakan distribusi D_t
2. Dapatkan *weak hypothesis* $h_t : X \times \mathrm{Y} \rightarrow [0,1]$ dengan *pseudo-loss* pada persamaan (2.3). Jika diperoleh nilai $e_t > 0{,}5$, maka proses *learning* berhenti.
3. Hitung $\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$ (2.4)
4. *Update* nilai bobot

$$D_{t+1}(i, y) = \frac{D_t(i, y)}{Z_t} \times \beta_t^{\left(\frac{1}{2}\right)\left(1+h_t(x_i, y_i)-h_t(x_i, y_i)\right)} \quad (2.5)$$

Dimana Z_t adalah sebuah konstanta normalisasi yang membuat $\sum_{i=1}^{m} D_{t+1}(i, y) = 1$

Output dari hipotesis akhir yaitu

$$H(x) = \arg\max_{y \in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t} \right) h_t(x, y) \quad (2.6)$$

Gambar 2.3 Algoritma AdaBoost.M2

AdaBoost.M2 secara linier mengkombinasikan semua *weak classifier* menjadi sebuah hipotesis akhir $H(\mathbf{x})$. Hipotesis akhir $H(\mathbf{x})$ adalah sebuah *majority vote* yang terboboti dari *T weak hypothesis* dimana β_t adalah suatu bobot yang ditugaskan untuk h_t. Nilai $\beta_t \geq 0$ jika $e_t \leq 0.5$ dan β_t akan lebih besar jika e_t lebih kecil (Freund dan Schapire, 1996).

Bagaimanapun algoritma *boosting* seperti AdaBoost.M2 dapat menghasilkan *overfitting* dan masalah generalisasi karena algoritma tersebut

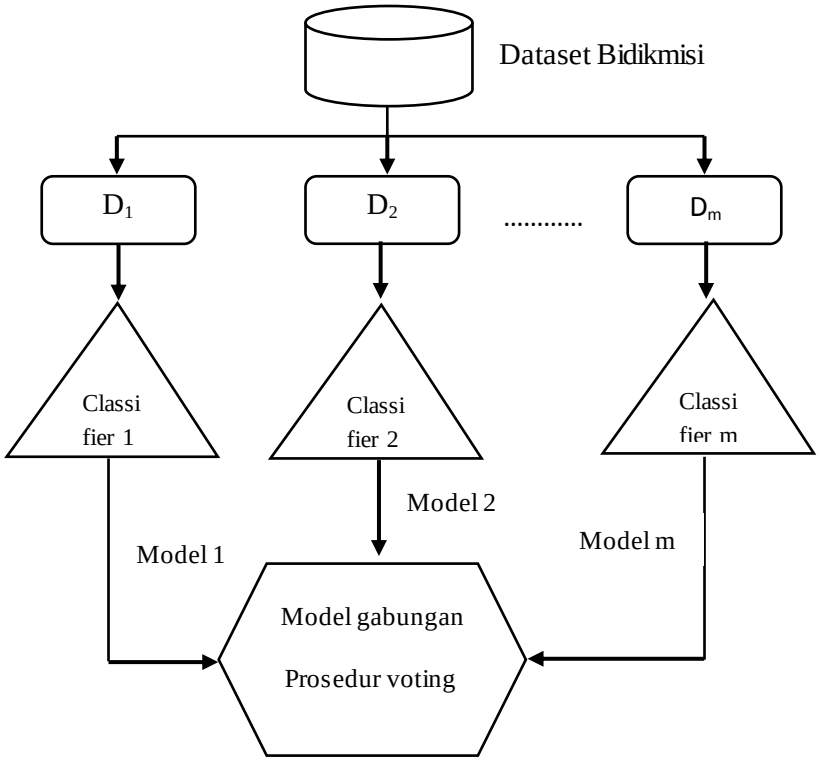
mencoba untuk memaksimalkan akurasi aritmatik. *Error* dari *classifier* e_t dan kinerja metode dari *classifier* β_t diukur berdasarkan rata-rata aritmatik. Suatu ukuran berdasarkan rata-rata aritmatik mungkin menjadi tidak valid sebagai suatu fungsi obyektif yang berguna karena fungsi obyektif yang diukur berdasarkan ukuran aritmatik akan membangkitkan suatu fungsi klasifikasi yang bias terhadap kelas mayor atau kelas dengan *similarity* yang tinggi antar sampel-sampelnya. Khususnya saat suatu algoritma *boosting* diaplikasikan setelah SMOTE dilakukan. Gagasan akurasi geometrik bisa dipertimbangkan untuk meringankan masalah ini (Kim dkk, 2015)

2.2.5 Algoritma Bagging

Bagging adalah metode untuk memodifikasi hasil algoritma klasifikasi dalam *machine learning*. Metode ini diperkenalkan oleh Leo Breiman dan menjadi singkatan dari kata "**b**ootstrap **agg**regat**ing**". Di dalam kasus klasifikasi dengan kemungkinan dua kelas, algoritma klasifikasi membentuk *classifier* H: D → {-1,1} sebagai dasar data *training*. Metode bagging membuat urutan *classifier* H_t, dimana $t=1,...,T$ sebagai modifikasi dari data *training*. *Classifier* tersebut kemudian digabungkan menjadi satu *classifier* gabungan. Hasil prediksi *classifier* gabungan kemudian diberikan sebagai bobot gabungan *classifier* individu atau disebut sebagai prosedur voting.

$$H(d_i) = sign\left(\sum_{t=1}^{T} \alpha_t H_t(d_i)\right) \tag{2.7}$$

Bagging merupakan salah satu metode yang berdasar pada *ensemble method*, secara umum tahap-tahap pada metode bagging disajikan seperti Gambar 2.4. Berdasarkan persamaan 2.7 sebuah contoh d_i diklasifikasikan ke dalam kelas mayoritas voting *classifier*. Parameter α_t, $t=1,...,T$ ditentukan sedemikian rupa sehingga *classifier* yang lebih tepat memiliki pengaruh lebih kuat ketika prediksi akhir daripada *classifier* yang kurang tepat. Ketepatan pengklasifikasian dasar H_t hanya bisa sedikit lebih tinggi daripada ketepatan klasifikasi acak. Itulah mengapa pengklasifikasian H_t disebut pengklasifikasi lemah.



Gambar 2.4 Proses *Bagging*

Berikut algoritma *bagging* secara umum.

1. Inisialisasi data *training D*
2. Untuk $t=1,...,T$
   2.1 Membuat dataset baru D_t dengan ukuran yang sama dengan | D | dengan seleksi secara random dari data *training D* (beberapa *training* data dapat diseleksi secara berulang-ulang dan beberapa tidak dipilih sama sekali)
   2.2 Mendapatkan *classifier* khusus $H_t : D_t \rightarrow R$ dengan algoritma yang telah diberikan berdasarkan data *training* asli D_t
3. Gabungan *classifier H* dibuat sebagai aggregasi pengklasifikasian khusus H_t: $t=1,...,T$ dan sebuah contoh d_i diklasifikasikan ke dalam kelas c_j sesuai dengan jumlah vote yang diperoleh dari pengklasifikasian khusus H_t

$$H\left(d_i, c_j\right) = sign\left(\sum_{t=1}^{T} \alpha_t H_t\left(d_i, c_j\right)\right) \quad (2.8)$$

Gambar 2.5 Algoritma *Bagging*

Menurut Breiman (1996), *bagging* adalah teknik ensemble yang efektif untuk algoritma *learning* yang tidak stabil dimana perubahan kecil dalam kumpulan data *training* menghasilkan perubahan besar dalam prediksi untuk misalnya Pohon Keputusan, Jaringan Saraf, dll.

2.2.6 Algoritma SMOTE-Boosting

Algoritma ini diusulkan oleh Chawla pada tahun 2002. SMOTE-Boosting mengkombinasikan algoritma SMOTE dan prosedur standart *boosting*, dengan memanfaatkan SMOTE untuk meningkatkan prediksi kelas minoritas dan memanfaatkan *boosting* supaya tidak mengorbankan akurasi atas seluruh kumpulan data. Tujuan dari penggabungan algoritma SMOTE dan AdaBoost yaitu untuk meningkatkan nilai True Positive (TP) rate (Chawla, 2002). SMOTE-Boosting sukses mengkombinasikan AdaBoost dan SMOTE. Sementara AdaBoost mencoba meningkatkan akurasi dari *classifier* dengan fokus pada pengamatan yang sulit diklasifikasikan yang berasal dari kedua kelas, SMOTE mencoba meningkatkan kinerja metode dari *classifier* hanya pada pengamatan pada kelas minoritas. Oleh karena itu dalam beberapa iterasi *boosting* berurutan, SMOTE-Boosting mampu membuat daerah keputusan yang lebih luas untuk kelas minoritas dibandingkan metode *boosting* standar. SMOTE-Boosting awalnya menggunakan prosedur iterasi dari *boosting* AdaBoost.M2 oleh (Freund dan Schapire, 1996). Dalam prosedur iterasi AdaBoost.M2, hasil klasifikasi dari komponen *classifier* terlebih dahulu dibawa ke dalam bentuk probabilitas [0,1] guna nantinya dipakai dalam menghitung *pseudo loss*. Algoritma SMOTE-Boosting dengan prosedur iterasi AdaBoost.M2 ditampilkan pada Gambar 2.6

Input: $(\mathbf{x}_i, y_i),...,(\mathbf{x}_m, y_m)$ dimana $\mathbf{x}_i \in \mathbf{X}$ dan $y_i \in \mathbf{Y} = \{1,...,k\}$

Diberikan: $B = \{(i,y) : i \in \{1,...,m\}, y \neq y_i\}$

Inisialisasi: $D_1(i,y) = \frac{1}{|B|}$ for $(i,y) \in B$

Untuk t = 1,...., T

1. Memodifikasi distribusi D_t dengan mereplikasipengamatan kelas minor menggunakan algoritma SMOTE.
2. Latih *weak learner* menggunakan distribusi D_t

3. Dapatkan *weak hypothesis* dengan *pseudo loss* pada persamaan.

$$\varepsilon_t = \sum_{(i,y)\in B} D_t(i,y)(1 - h_t(x_i,y_i) + h_t(x_i,y_i)) \quad (2.9)$$

Jika diperoleh nilai $e_t > 0{,}5$, maka proses *learning* berhenti.

4. Hitung $\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$

5. Update nilai bobot:

$$D_{t+1}(i,y) = \frac{D_t(i,y)}{Z_t} \times \beta_t^{\left(\frac{1}{2}\right)\left(1+h_t(x_i,y_i)-h_t(x_i,y_i)\right)} \quad (2.10)$$

dimana Z_t adalah konstanta normalisasi

6. Output dari hipotesis akhir yaitu

$$H(x) = \arg\max_{y\in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t} \right) h_t(x,y) \quad (2.11)$$

Gambar 2.6 Algoritma SMOTE-Boosting

2.2.7 Algoritma SMOTE-Bagging

SMOTE-Bagging adalah kombinasi dari algoritma SMOTE dan *Bagging*. SMOTE-Bagging melibatkan langkah generasi data *synthetic* selama konstruksi subset (Wang dan Yao, 2009). *Synthetic Minority Oversampling Technique* (SMOTE) adalah salah satu metode *oversampling* yang pertama kali dikenalkan oleh Chawla dkk (2002). Seperti yang telah dijelaskan pada subbab sebelumnya, SMOTE bekerja dengan mengenerasi data *synthetic* sampai banyaknya data minor sama dengan banyaknya data mayor. Berdasarkan SMOTE-Bagging, setiap subset didapatkan melalui proses *bootstrap* diseimbangkan dengan SMOTE sebelum dibentuk model. Data *synthetic* dibuat berdasarkan dua parameter, yaitu jumlah *oversampling* (*N*) dari minoritas kelas dan *k* tetangga terdekat. Total *oversampling* diputuskan sedemikian hingga banyaknya kelas mayor dan kelas minor seimbang. Algoritma SMOTE-Bagging ditampilkan pada Gambar 2.7

1. Inisiasi data *training* D
2. Untuk $t = 1,\ldots, T$
    a. Membuat dataset D_t menggunakan *resample* kelas minor (*N*) dengan pengembalian, dimana *N* adalah kelipatan 100%
    b. Generalisasi data baru dengan SMOTE

c. Mendapatkan *classifier* khusus $H_t : D_t \rightarrow R$ dengan algoritma yang telah diberikan berdasarkan data *training* asli D_t

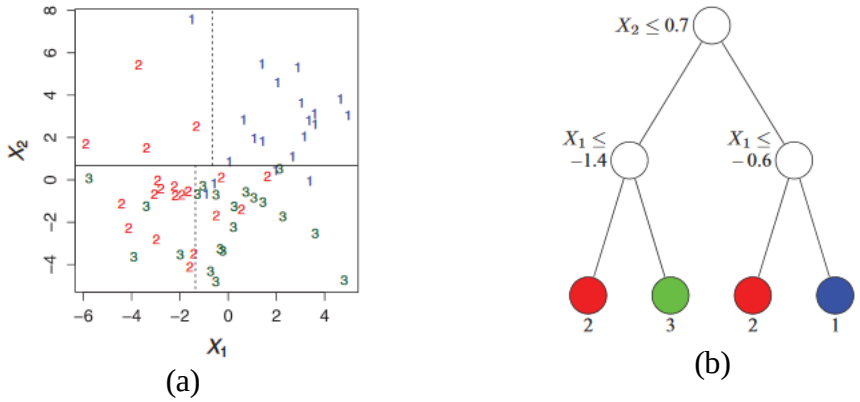
4. Gabungan *classifier* H dibuat sebagai agregasi pengklasifikasian khusus H_t: $t=1,...,T$ dan sebuah contoh d_i diklasifikasikan ke dalam kelas c_j sesuai dengan jumlah vote yang diperoleh dari pengklasifikasian khusus H_t

$$H(d_i, c_j) = sign\left(\sum_{t=1}^{T} \alpha_t H_t(d_i, c_j)\right) \quad (2.12)$$

Gambar 2.7 Algoritma SMOTE-Bagging

2.3 Pohon Klasifikasi atau Pohon Keputusan (*Decision Tree*)

Di dalam sebuah masalah klasifikasi, diberikan *training* sampel dengan n observasi pada sebuah kelas variabel Y dimana $Y=1,2,...,k$ dan sebanyak p variabel prediktor, $X_1,...,X_p$. Tujuannya adalah untuk menemukan model untuk memprediksi nilai Y dari nilai X baru. Secara teori, solusinya hanyalah sebuah partisi dari ruang X ke dalam himpunan yang diuraikan sebanyak k, yaitu $A_1, A_2, ..., A_k$ dimana nilai prediksi Y adalah j jika X adalah milik A_j, untuk $j = 1, 2, ..., k$. Metode pohon keputusan menghasilkan set persegi panjang A_j dengan membagi secara rekursif *dataset* untuk setiap X. Sebagai contoh, pada Gambar 2.8 terdapat tiga kelas dan dua variabel X. Gambar (a) memilah titik data dengan partisinya, dan Gambar (b) menunjukkan pohon keputusan yang sesuai struktur. Keuntungan utama dari pohon keputusan adalah dapat digunakan untuk sembarang variabel, sedangkan plot sebelah kiri terbatas paling banyak dua variabel.



(a) (b)

Gambar 2.8 Ilustrasi Plot Klasifikasi dan Pohon Klasifikasi

Sebuah pohon keputusan adalah *classifier* yang dinyatakan sebagai partisi rekursif dari ruang contoh. Pohon keputusan adalah struktur *flowchart* yang menyerupai *tree* (pohon), dimana setiap simpul internal menandakan suatu tes pada atribut, setiap cabang mempresentasikan hasil tes, dan simpul daun merepresentasikan kelas atau distribusi kelas. Alur pada pohon keputusan di telusuri dari simpul akar ke simpul daun yang memegang prediksi ke aturan klasifikasi (*classification rules*). Konsep pohon keputusan adalah mengubah data menjadi pohon keputusan dan aturan-aturan keputusan. Pada pohon keputusan setiap simpul daun menandai label kelas. Simpul yang bukan simpul akhir terdiri dari akar dan simpul internal yang terdiri dari kondisi tes atribut pada sebagian *record* yang mempunyai karakteristik yang berbeda. Simpul akar dan simpul internal ditandai dengan bentuk oval dan simpul daun ditandai dengan bentuk segi empat (Han, 2006). Bagian-bagian pada pohon keputusan adalah sebagai berikut menurut Cahyono (2010):

a. *Root node*

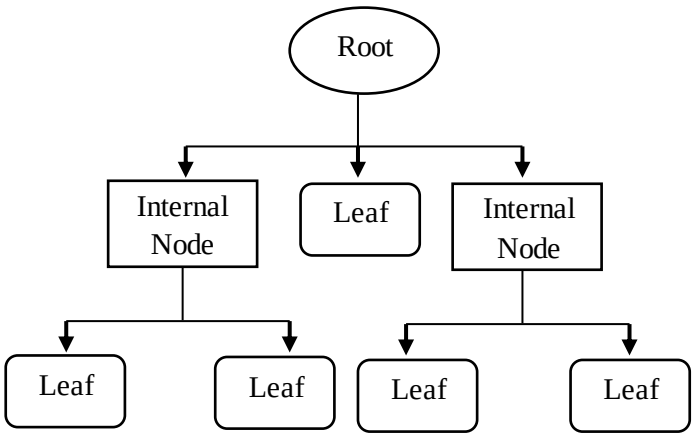
   Node ini merupakan node yang terletak paling atas dari suatu pohon, pada *node* ini tidak ada *input* dan mempunyai *output* lebih dari satu atau bisa juga tidak mempunyai *output*.

b. *Internal node*

   Node ini merupakan *node* percabangan, hanya terdapat satu *input* serta mempunyai minimal dua *output*.

c. *Leaf node*

   Node ini merupakan *node* akhir, hanya memiliki satu *input*, dan tidak memiliki *output*.



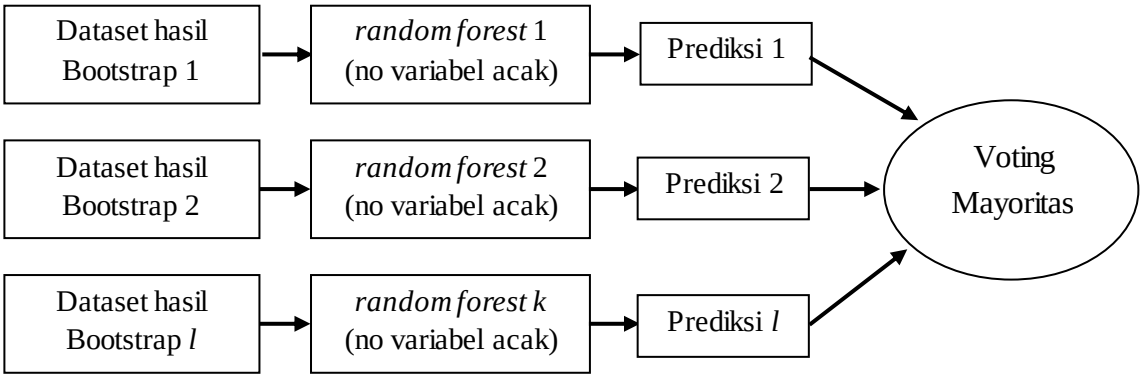
Gambar 2.9 Model Pohon Keputusan untuk Klasifikasi

2.4 Random Forest

Metode *random forest* mulai banyak diperbincangkan sejak tulisan Breiman (2001) muncul pada jurnal Machine Learning. Liaw dan Wiener (2002) menyatakan bahwa *random forest* dikembangkan oleh Breiman (2001). Secara jeli, Breiman (2001) berupaya untuk memperbaiki proses pendugaan yang dilakukan menggunakan metode *bagging*. Metode *random forest* adalah pengembangn dari metode CART, yaitu dengan menerapkan metode *bootstrap aggregating* (*bagging*) dan *random feature selection*. Dalam *random forest*, banyak pohon ditumbuhkan sehingga terbentuk hutan (*forest*), kemudian analisis dilakukan pada kumpulan pohon tersebut. Secara sederhana, algoritma pembentukan *random forest* dapat disebutkan sebagai berikut. Andaikan data training yang kita miliki berukuran *n* dan terdiri atas *p* variabel penjelas (prediktor). Tahapan penyusunan dan pendugaan menggunakan *random forest* adalah:

1. (Tahapan *bootstrap*) tarik sampel acak dengan pengembalian berukuran *n* dari data training.
2. Dengan menggunakan contoh *bootstrap*, pohon dibangun sampai mencapai ukuran maksimum (tanpa pengembalian). Susun pohon berdasarkan data tersebut, namun pada setiap proses pemisahan pilih secara acak $m < p$ peubah penjelas, dan dilakukan pemisahan terbaik (tahapan *random sub-setting*).
3. Ulangi langkah 1-2 sebanyak *l* kali sehingga terbentuk sebuah hutan yang terdiri atas *l* pohon.

4. Lakukan pendugaan gabungan berdasarkan *l* buah pohon tersebut (misal menggunakan *majority vote* untuk kasus klasifikasi atau rata-rata untuk kasus regresi).



Gambar 2.10 Ilustrasi *Random Forest*

Respons suatu amatan diprediksi dengan menghubungkan (*Aggregating*) hasil prediksi *l* pohon. Pada masalah klasifikasi dilakukan berdasarkan *majority vote* (suara terbanyak).

Proses penggabungan nilai dugaan dari banyak pohon yang dihasilkan serupa dengan yang dilakukan pada metode *bagging*. perhatikan bahwa pada setiap kali pembentukan pohon, kandidat peubah penjelas yang digunakan untuk melakukan pemisahan bukanlah seluruh variabel yang terlibat namun hanya sebagian saja hasil pemilihan secara acak. Bisa dibayangkan bahwa proses ini menghasilkan kumpulan pohon tunggal dengan ukuran dan bentuk yang berbeda-beda. Hasil yang diharapkan adalah kumpulan pohon tunggal memiliki korelasi yang kecil antar pohonnya. Korelasi kecil ini mengakibatkan ragam dugaan hasil *random forest* menjadi kecil (Hastie dkk, 2008) dan lebih kecil dibandingkan ragam dugaan hasil bagging (Zhu, 2008).

Error klasifikasi *random forest* diduga melalui error OOB yang diperoleh dengan cara (Breiman 2001; Liaw & Wiener 2002):

1. Lakukan prediksi terhadap setiap data OOB pada pohon yang bersesuaian. Data OOB (*Out of Bag*) adalah data yang tidak termuat dalam contoh *bootstrap*.
2. Secara rata-rata, setiap amatan data asli akan menjadi OOB sebanyak sekitar 36% dari banyak pohon. Oleh karena itu, pada langkah l, masing-masing amatan data asli mengalami prediksi sebanyak sekitar sepertiga kali dari banyaknya pohon. Jika *a* adalah sebuah amatan dari gugus data asli, maka hasil prediksi

random forest terhadap *a* adalah gabungan dari hasil prediksi setiap kali *a* menjadi data OOB.

3. Error OOB dihitung dari proporsi misklasifikasi hasil prediksi *random forest* dari seluruh amatan gugus data asli.

Breiman (2001) manyarankan untuk mengamati error OOB, lalu memilih *m* yang menghasilkan error OOB terkecil. Jika *random forest* dilakukan dengan menghasilkan *variable importance,* disarankan untuk menggunakan banyak pohon, misalnya 1000 pohon atau lebih. Jika peubah penjelas yang dianalisis sangat banyak, nilai tersebut dapat lebih besar agar *variable importance* yang dihasilkan semakin stabil. Jika melihat secara seksama algoritma pembentukan *random forest*, salah satu yang bisa kita ubah adalah nilai *m*, yaitu banyaknya peubah penjelas yang digunakan sebagai kandidat pemisah dalam pembentukan pohon. Nilai *m* yang semakin besar akan menyebabkan korelasi semakin besar. Contoh ekstrim adalah jika kita gunakan *m = p* yang menyebabkan setiap kali pengulangan akan menghasilkan pohon yang sama sehingga nilai korelasi akan menjadi maksimum yaitu sebesar 1. Namun jika, nilai *m* kita buat sekecil mungkin yaitu hanya 1 peubah penjelas saja yang dijadikan kandidat perintah, maka pohon yang diperoleh akan menjadi pohon dengan akurasi yang sangat rendah atau nilai *s* yang kecil.

Dengan demikian jelas bahwa pemilihan *m* memegang peranan dalam menentukan kebaikan *random forest* yang dihasilkan. Nilai *m* yang disarankan adalah saat m $=\left\{\frac{1}{2}\left|\sqrt{p}\right|, \left|\sqrt{p}\right|, 2\left|\sqrt{p}\right|\right\}$ (Breiman dan Cutler, 2003). Salah satu paket yang popular digunakan untuk menghasilkan *random forest* adalah randomForest package di R. Namun demikian Hastie *dkk* (2008) menyarankan pengguna untuk juga mencoba menggunakan nilai *m* karena ada kemungkinan memperoleh *random forest* dengan prediksi yang lebih baik.

2.5 *Stratified* K-*Fold Cross Validation*

Cross validation adalah metode statistik untuk mengevaluasi dan membandingkan algoritma *learning* dengan membagi data menjadi dua bagian yaitu data *training* yang digunakan untuk pelatihan dan data *testing* yang digunakan untuk memvalidasi model. Bentuk dasar *cross-validation* adalah *k-fold cross*

validation. Pada k-*fold cross validation*, data dipartisi secara acak menjadi *k* bagian yang bersifat *mutually exlclusive* $D_1, D_2, ..., D_k$, yang masing masing memiliki ukuran yang sama. Proses *training* dan *testing* dilakukan sebanyak *k* kali. Dalam iterasi ke-*i*, data partisi D_i diposisikan sebagai data *testing*, sementara partisi lain yang tersisa secara kolektif digunakan untuk melatih model. Misalkan pada iterasi pertama, D_1 digunakan sebagai data *testing*, sementara $D_2, D_3, ..., D_k$ digabungkan untuk digunakan sebagai data *training* untuk menghasilkan model, yang kemudian diuji pada data D_1. Pada iterasi kedua, data $D_1, D_3, ..., D_k$ digabungkan untuk digunakan sebagai data *training* dan kemudian model yang dihasilkan dilakukan pengujian menggunakan data D_2. Akurasi klasifikasi model diperoleh dengan cara merata-ratakan akurasi dari setiap iterasi (Han dkk, 2012).

Stratifikasi adalah proses penyusunan ulang data untuk memastikan setiap *fold* merupakan representasi yang baik dari keseluruhan data. Misalnya dalam masalah klasifikasi biner dimana masing-masing kelas terdiri dari 50% data, cara yang terbaik adalah dengan mengatur data sedemikian rupa sehingga dalam setiap *fold*, setiap kelasnya terdapat sekitar setengah sampel. Proses ini diulangi sebanyak K subsets dan hasil akurasi klasifikasi yaitu hasil rata-rata dari setiap data *training* dan *testing*. K-*fold* yang biasa digunakan adalah 3, 5, 10 dan 20 (Bolon dkk, 2015).

2.6 Kriteria Evaluasi Kinerja (*Performance*) Metode Klasifikasi

Data aktual dan data hasil prediksi dari model klasifikasi disajikan dengan menggunakan tabulasi silang (*Confusion matrix*), yang mengandung informasi tentang kelas data yang actual direpresentasikan pada baris matriks dan kelas data hasil prediksi pada kolom (Han dkk, 2006).

Tabel 2.1 Confusion Matrix

	Predictive Positive Class	*Predictive Negative Class*
Real Positive Class	*True Positive* (TP)	*False Negative* (FN)
Real Negative Class	*False Positive* (FP)	*True Negative* (TN)

1. *True Postive* (TP) menunjukan bahwa kelas yang dihasilkan prediksi klasifikasi adalah positif dan kelas sebenarnya adalah positif
2. *True Negatif* (TN) menunjukan bahwa kelas yang dihasilkan dari prediksi klasifikasi adalah negatif dan kelas sebenarnya adalah negatif.
3. *False Positif* (FP) menunjukan bahwa kelas yang dihasilkan dari prediksi klasifikasi adalah negatif dan kelas sebenarnya adalah positif
4. *False Negatif* (FN) menunjukan bahwa kelas yang dihasilkan dari prediksi klasifikasi adalah positif dan kelas sebenarnya adalah negatif.

Didalam kasus kelas *imbalance* dimana kelas mayoritas 98-99% dari keseluruhan populasi, maka hasil klasifikasi akan mencapai akurasi tinggi karena hanya melihat kelas mayoritas saja. Jelas bahwa untuk kasus *imbalance*, akurasi klasifikasi tidak cukup sebagai ukuran kriteria standar. *Area Under Curve* (AUC) dan metric seperti presisi, *recall* dan *f-value* telah digunakan untuk mmahami kinerja algoritma *learning* pada kelas minoritas.

$$Presisi = \frac{TP}{(TP + FP)} \tag{2.13}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{2.14}$$

$$F\text{-}Value = \frac{2(\mathrm{Re}call \times \mathrm{Pr}esisi)}{(\mathrm{Re}call + \mathrm{Pr}esisi)} \tag{2.15}$$

Nilai recall memberikan informasi seberapa banyak kelas minoritas yang diidentifikasi, namun bisa jadi mengorbankan presisi dengan salah klasifikasi kelas mayoritas. Nilai F-Value menggabungkan presisi dan recall, untuk mengukur kebaikan dari algoritma *learning* pada kelas. Untuk melakukan evaluasi kinerja metode secara keseluruhan, dapat digunakan geometric mean (G-*mean*) dan analisis AUC. *Geometric Mean* (*G-mean*) merupakan rata-rata geometrik *Sensitivity* dan *Specificity*. Apabila semua kelas positif tidak dapat diprediksi maka *G-mean* akan bernilai nol sehingga diharapkan suatu algoritma klasifikasi mencapai nilai *G-mean* yang tinggi (Kubat dan Matwin dalam Sain, 1997).

$$Specificity = \frac{TN}{(TN + FP)} \times 100\% \quad (2.16)$$

$$Sensitivity = \frac{TP}{(TP + FN)} \times 100\% \quad (2.17)$$

$$G\text{-}Mean = \sqrt{Sensitivity \times Specificity} \quad (2.18)$$

AUC menyediakan ukuran tunggal kinerja *classifier* untuk evaluasi model mana yang lebih baik secara rata-rata. Ukuran AUC diperoleh dengan menghitung nilai *true posotive rate* (TPR) yaitu jumlah objek pada kelas positif yang diklasifikasikan dengan benar dan *false positive rate* (FPR) yaitu jumlah objek pada kelas positif yang salah diklasifikasikan.

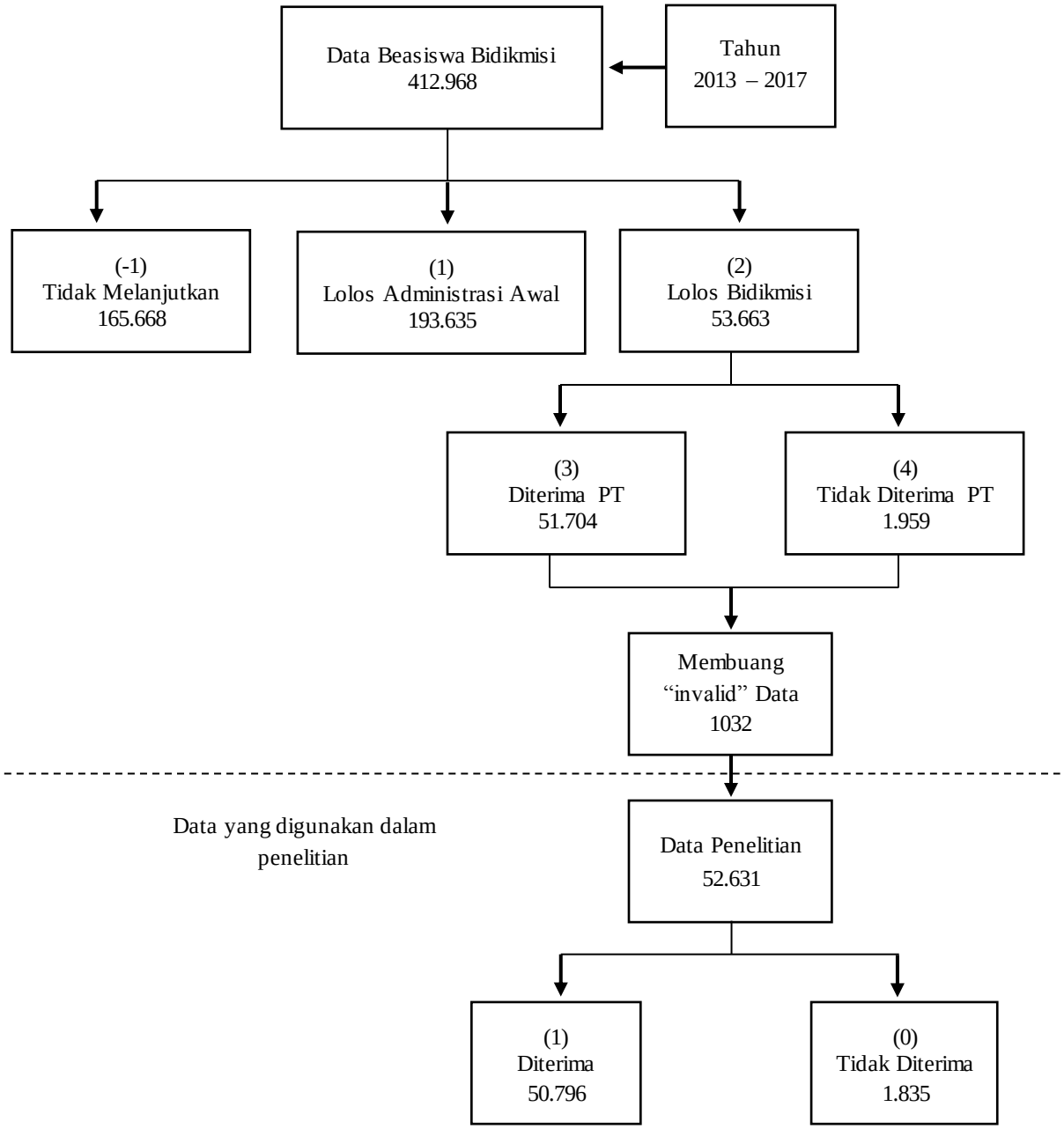
$$TPR = \frac{TP}{(TP + FN)} \quad (2.19)$$

$$FPR = 1 - \frac{TN}{(TN + FP)} \quad (2.20)$$

$$\text{AUC} = \frac{1 + TPR - FPR}{2} \quad (2.21)$$

2.7 Klasifikasi Data Pelamar Bidikmisi

Beasiswa Bidikmisi merupakan bantuan biaya hidup dan biaya pendidikan untuk calon mahasiswa dari keluarga kurang mampu yang berpotensi akademik unggul sebagai jaminan bahwa calon mahasiswa tersebut nantinya dapat menyelesaikan pendidikan di Perguruan Tinggi dengan tepat waktu (Direktorat Jenderal Pembelajaran dan Kemahasiswaan, 2018). Program bantuan biaya pendidikan Bidikmisi diluncurkan mulai tahun 2010 oleh pemerintah melalui Direktorat Jenderal Pendidikan Tinggi Kelembagaan. Program Bidikmisi telah berjalan 8 tahun, jumlah peminatnya menunjukkan peningkatan yang sangat signifikan dari tahun ke tahun.



Gambar 2.11 Proses Memperoleh Data Penelitian (Pelamar Bidikmisi) di Jawa Timur

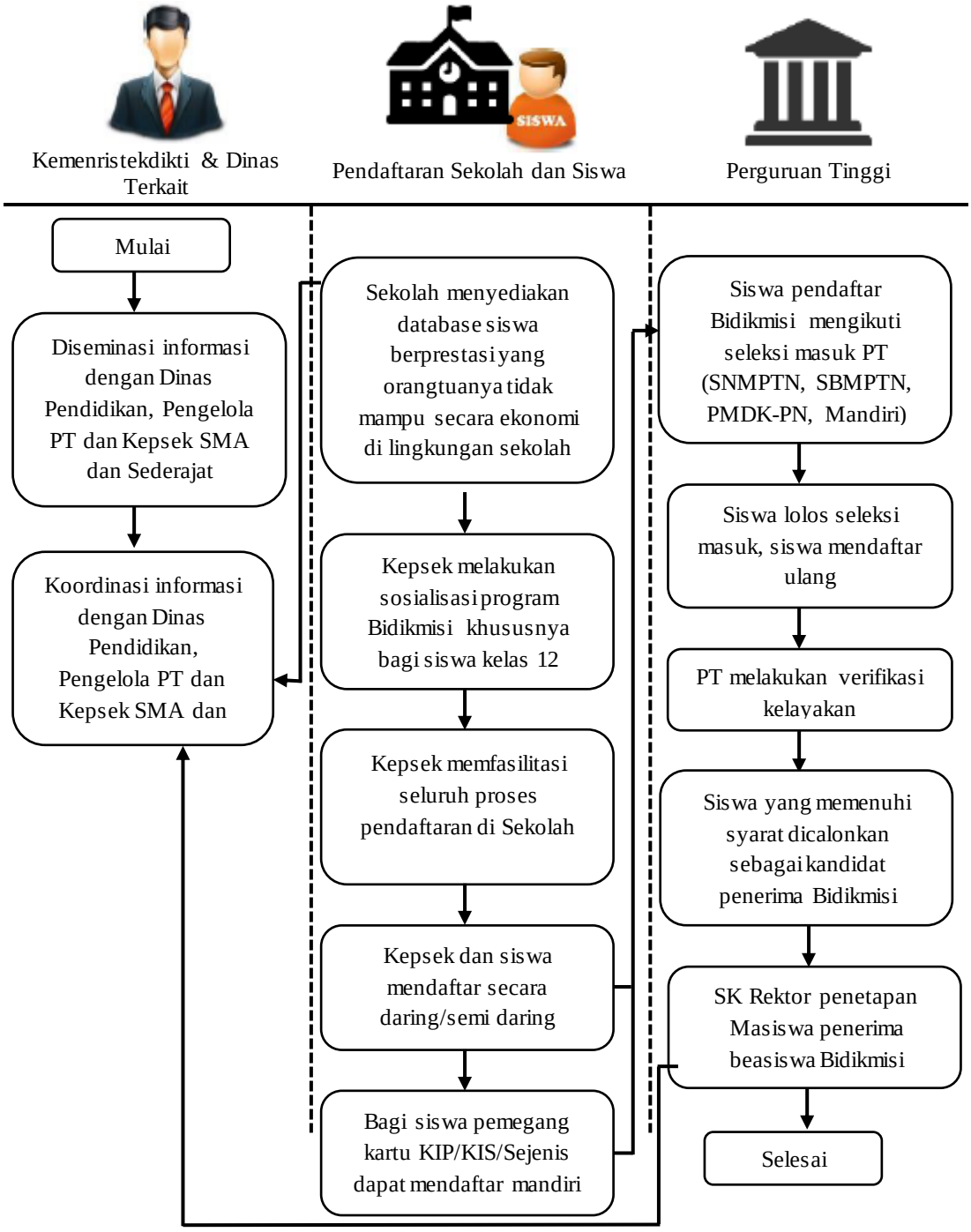
Dalam seleksi pendaftar Bidikmisi, faktor utama yang dilihat adalah pendapatan orangtua dibagi dengan jumlah tanggungan keluarga (*income per capita*) tidak melebihi Rp. 750.000., per orang. Kriteria ini yang kemudian digunakan dalam pengambilan keputusan penerima beasiswa. Data penelitian yang digunakan sebanyak 52.631 pelamar, kemudian yang menerima beasiswa sebanyak 50.796 pelamar, sedangkan yang tidak diterima adalah sebanyak 1.835 pelamar

seperti yang terlihat di Gambar 2.11. Keadaan data yang *imbalance* tersebut mengakibatkan mesin klasifikasi akan condong memprediksi ke kelas mayoritas dibandingkan kelas minoritas sehingga *misclassification rate* nya akan besar.

Prediksi dan pengambilan keputusan sering digunakan dalam aplikasi dunia nyata. Dalam pengambilan keputusan, kita perlu tahu keputusan yang terbaik dengan melihat tingkat kesalahan klasifikasinya (*misclassification rate*). *Misclassification* terjadi ketika suatu obyek ditempatkan pada kelas yang salah dalam suatu populasi atau subgroup karena ada beberapa perhitungan yang salah. Namun, ada indikasi masalah dalam pelaksanaan program beasiswa Bidikmisi, yaitu adanya kondisi penerimaan yang tidak dapat diterima. Kondisi tersebut adalah ketika penerima dikategorikan *"benar"* jika siswa diterima Bidikmisi tersebut memenuhi kategori tidak mampu dan sebaliknya. Kondisi kedua ketika penerima dikategorikan *"salah"* jika siswa diterima Bidikmisi dengan kondisi keluarga yang mampu dan sebaliknya.

2.8 Tahapan Program Bidikmisi

Tahapan pelaksanaan program Bidikmisi disajikan pada Gambar 2.12



Gambar 2.12 Tahapan Program Bidikmisi

(*halaman ini sengaja dikosongkan*)

BAB 3
METODOLOGI PENELITIAN

Pada bab ini diuraikan secara rinci mengenai data dan metode serta langkah-langkah penelitian untuk menyelesaikan permasalahan dalam penelitian pada data Bidikmisi Jawa Timur tahun 2017.

3.1 Sumber Data

Data yang digunakan dalam penelitian ini adalah data sekunder pelamar Beasiswa Bidikmisi Jawa Timur tahun 2017 dari Kementerian Riset, Teknologi, dan Pendidikan Tinggi Republik Indonesia (Kemenristek DIKTI) kanal Bidikmisi, dengan jumlah data yang digunakan pada penelitian ini sebanyak 52631 data individu seluruh kabupaten/kota provinsi Jawa Timur. Program Beasiswa Bidikmisi mengutamakan masyarakat yang tidak mampu atau tergolong masyarakat miskin. Untuk lebih jelasnya variabel apa saja yang digunakan dapat dilihat pada Tabel 3.2

Tabel 3.1 Deskripsi Data

Data	Jumlah Data (N)	Feature (Variabel)	Kategori
Bidikmisi Jawa Timur tahun 2017	52631	11	Diterima = 50796 Tidak diterima = 1835

3.2 Variabel Penelitian

Program Beasiswa Bidikmisi mengutamakan masyarakat yang tidak mampu atau tergolong masyarakat miskin. Ekonomi merupakan faktor utama yang dianggap sesuai dalam mempengaruhi siswa SMA/SMK/MA tidak melanjutkan ke jenjang Perguruan Tinggi. Berikut ini adalah variabel-variabel yang digunakan untuk membentuk model klasifikasi.

Tabel 3.2 Identifikasi Variabel

No	Variabel	Deskripsi	Skala Data	Keterangan
1	*Y*	Status Penerimaan Beasiswa Bidikmisi	1=Diterima 2=Tidak Diterima	Diterima = 50796 Tidak Diterima = 1835
2	X_1	Pekerjaan Ayah	1=Petani, Nelayan, Lainnya 2=TNI/POLRI 3=Wirausaha 4= Peg. Swasta 5= PNS	-
3	X_2	Pekerjaan Ibu	1=Petani, Nelayan, Lainnya 2=TNI/POLRI 3=Wirausaha 4= Peg. Swasta 5= PNS	-
4	X_3	Pendidikan Ayah	1=Tidak Sekolah 2=Pendidikan Dasar (SD/MI dan SMP/MTs) 3=Pendidikan Menengah (SMA/MA) 4=Pendidikan Tinggi (D1,D2/D3,S1/D4)	-
5	X_4	Pendidikan Ibu	1=Tidak Sekolah 2=Pendidikan Dasar (SD/MI dan SMP/MTs) 3=Pendidikan Menengah (SMA/MA) 4=Pendidikan Tinggi (D1,D2/D3,S1/D4)	-
6	X_5	Kepemilikan Rumah Tinggal Keluarga	1= Tidak memiliki rumah 2= Sewa (Tahunan, Bulanan) dan Menumpang 3= Sendiri	-
7	X_6	Luas Tanah Rumah Tinggal Keluarga	1=25-50 m^2 2=50 – 99 m^2 3= > 100 m^2	-
8	X_7	Luas Bangunan Rumah Tinggal Keluarga	1=25-50 m^2 2=50 – 99 m^2 3= > 100 m^2	-
9	X_8	Sumber Listrik yang Digunakan	1=Tidak Ada 2=Genset/Mandiri, Tenaga Surya 3=PLN	-
10	X_9	Sumber Air yang Digunakan Keluarga	1=Sumur, Sungai/Mata Air 2= PDAM, Kemasan	-

Tabel 3.2 Lanjutan Identifikasi Variabel

No	Variabel	Deskripsi	Skala Data	Keterangan
11	X_{10}	Kepemilikan Fasilitas Mandi, Cuci, Kakus	1=Berbagi pakai 2=Kepemilikan sendiri	-
12	X_{11}	Jumlah Tanggungan Kepala Keluarga	Rasio	-

Definisi Operasional Variabel Penelitian

1. Status Penerimaan Beasiswa Bidikmisi

   Status penerimaan siswa SLTA/SMA/MA kelas 12 yang mendaftar beasiswa Bidikmisi di Kabupaten/Kota Propinsi Jawa Timur.

2. Pekerjaan Ayah

   Pekerjaan Ayah adalah suatu aktivitas sehari-hari yang dilakukan Ayah untuk memenuhi kebutuhan harian.

3. Pekerjaan Ibu

   Pekerjaan Ibu adalah suatu aktivitas sehari-hari yang dilakukan Ibu untuk memenuhi kebutuhan harian.

4. Pendidikan Ayah

   Pendidikan Ayah adalah pendidikan formal tertinggi yang dimiliki oleh Ayah siswa pendaftar beasiswa Bidikmisi.

5. Pendidikan Ibu

   Pendidikan Ibu adalah pendidikan formal tertinggi yang dimiliki oleh Ibu siswa pendaftar beasiswa Bidikmisi.

6. Kepemilikan Rumah Tinggal Keluarga

   Kepemilikan Rumah Tinggal Keluarga adalah status kepemilikan rumah tinggal keluarga yang siswa pendaftar beasiswa Bidikmisi.

7. Luas Tanah Rumah Tinggal Keluarga

   Luas tanah yang dimaksud adalah luas tanah yang ditempati dan digunakan oleh rumah tinggal keluarga siswa pendaftar beasiswa Bidikmisi. Luas tanah yang ditempati dan digunakan untuk keperluan sehari-hari (sebatas atap). Luas tanah dinyatakan dengan satuan meter persegi (m^2).

8. Luas Bangunan Rumah Tinggal Keluarga

   Luas bangunan yang dimaksud adalah luas bangunan yang ditempati dan digunakan oleh rumah tinggal keluarga siswa pendaftar beasiswa Bidikmisi. Luas bangunan dinyatakan dengan satuan meter persegi (m^2).

9. Sumber Listrik yang digunakan Keluarga

   Sumber listrik sehari-hari yang digunakan oleh rumah tinggal keluarga siswa pendaftar beasiswa Bidikmisi.

10. Sumber Air yang digunakan Keluarga

    Sumber air sehari-hari yang digunakan oleh rumah tinggal keluarga siswa pendaftar beasiswa Bidikmisi untuk keperluan minum dan memasak.

11. Kepemilikan Fasilitas Mandi Cuci Kakus

    Kepemilikan Fasilitas Mandi Cuci Kakus adalah ketersediaan fasilitas mandi, cuci, kakus yang digunakan oleh rumah tangga siswa pendaftar beasiswa Bidikmisi.

12. Jumlah Tanggungan Kepala Keluarga

    Jumlah Tanggungan Kepala Keluarga adalah jumlah semua anak dan termasuk istri yang masih dalam tanggung jawab ayah

3.3 Struktur Data

Struktur data dari penelitian ini berdasarkan variabel-variabrl yang telah disebutkan sebelumnya disajikan dalam Tabel 3.3.

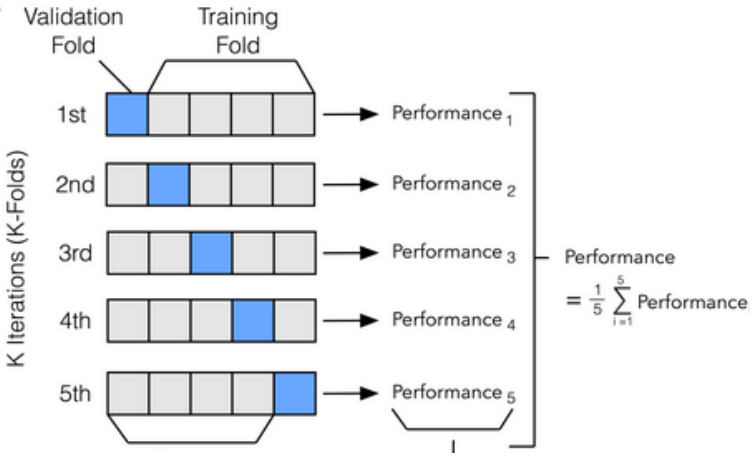
Tabel 3.3 Struktur Data

Objek/ Siswa (i)	Respon (Y_i)	Prediktor			
		X_1	X_2	...	X_{11}
1	Y_1	$X_{1.1}$	$X_{1.2}$	...	$X_{1.11}$
2	Y_2	$X_{2.1}$	$X_{2.2}$	...	$X_{2.11}$
3	Y_3	$X_{3.1}$	$X_{3.2}$	...	$X_{3.11}$
⋮	⋮	⋮	⋮	⋮	⋮
n	Y_n	$X_{n.1}$	$X_{n.2}$	...	$X_{n.11}$

3.4 Tahapan Penelitian

Dalam teknik data mining, yang pertama kali dilakukan sebelum mengolah data adalah melakukan *pre-processing* data. Dalam penelitian ini *pre-processing* yang dilakukan adalah menangani missing value, sehingga data yang mentah

menjadi data yang siap untuk diolah. Kemudian membagi data ke dalam data *training* dan data *testing* dengan menggunakan 5-*fold cross validation* dengan stratifikasi dimana komposisi dari masing-masing *fold* berisi 20% dari jumlah data mayor dan 20% dari jumlah data minor. Proses pemilihan anggota *fold* dilakukan dengan acak dan pengamatan-pengamatan di setiap *fold* tidak tumpang tindih.



Gambar 3.1 Ilustrasi Proses 5-*fold* Validasi

Gambar 3.1 menunjukkan pada validasi pertama *fold* pertama, kedua, ketiga sampai keempat digunakan sebagai data *training* dan *fold* kelima digunakan sebagai testing, kemudian dicari rata-rata model yang terbentuk untuk masing-masing *fold* setelah itu diuji terhadap data *testing*.

Adapun tahap dalam penelitian untuk penelitian setelah dilakukan *pre-processing* adalah sebagai berikut:

1. Menerapkan algoritma *random forest* pada data Beasiswa Bidikmisi tahun 2017 di Jawa Timur. Langkah-langkahnya adalah sebagai berikut:
   a. Menentukan *m* jumlah variabel predictor yang diambil secara acak dan *k* pohon yang akan dibentuk untuk digunakan dalam klasifikasi *random forest*. Nilai *m* dan *k* yang digunakan adalah:

$$m = \begin{cases} m_1 = \frac{1}{2}\left|\sqrt{11}\right| = 2 \\ m_2 = \left|\sqrt{11}\right| = 4 \\ m_3 = 2\left|\sqrt{11}\right| = 8 \end{cases}, \quad k = \begin{cases} k_1 = 25 \\ k_2 = 50 \\ k_3 = 100 \\ k_4 = 500 \end{cases}$$

b. Membentuk *tree model* dari dataset *testing* dengan kombinasi *m* variabel predictor yang diambil secara acak dan *k* buah ukuran pohon;
c. Membangun model *random forest* sebagai *base classifier* model AdaBoost.M2, SMOTE-Boosting dan SMOTE-Bagging pada data tiap *training* dengan suatu parameter yang diperoleh pada bagian (a) hingga diperoleh parameter optimum.
d. Mengklasifikasikan pengamatan-pengamatan pada data *testing* menggunakan fungsi klasifikasi AdaBoost.M2, SMOTE-Boosting dan SMOTE-Bagging.
e. Membentuk *confussion matrix* dan menghitung kinerja metode klasifikasi berdasarkan nilai AUC;
f. Kembali ke tahap (c) sampai ke tahap (e) untuk validasi kedua, ketiga, sampai kelima. Kemudian menghitung nilai rata-rata kinerja metode klasifikasi;
g. Menarik kesimpulan.

2. Menerapkan algoritma *adaptive boosting M2* (AdaBoost.M2) pada data Beasiswa Bidikmisi 2017 di Jawa Timur. Langkah-langkahnya adalah sebagai berikut:
   a. Membangun *base* model *classifier random forest* berdasarkan tahap (1) dan model AdaBoost.M2 menggunakan data *training*. Optimasi fungsi tujuan *random forest* menggunakan persamaan (2.3) dengan fungsi kendala $\sum_{i=1}^{n} a_i y_i = 0, \quad 0 \le a_i \le C \quad , i = 1, ..., n$. Tahapan untuk AdaBoost.M2 adalah sebagai berikut:
      i. Memboboti setiap pengamatan pada data *training* dengan bobot $D_1(i) = 1/m$, dimana *m* adalah banyaknya pengamatan di dalam data *training*.
      ii. Menentukan jumlah iterasi maksimal *boosting* yaitu 5, 10, 15, 25, 30, dan 50 iterasi.
      iii. Mengurutkan setiap pengamatan pada data *training* berdasarkan bobot $D_t(i)$ terbesar, kemudian membagi data ke dalam m' grup,

dan pada setiap grup diambil sampel secara acak sehingga diperoleh sebanyak m' sampel. m' yang digunakan pada penelitian ini yaitu kira-kira 40-60% dari total data *training*.

iv. Menghitung *error* menggunakan persamaan (2.3), bila diperoleh nilai *error* $e_t > 0.5$ maka iterasi berhenti.

v. Menghitung akurasi klasifikasi atau pembobot hipotesis *boosting* a_t Perhitungan berdasarkan persamaan (2.4)

vi. Memperbaharui bobot dari setiap pengamatan pada data *training* menggunakan persamaan (2.5).

vii. Mengulangi tahap (iii) sampai (vi) sampai iterasi maksimal tercapai.

viii. Kemudian diperoleh fungsi klasifikasi akhir seperti pada persamaan (2.6).

b. Mengklasifikasikan pengamatan-pengamatan pada data *testing* menggunakan fungsi klasifikasi Adam2 yang diperoleh pada tahapan (a);

c. Membentuk *confussion matrix* dan menghitung kinerja metode klasifikasi berdasarkan ukuran kinerja metode klasifikasi;

d. Kembali ke tahap (a) sampai ke tahap (c) untuk validasi kedua, ketiga, sampai kelima. Kemudian menghitung nilai rata-rata kinerja metode klasifikasi;

e. Menarik kesimpulan.

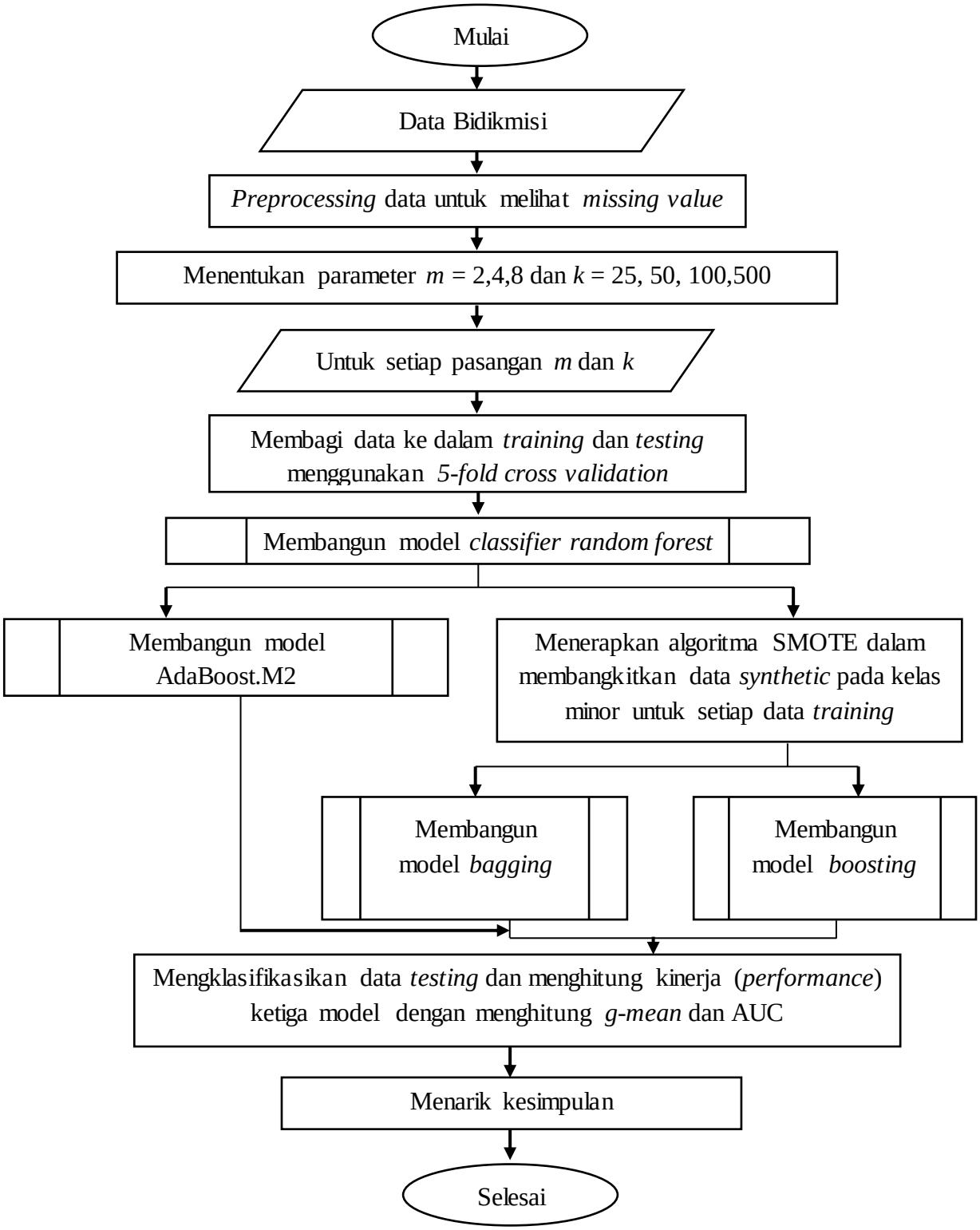
3. Menerapkan algoritma SMOTE-Boosting untuk data Bidikmisi. Langkah-langkahnya adalah sebagai berikut:

a. Membangun model *boosting* menggunakan data *training*. Tahapan untuk metode *boosting* bisa dilihat pada subbab 2.2.3:

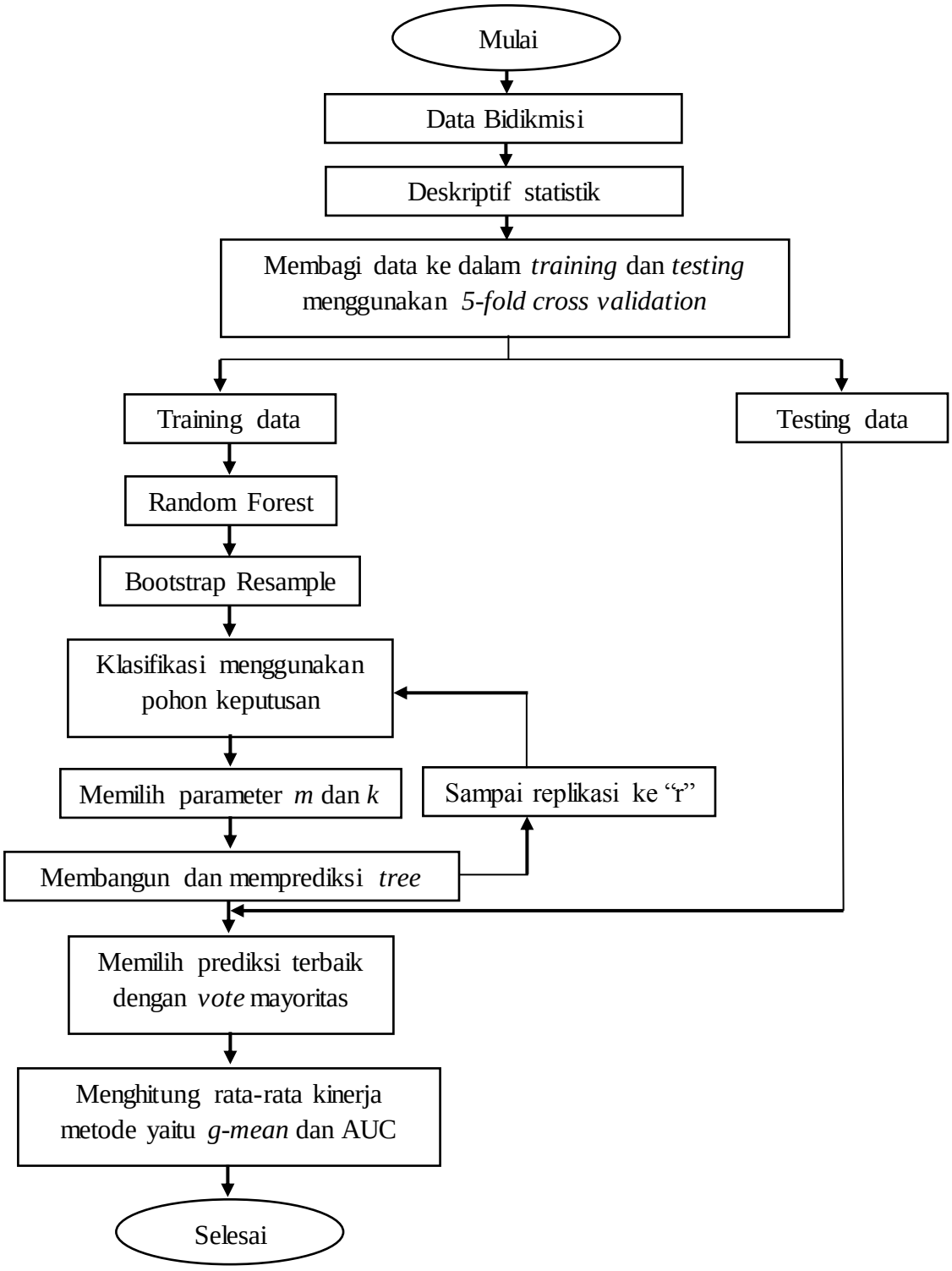
b. Membangkitkan data *synthetic* untuk menyeimbangkan komposisi kelas mayor dan kelas minor pada setiap data *training* menggunakan algoritma SMOTE. 5 tetangga terdekat dalam proses pembangkitan data *synthetic* berdasarkan rekomendasi Chawla dkk (2002).

c. Membangun model *classifier random forest* dan *boosting* menggunakan data *training* pada tahap (b) yang telah diseimbangkan oleh algoritma SMOTE;
d. Mengklasifikasikan pengamatan-pengamatan pada data *testing* menggunakan fungsi klasifikasi SMOTEBoosting yang diperoleh pada tahapan (c);
e. Membentuk *confussion matrix* dan menghitung kinerja metode klasifikasi berdasarkan nilai kinerja metode klasifikasi;
f. Kembali ke tahap (b) sampai ke tahap (e) untuk validasi kedua, ketiga, sampai kelima. Kemudian menghitung nilai rata-rata kinerja metode klasifikasi;
g. Menarik kesimpulan.

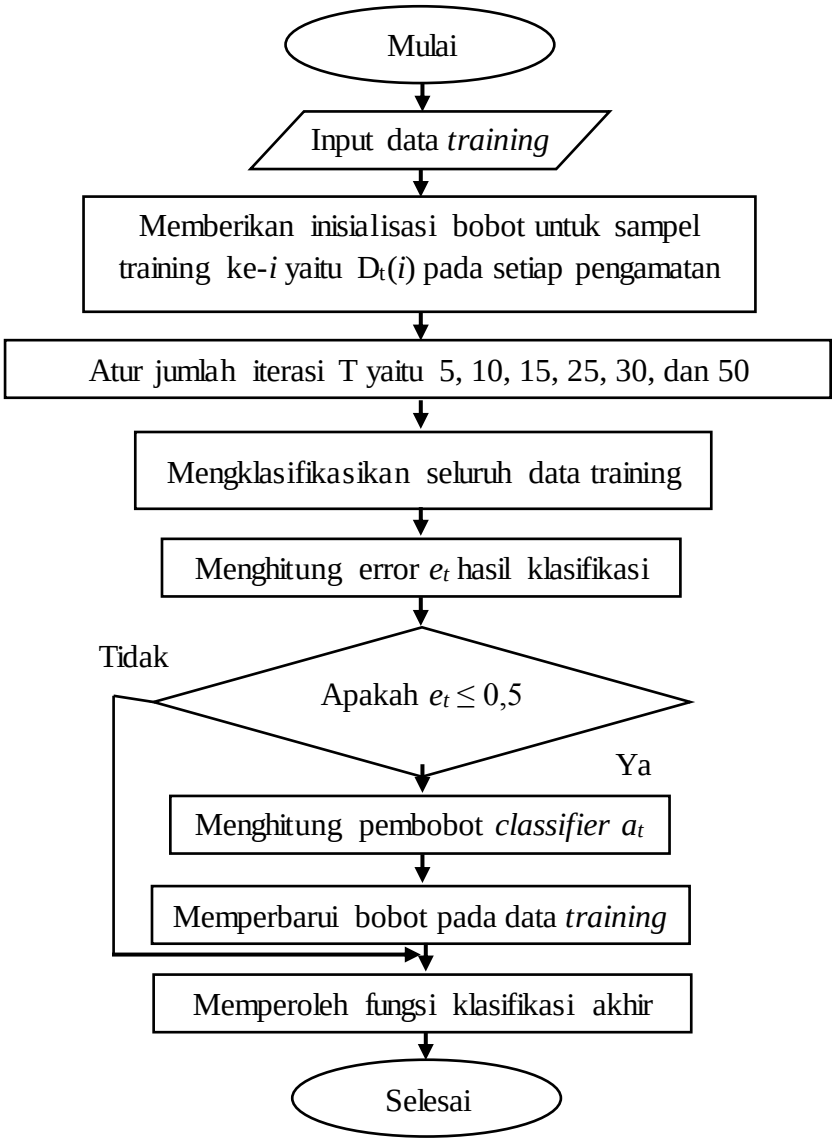
4. Menerapkan algoritma SMOTE-Bagging untuk data Bidikmisi. Langkah-langkahnya adalah sebagai berikut:
   a. Membangun model *bagging* menggunakan data *training*. Tahapan untuk metode *bagging* bisa dilihat pada subbab 2.2.4:
   b. Membangkitkan data *synthetic* untuk menyeimbangkan komposisi kelas mayor dan kelas minor pada setiap data *training* menggunakan algoritma SMOTE. 5 tetangga terdekat dalam proses pembangkitan data *synthetic* berdasarkan rekomendasi Chawla dkk (2002).
   c. Membangun model *classifier random forest* dan *bagging* menggunakan data *training* pada tahap (b) yang telah diseimbangkan oleh algoritma SMOTE.
   d. Mengklasifikasikan pengamatan-pengamatan pada data *testing* menggunakan fungsi klasifikasi SMOTE-Bagging yang diperoleh pada tahapan (c);
   e. Membentuk *confussion matrix* dan menghitung kinerja metode klasifikasi berdasarkan nilai kinerja metode klasifikasi;
   f. Kembali ke tahap (b) sampai ke tahap (e) untuk validasi kedua, ketiga, sampai kelima. Kemudian menghitung nilai rata-rata kinerja metode klasifikasi;
   h. Menarik kesimpulan.



Gambar 3.2 Flowchart Analisis pada Data Bidikmisi Tahun 2017



Gambar 3.3 Flowchart Analisis Random Forest pada Data Bidikmisi Tahun 2017



Gambar 3.4 Flowchart Analisis AdaBoost.M2 pada Data Bidikmisi Tahun 2017

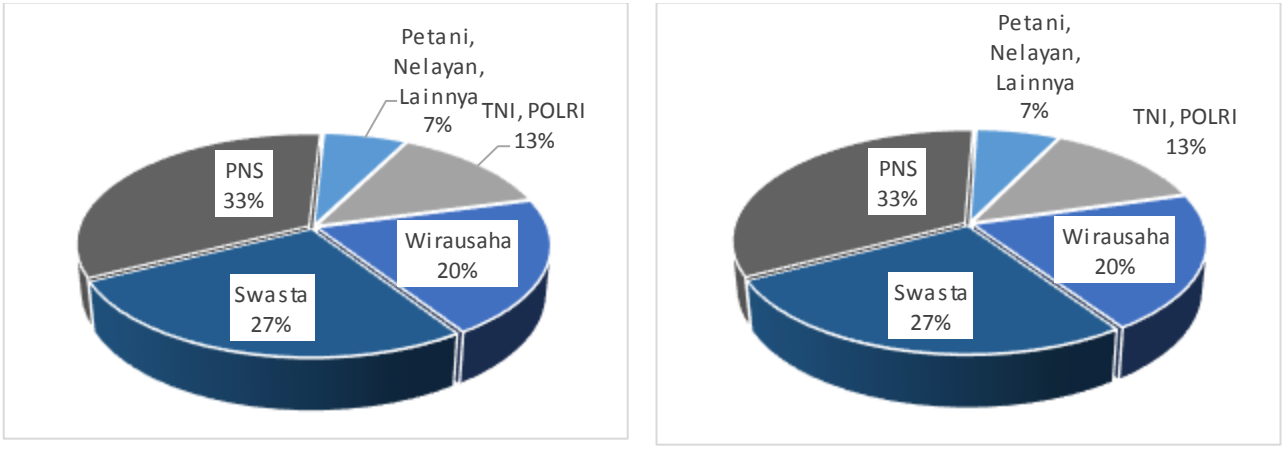
(halaman ini sengaja dikosongkan)

BAB 4
HASIL DAN PEMBAHASAN

4.1 Analisis Statistika Deskriptif

Langkah awal sebelum melakukan analisis klasifikasi pada Data Bidikmisi di Jawa Timur adalah melakukan analisis deskriptif untuk melihat karakteristik dari faktor-faktor yang mempengaruhi diterima atau tidaknya pelamar Bidikmisi. Data yang digunakan dalam penelitian ini merupakan data dengan *large* observasi yaitu sebanyak 53661. Karakteristik data dilihat dari banyaknya pengamatan tiap kelas dan persebaran data dari tiap faktor.

4.1.1 Karakteristik Data Bidikmisi Berdasarkan Pekerjaan Ayah dan Ibu

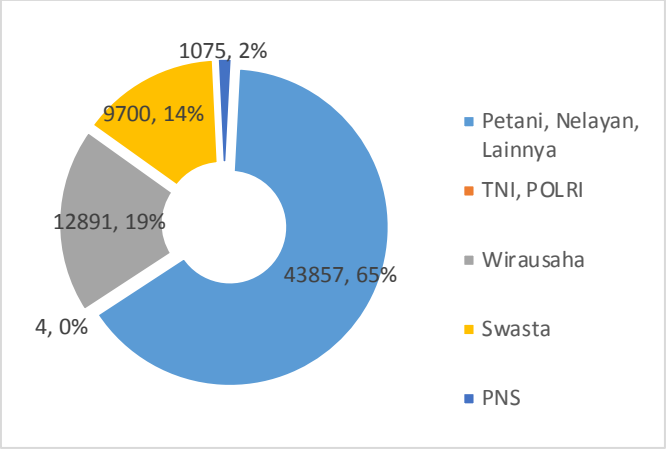


(a) Pekerjaan Ayah Berdasarkan Status Siswa Diterima Bidikmisi

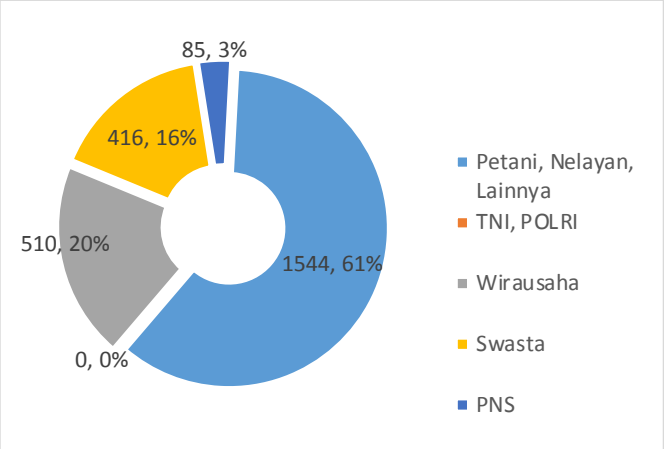
(b) Pekerjaan Ayah Berdasarkan Status Siswa Tidak Diterima Bidikmisi

Gambar 4.1 Karakteristik Pekerjaan Ayah Berdasarkan Status Siswa

Berdasarkan Gambar 4.1 (a) dan Gambar 4.1 (b) terlihat bahwa tidak ada perbedaan karakteristik pekerjaan ayah yang dominan berbeda untuk siswa yang diterima Bidikmisi dengan yang tidak diterima beasiswa. Dapat diartikan bahwa penerima Bidikmisi berasal dari berbagai golongan, dengan latar belakang pekerjaan ayah yang berbeda-beda. Pekerjaan ayah yang dominan adalah PNS sebesar 33%, kemudian disusul pegawai swasta dengan 27%, dan yang lainnya mempunyai persentase dibawah 20%.



(a) Pekerjaan Ibu Berdasarkan Status Siswa Diterima Bidikmisi

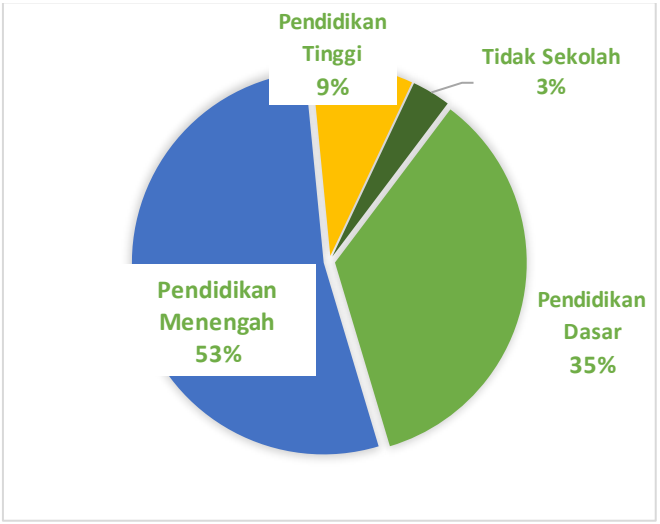


(b) Pekerjaan Ibu Berdasarkan Status Siswa Tidak Diterima Bidikmisi

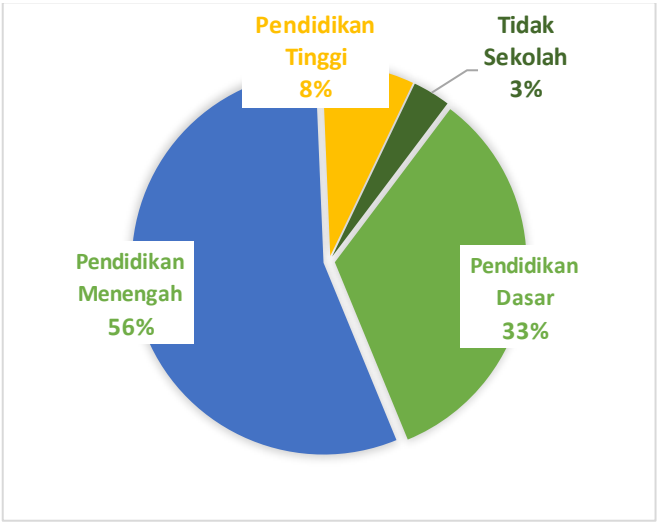
Gambar 4.2 Karakteristik Pekerjaan Ibu Berdasarkan Status Siswa

Berdasarkan Gambar 4.2 (a) dan Gambar 4.2 (b) terlihat bahwa karakteristik pekerjaan ibu mempunyai pola yang sama untuk siswa yang diterima Bidikmisi dengan yang tidak diterima beasiswa. Namun pekerjaan ibu yang dominan disini adalah petani, nelayan, atau lainnya dengan persentase yang mencapai 65%, kemudian disusul wirausaha sebesar 20%, dan yang lainnya dibawah 20%.

4.1.2 Karakteristik Data Bidikmisi Berdasarkan Pendidikan Ayah dan Ibu

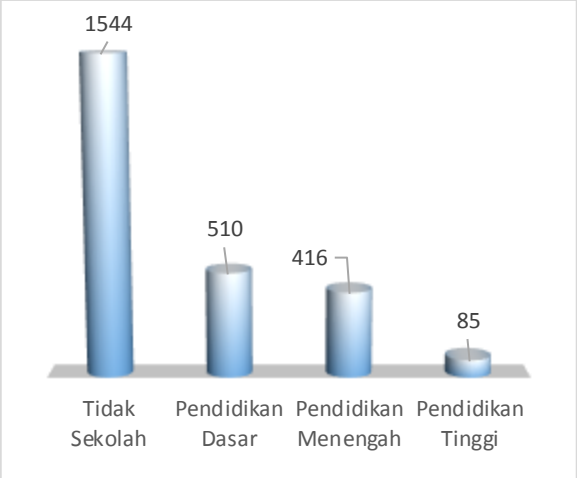


(a) Pendidikan Ayah Berdasarkan Status Siswa Tidak Diterima Bidikmisi

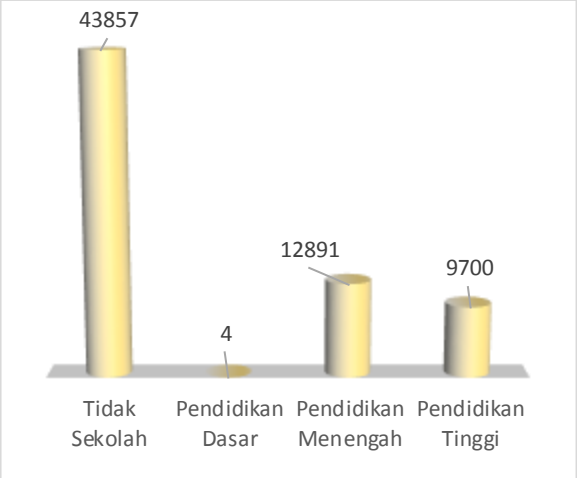


(b) Pendidikan Ayah Berdasarkan Status Siswa Diterima Bidikmisi

Gambar 4.3 Karakteristik Pendidikan Ayah Berdasarkan Status Siswa



(a) Pendidikan Ibu Berdasarkan Status Siswa Tidak Diterima Bidikmisi

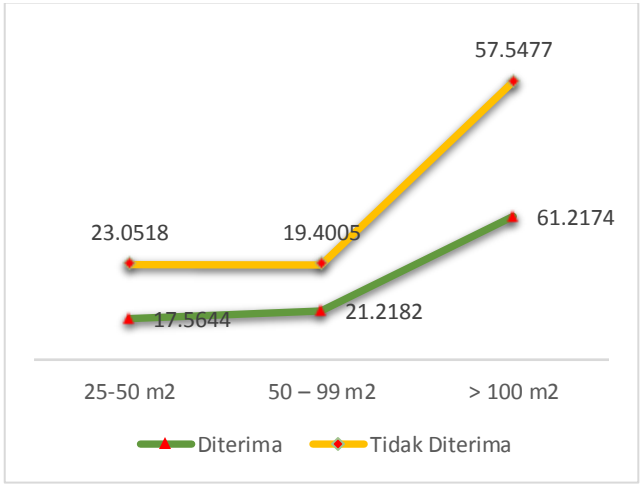


(b) Pendidikan Ibu Berdasarkan Status Siswa Diterima Bidikmisi

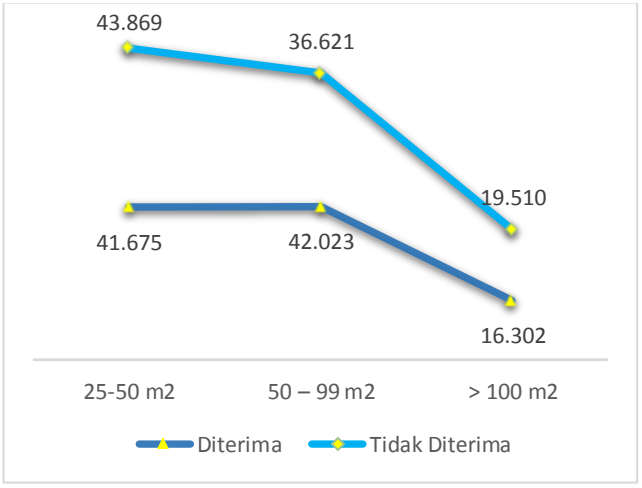
Gambar 4.4 Karakteristik Pendidikan Ibu Berdasarkan Status Siswa

Seperti yang terlihat pada Gambar 4.3 (a) dan Gambar 4.3 (b) bahwa lebih dari 50% pendidikan terakhir ayah adalah pendidikan menengah atau SMA, dan hanya sedikit yang tidak sekolah yaitu sekitar 3%. Berkebalikan dengan pendidikan terakhir ayah, pendidikan terakhir ibu pada Gambar 4.4 (a) dan Gambar 4.4 (b) yang dominan, atau rata-rata sekitar 68.21% adalah tidak mengemban pendidikan secara formal.

4.1.3 Karakteristik Data Berdasarkan Luas Tanah dan Luas Bangunan Rumah



(a) Luas Tanah Berdasarkan Status Siswa

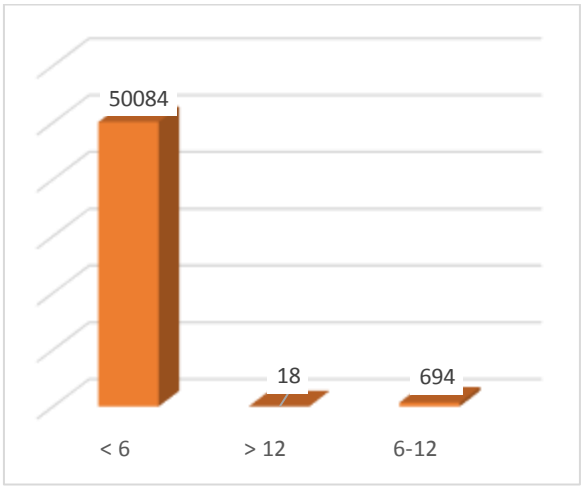


(b) Luas Bangunan Rumas Berdasarkan Status Siswa

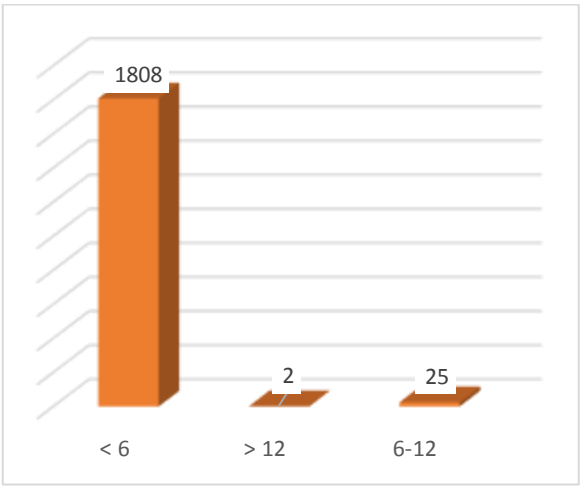
Gambar 4.5 Karakteristik Data Berdasarkan Luas Tanah dan Luas Bangunan Rumah

Gambar 4.5 (a) dan Gambar 4.5 (b) menyajikan plot karakteristik luas tanah dan luas bangunan rumah untuk setiap status siswa, dari plot tersebut terlihat bahwa siswa yang diterima bidikmisi lebih dari 60% mempunyai luas tanah lebih dari 100 m^2, namun cenderung mempunyai luas bangunan rumah yang kecil yaitu 25-50 m^2 dan hanya sekitar 16% yang luas bangunannya sama dengan luas tanahnya.

4.1.4 Karakteristik Data Berdasarkan Jumlah Tanggungan Kepala Keluarga



(a) Jumlah Tanggunan Kepala Keluarga Siswa Diterima



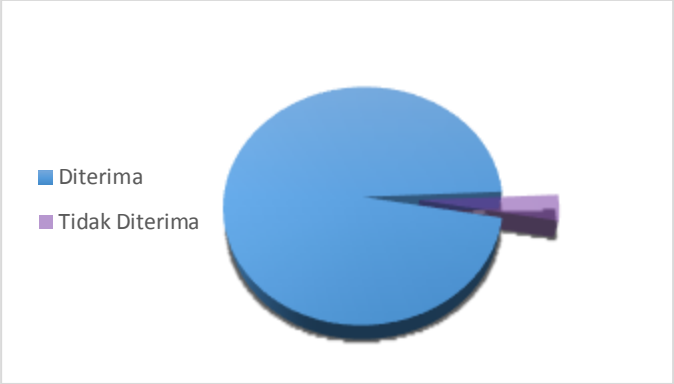
(b) Jumlah Tanggunan Kepala Keluarga Siswa Tidak Diterima

Gambar 4.6 Karakteristik Data Jumlah Tanggungan Kepala Keluarga Berdasarkan Status Siswa

Gambar 4.6 (a) dan Gambar 4.6 (b) menunjukkan *bar plot* karakteristik jumlah tanggungan yang dikategorikan menjadi 3. Jelas sekali terlihat bahwa jumlah data untuk status siswa diterima berbeda jauh dengan status siswa yang tidak diterima. Untuk kategori "kurang dari 6" dengan status siswa diterima terdapat 50.000 lebih data, sedangkan untuk kategori yang sama status siswa tidak diterima hanya sekitar 1800 data.

4.1.5 Karakteristik Data Berdasarkan Status Siswa

Bagian ini memuat karakteristik dari data bidikmisi berdasarkan kelas status penerimaannya. Data tersebut memiliki karakteristik yang dapat dilihat dari pola persebaran data dari setiap atribut-atribut dan kategori kelasnya.



Gambar 4.7 Karakteristik Data Berdasarkan Status Siswa

Seperti yang terlihat pada Gambar 4.7, kelas mayoritas adalah status siswa diterima yang mempunyai persentase 97% dan hanya 3% nya adalah kelas minoritas yaitu status siswa tidak diterima. Kondisi seperti ini akan menyebabkan *classifier* bias terhadap kelas mayoritas, artinya bahwa mesin klasifikasi akan cenderung memprediksi ke kelas mayoritas dan mengabaikan kelas minor. Oleh karena itu perlu digunakan metode klasifikasi *ensemble* yang mampu menangani masalah bias tersebut, akan dibahas pada subbab selanjutnya.

4.2 Deteksi *Missing Value* pada Data Bidikmisi

Tahapan pertama yang dilakukan dalam abalisis *pre-processing* adalah mendeteksi *missing value*. Deteksi *missing value* bertujuan untuk mengatasi permasalahan dimana adanya sel-sel kosong pada satu atau beberapa variabel.

Tabel 4.1 Jumlah Objek *Missing Value* Data Bidikmisi

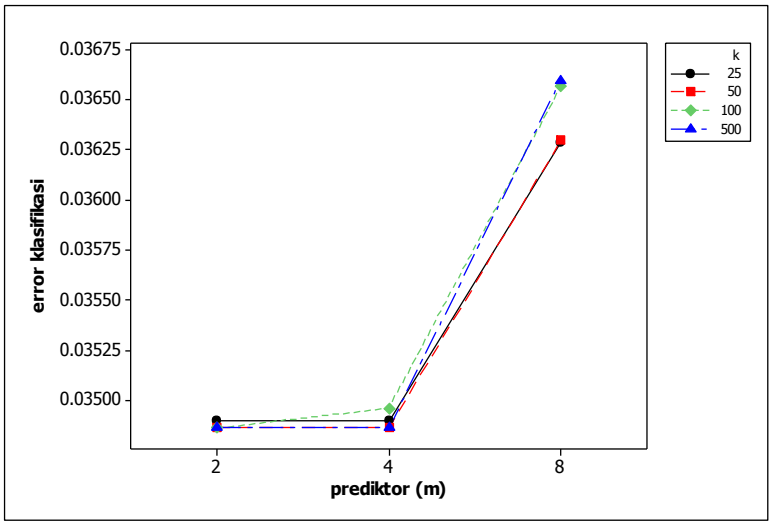
	N		%
	Valid	Missing	Missing
X1	53661	0	0
X2	53661	0	0
X3	53661	0	0
X4	53661	0	0
X5	53661	0	0
X6	52861	800	1,5
X7	52884	777	1,4
X8	53661	0	0
X9	53661	0	0
X10	53130	531	1
X11	53190	471	0,9

Dari Tabel 4.1 dapat diketahui berapa *missing value* yang terdapat pada masing-masing variabel data, dengan demikian dapat diketahui persentase *missing value* masing-masing variabel. Ketika *missing value* dibawah 10% untuk kasus diatas secara umum dapat diabaikan kecuali ketika *missing value* muncul secara nonrandom. Jumlah kasus tanpa *missing value* harus cukup untuk teknik analisis yang dipilih jika nilai penggantian tidak akan diganti/ diperhitungkan untuk *missing value*. Namun dalam penelitian ini, untuk variabel dengan *missing value* akan dihapus dari penelitian. Sehingga total data yang digunakan untuk penelitian adalah 52631, setelah dilakukan deteksi *missing value* dan ditangani dengan menghapus data tersebut.

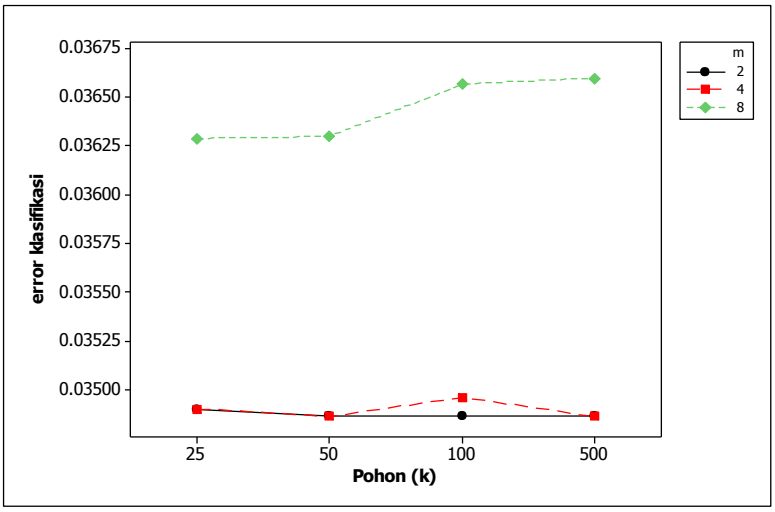
4.3 Analisis Klasifikasi Bidikmisi dengan Metode *Random Forest*

Telah dijelaskan pada bab 3 mengenai langkah-langkah analisis metode *random forest*. Langkah pertama adalah menentukan nilai *m* yaitu nilai variabel prediktor penting dan *k* yaitu nilai pohon yang digunakan dalam analisis, dimana nilai *m* yang digunakan adalah $\frac{1}{2}\left|\sqrt{p}\right|, \left|\sqrt{p}\right|$, dan $2\left|\sqrt{p}\right|$. *p* adalah banyaknya variabel predictor yang digunakan pada penelitian ini yaitu $p = 11$. Sehingga nilai *p* yang digunakan adalah 2, 4, dan 8. Langkah selanjutnya adalah menentukan nilai *k*. Umumnya nilai $k = 50$ sudah memberikan hasil yang memuaskan pada masalah klasifikasi (Breimann, 1996), sedangkan Sutton (2005) menyarankan menggunakan $k \geq 100$ karena dengan nilai tersebut cenderung menghasilkan misklasifikasi yang konstan. Sehingga *k* yang digunakan pada penelitian ini adalah 25, 50, 100, dan 500. Selanjutnya akan disajikan perbandingan nilai *m* dan *k* untuk melihat kombinasi mana yang menghasilkan nilai misklasifikasi paling kecil disajikan pada Gambar 4.8 dan Gambar 4.9.

Gambar 4.8 menunjukkan perubahan nilai *m* menyebabkan error klasifikasi berubah-ubah. Nilai error klasifikasi menurun ketika $m = 4$, kemudian naik drastis ketika $m = 8$, untuk setiap nilai *k* polanya sama. Sehingga dapat disimpulkan bahwa nilai *m* optimum ketika jumlah variabel predictor sama dengan 4. Gambar 4.9 menunjukkan perubahan error klasifikasi akibat berubahnya nilai *k*. Pada saat nilai $m = 8$, semakin besar nilai *k* maka semakin besar pula nilai error klasifikasinya.



Gambar 4.8 Error Klasifikasi *Random Forest* Berukuran Pohon *k* Untuk Setiap Prediktor *m*



Gambar 4.9 Error Klasifikasi *Random Forest* Prediktor *m* Untuk Setiap Pohon *k*

Lain halnya untuk $m = 4$, ketika ukuran k antara 25 sampai 50 nilai error klasifikasinya menurun kemudian naik pada saat $k = 100$, kemudian turun lagi ketika $k = 500$. Untuk $m = 2$ nilai error klasifikasi cenderung turun pada saat nilai k antara 25 sampai 100, kemudian stabil pada saat $k = 500$. Berdasarkan Gambar 4.20 dan Gambar 4.21, nilai error terendah diperoleh pada saat $m = 4$ dan $k = 100$. Dapat dikatakan bahwa akurasi untuk metode *random forest* akan mencapai optimal ketika prediktornya berjumlah 4 dan konvergen pada saat menggunakan 100 pohon klasifikasi.

4.4 Analisis Klasifikasi Data Bidikmisi dengan Metode *Adaptive Bossting M2* (AdaBoost.M2) dengan *Random Forest* Sebagai Base *Classifier*

Pembentukan model dilakukan dengan membagi data menjadi data *training* dan data *testing*. Data *training* digunakan untuk membangun model dan data *testing* untuk validasi model. Penelitian ini menggunakan 5-*fold cross validation* dengan partisi 20% untuk setiap foldnya. AdaBoost.M2 adalah perluasan dari AdaBoost, dimana AdaBoost.M2 menggunakan *pseudo-loss* untuk menghitung residualnya. Berikut adalah hasil *confussion matrix* AdaBoost.M2 menggunakan *random forest* sebagai *base classifier*.

Tabel 4.2 *Confussion Matrix* AdaBoost.M2 untuk Masing-masing Iterasi

Iterasi 5				Iterasi 10			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	0	367	367	Tidak Diterima	1	366	367
Diterima	3	10156	10159	Diterima	25	10134	10159
Total	3	10523	10526	Total	26	10500	10526
Iterasi 15				Iterasi 25			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	1	367	368	Tidak Diterima	1	366	367
Diterima	22	10137	10159	Diterima	22	10137	10159
Total	23	10504	10527	Total	23	10503	10526
Iterasi 30				Iterasi 50			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	2	367	369	Tidak Diterima	3	364	367
Diterima	30	10129	10159	Diterima	31	10128	10159
Total	32	10496	10528	Total	34	10492	10526

Tabel 4.2 menyajikan *confussion matrix* untuk metode AdaBoost.M2, dari tabel tersebut dapat pula dihitung nilai kinerja metode secara manual untuk Iterasi ke-10 (untuk perhitungan lebih lengkap lihat **Lampiran B1**).

$$\text{Presisi} = \frac{\text{TP}}{(\text{TP+FP})} = \frac{1}{(1+25)} = 0,0385$$

$$\text{Recall} = \frac{\text{TP}}{(\text{TP+FN})} = \frac{1}{(1+366)} = 0,0027$$

$$\text{F-Value} = \frac{2(\text{Recall} \times \text{Presisi})}{(\text{Recall+Presisi})} = \frac{0,00021}{0,04123} = 0,0051$$

$$\text{Sensitivity} = 0,00273$$

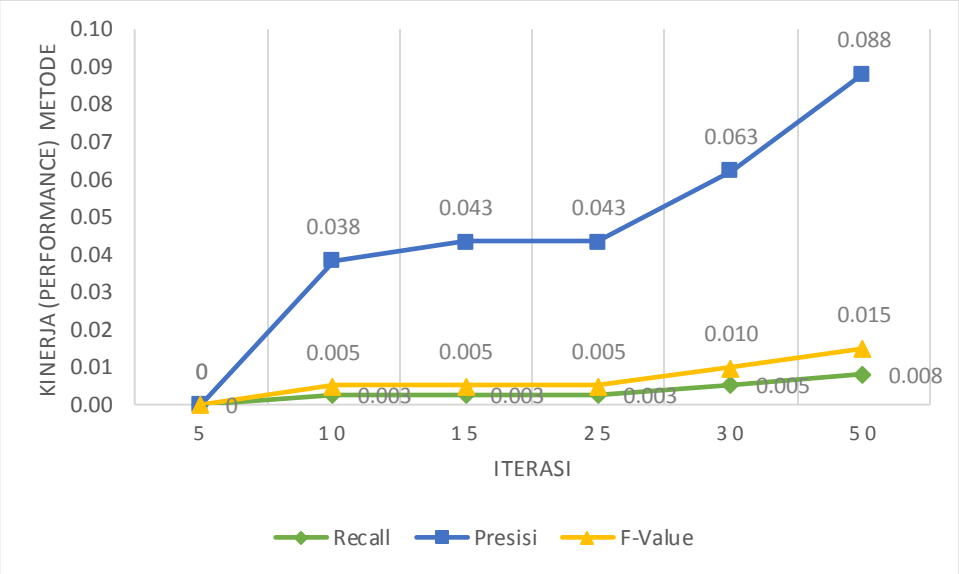
$$\text{Specificity} = \frac{\text{TN}}{(\text{TN+FP})} = \frac{10134}{(10134+25)} = 0,9975$$

$$\text{G-Mean} = \sqrt{\text{Sensitivity} \times \text{Specificity}} = \sqrt{0,00273 \times 0,9975} = 0,0521$$

$$\text{TPR} = 0,00272$$

$$\text{FPR} = 1 - \text{Specificity} = 1 - 0,9975 = 0,00246$$

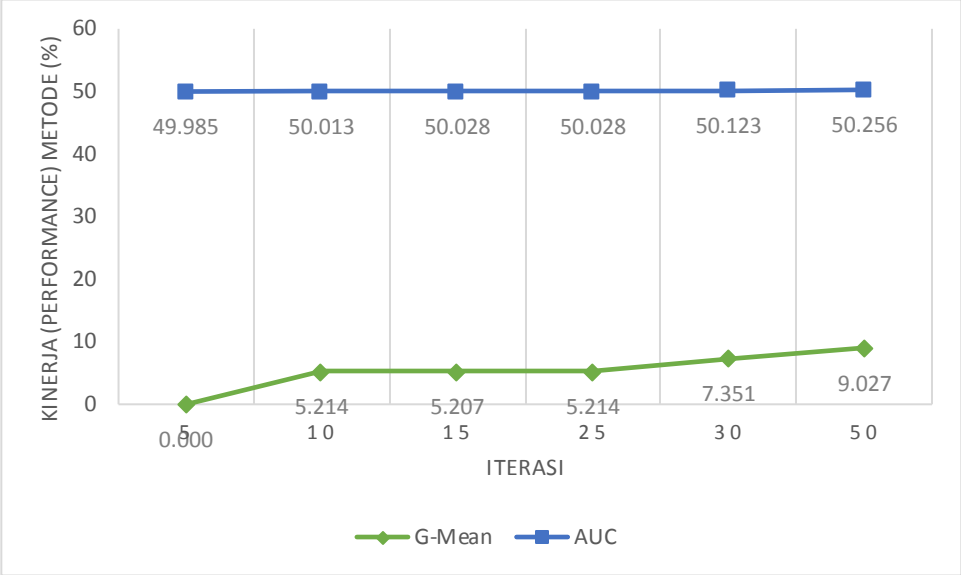
$$\text{AUC} = \frac{1+\text{TPR-FPR}}{2} = \frac{1,00033}{2} = 0,50013$$



Gambar 4.10 Kinerja (*Performance*) Metode pada Beberapa Iterasi AdaBoost.M2

Gambar 4.10 menyajikan kinerja metode klasifikasi data Bidikmisi pada tiap iterasi yang digunakan. Nilai *recall*, *presisi* dan *f-value* cenderung mengalami peningkatan, walaupun dengan nilai yang masih sangat kecil. Nilai *presisi* berkaitan dengan banyaknya objek yang diprediksi ke dalam kelas *positif*, dengan semakin naiknya kurva maka dapat dikatakan bahwa, semakin tinggi jumlah iterasi, maka

semakin banyak objek yang diklasifikasikan masuk ke dalam kelas *positif*. Namun, jumlah *false positif* lebih banyak dibandingkan *true positif*nya sehingga nilai *presisi* sangat kecil.



Gambar 4.11 Nilai G-Mean dan AUC pada Beberapa Iterasi AdaBoost.M2

Gambar 4.10 menunjukkan ukuran kinerja metode yaitu G-Means dan AUC untuk model AdaBoost.M2. Nilai G-mean dan AUC cenderung mencapai maksimal pada iterasi ke-50, dengan nilai 50,256% dan 9,027%.

4.5 Analisis Klasifikasi Data Bidikmisi dengan Metode SMOTE-Boosting dengan *Random Forest* Sebagai Base *Classifier*

Seperti penjelasan sebelumnya, data dibagi menjadi data *training* dan data *testing* menggunakan 5-*fold cross validation*. Kemudian membangkitkan data *synthetic* untuk menyeimbangkan komposisi kelas mayoritas dan kelas minor menggunakan algoritma SMOTE (lihat subbab 3.3). Dengan menggunakan beberapa iterasi dan nilai *over-sampling rate* dari SMOTE, Tabel 4.3 menyajikan *confussion matrix* untuk metode SMOTE-Boosting, dari tabel tersebut dapat pula dihitung nilai kinerja metode secara manual. Diberikan contoh untuk menghitung kinerja metode Iterasi ke-10 (untuk perhitungan lebih lengkap lihat **Lampiran B2**).

$$\text{Presisi} = \frac{\text{TP}}{(\text{TP+FP})} = \frac{32}{(32+732)} = 0,042$$

$$\text{Recall}=\frac{\text{TP}}{(\text{TP}+\text{FN})}=\frac{32}{(32+335)}=0,087$$

$$\text{F-Value}=\frac{2(\text{Recall}\times\text{Presisi})}{(\text{Recall}+\text{Presisi})}=\frac{0,00771}{0,129}=0,057$$

$$\text{Sensitivity}=0,087$$

$$\text{Specificity}=\frac{\text{TN}}{(\text{TN}+\text{FP})}=\frac{9427}{(9427+732)}=0,928$$

$$\text{G-Mean}=\sqrt{\text{Sensitivity}\times\text{Specificity}}=\sqrt{0,065\times0,999}=0,2841$$

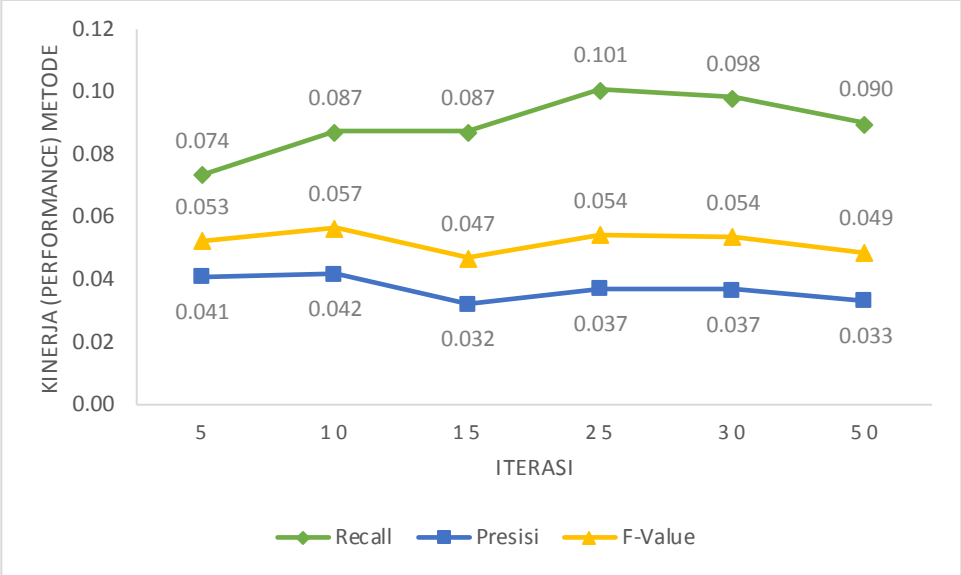
$$\text{TPR}=0,087$$

$$\text{FPR}=1-\text{Specificity}=1-0,999=0,072$$

$$\text{AUC}=\frac{1+\text{TPR-FPR}}{2}=\frac{1,015}{2}=0,5075$$

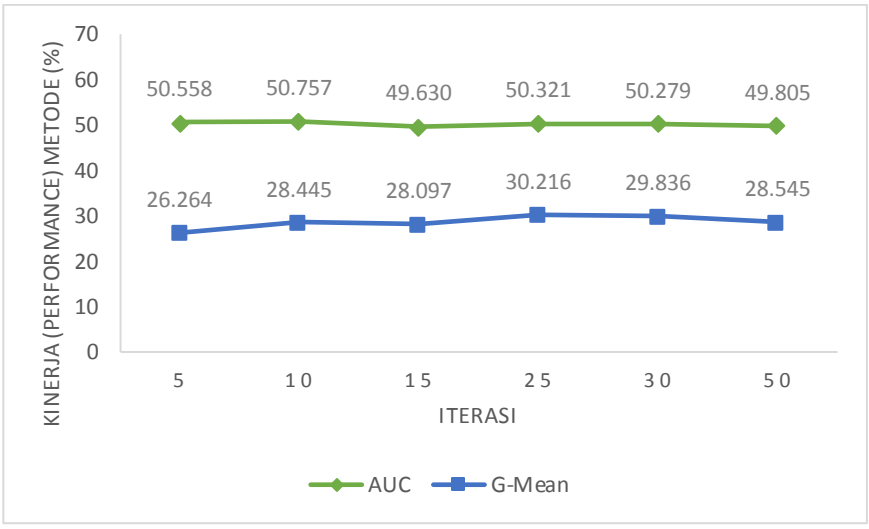
Tabel 4.3 *Confussion Matrix* SMOTE-Boosting untuk Masing-masing Iterasi

Iterasi 5				Iterasi 10			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	27	340	367	Tidak Diterima	32	335	367
Diterima	634	9525	10159	Diterima	732	9427	10159
Total	661	9865	10526	Total	764	9762	10526
Iterasi 15				Iterasi 25			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	32	335	367	Tidak Diterima	37	330	367
Diterima	961	9199	10160	Diterima	959	9201	10160
Total	993	9534	10527	Total	996	9531	10527
Iterasi 30				Iterasi 50			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	36	331	367	Tidak Diterima	33	334	367
Diterima	940	9220	10160	Diterima	953	9206	10159
Total	976	9551	10527	Total	986	9540	10526



Gambar 4.12 Kinerja (*Performance*) Metode pada Beberapa Iterasi SMOTE-Boosting

Gambar 4.12 menyajikan kinerja metode klasifikasi data Bidikmisi pada tiap iterasi yang digunakan. Nilai *recall* dan *f-value* mengalami fluktuasi, walaupun dengan nilai yang masih sangat kecil. Nilai *presisi* berkaitan dengan banyaknya objek yang diprediksi ke dalam kelas *positif*, karena jumlah *false positif* lebih banyak dibandingkan *true positif*nya sehingga nilai *presisi* sangat kecil.



Gambar 4.13 Nilai G-Mean dan AUC pada Beberapa Iterasi SMOTE-Boosting

Gambar 4.13 menunjukkan ukuran kinerja metode yaitu G-Means dan AUC untuk model SMOTE-Boosting. Nilai G-mean dan AUC cenderung stabil tanpa kenaikan yang drastis. Sementara dari perhitungan manual diperoleh nilai

specificity sebesar 0,928 berarti fungsi pemisah yang diperoleh berhasil mengidentifikasi 92,79% pengamatan yang berasal dari status diterima bidikmisi.

4.6 Analisis Klasifikasi Bidikmisi dengan Metode SMOTE-Bagging dengan *Random Forest* Sebagai Base *Classifier*

Langkah awal yang harus dilakukan dalam analisis metode *bagging* adalah dengan membagi data menjadi data *training* dan data *testing* dengan 5-*fold cross validation*. Kemudian dilakukan *bootstrapping* pada data *training*. Karena metode ini gabungan antara *bagging* dan SMOTE, maka setelah dilakukan *bootstrapping* data akan di bangkitkan lagi menggunakan SMOTE. Lebih jelasnya bisa dilihat pada subbab 3.3. Dengan beberapa iterasi yang digunakan, disajikan hasil kinerja metode SMOTE-Bagging dalam bentuk *confussion matrix* dan juga pada gambar sebagai berikut.

Tabel 4.4 *Confussion Matrix* SMOTE-Bagging untuk Masing-masing Iterasi

Iterasi 5				Iterasi 10			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	27	340	367	Tidak Diterima	44	323	367
Diterima	536	9623	10159	Diterima	859	9300	10159
Total	563	9963	10526	Total	903	9623	10526
Iterasi 15				Iterasi 25			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	37	330	367	Tidak Diterima	36	331	367
Diterima	712	9447	10159	Diterima	681	9478	10159
Total	749	9777	10526	Total	717	9809	10526
Iterasi 30				Iterasi 50			
Actual Class	Prediction Class		Total	Actual Class	Prediction Class		Total
	Tidak Diterima	Diterima			Tidak Diterima	Diterima	
Tidak Diterima	42	325	367	Tidak Diterima	43	324	367
Diterima	784	9375	10159	Diterima	760	9399	10159
Total	826	9700	10526	Total	803	9723	10526

Tabel 4.4 menyajikan *confussion matrix* untuk metode SMOTE-Bagging, dari tabel tersebut dapat pula dihitung nilai kinerja metode secara manual untuk iterasi ke-10 sebagai berikut (untuk perhitungan lebih lengkap lihat **Lampiran B3**). Selanjutnya akan dilihat bagaimana plot dari nilai kinerja metode untuk jumlah iterasi yang berbeda.

$$\text{Presisi}=\frac{\text{TP}}{(\text{TP}+\text{FP})}=\frac{44}{(44+859)}=0,049$$

$$\text{Recall}=\frac{\text{TP}}{(\text{TP}+\text{FN})}=\frac{44}{(44+323)}=0,1199$$

$$\text{F-Value}=\frac{2(\text{Recall}\times\text{Presisi})}{(\text{Recall}+\text{Presisi})}=\frac{0,0117}{0,1689}=0,069$$

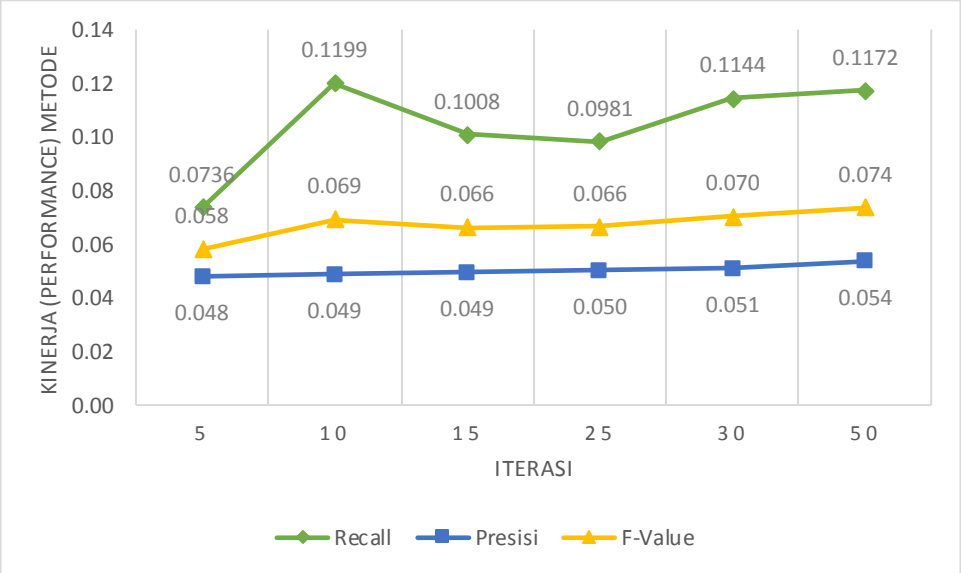
$$\text{Sensitivity}=0,1199$$

$$\text{Specificity}=\frac{\text{TN}}{(\text{TN}+\text{FP})}=\frac{9300}{(9300+859)}=0,9154$$

$$\text{G-Mean}=\sqrt{\text{Sensitivity}\times\text{Specificity}}=\sqrt{0,1199\times0,9154}=0,3313$$

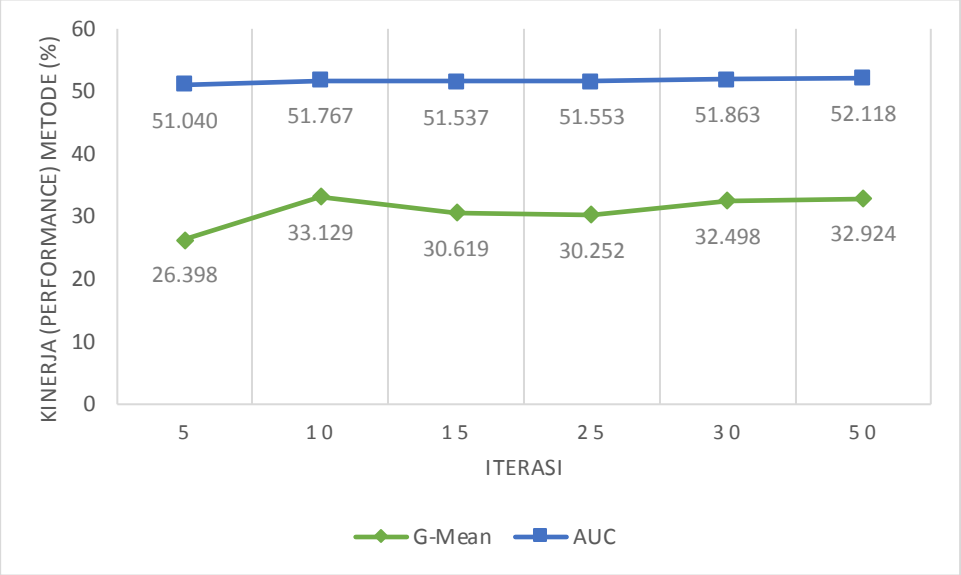
$$\text{TPR}=0,1199$$

$$\text{FPR}=1-\text{Specificity}=1-0,9154=0,0846$$



Gambar 4.14 Kinerja (*Performance*) Metode pada Beberapa Iterasi SMOTE-Bagging

Gambar 4.14 menyajikan kinerja metode klasifikasi data Bidikmisi pada tiap iterasi yang digunakan. Nilai *recall, presisi* dan *f-value* cenderung mengalami peningkatan, walaupun dengan nilai yang masih kecil. Nilai *presisi* berkaitan dengan banyaknya objek yang diprediksi ke dalam kelas *positif*, dengan semakin naiknya kurva maka dapat dikatakan bahwa, semakin tinggi jumlah iterasi, maka semakin banyak objek yang diklasifikasikan masuk ke dalam kelas *positif*. Namun, jumlah *false positif* lebih banyak dibandingkan *true positif*nya sehingga nilai *presisi* sangat kecil.



Gambar 4.15 Nilai G-Mean dan AUC pada Beberapa Iterasi SMOTE-Bagging

Gambar 4.15 menunjukkan ukuran kinerja metode yaitu G-Means dan AUC untuk model SMOTE-Bagging. Nilai G-mean mencapai maksimal pada iterasi ke-10, dengan nilai 33,129% dan nilai AUC maksimal pada iterasi ke-50 sebesar 32,924%. Sementara nilai *specificity* sebesar 0,9154 berarti fungsi pemisah yang diperoleh berhasil mengidentifikasi 91,54% pengamatan yang berasal dari status diterima bidikmisi.

4.7 Perbandingan Kinerja (*Performance*) Metode Klasifikasi

Setelah dilakukan analisis pada data Bidikmisi tahun 2017 di Jawa Timur menggunakan *random forest*, AdaBoost.M2 *random forest*, SMOTE-Boosting *random forest*, dan SMOTE-Bagging *random forest*, selanjunya pada bagian ini dilakukan perbandingan dari kinerja metode semua model optimum yang diperoleh.

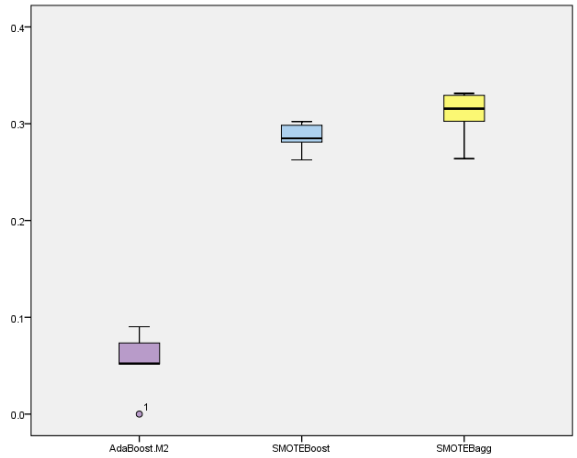
Perbandingan beberapa metode tersebut diukur menggunakan kinerja metode klasifikasi yang meliputi presisi, *recall*, *f-value*, *sensitivity*, dan *specificity*, *g-mean* dan AUC yang merupakan hasil klasifikasi dengan parameter terbaik dari masing-masing metode. Perbandingan hasil klasifikasi tersebut ditampilkan pada Tabel 4.4, dapat dilihat bahwa kinerja dari seluruh metode menunjukkan SMOTE-Boosting dan SMOTE-Bagging mempunyai nilai yang cenderung sama. Ketepatan klasifikasi kelas *positif* yang dilakukan oleh model AdaBoost.M2 yaitu rata-rata dari klasifikasi kelima *fold* sebesar 8,8235% yang berarti rata-rata hanya 8,8235% dari pengamatan di tiap fold data bidikmisi telah diklasifikasikan dengan benar. Jika dilihat dari nilai *sensitivity* dan *specificity*, AdaBoost.M2 hanya bisa mengklasifikasikan 0,8174% pengamatan yang berasal dari status tidak diterima (minoritas) sebagai kelas tidk diterima namun berhasil mengklasifikasikan 99,97% pengamatan yang berasal dari status diterima (mayoritas) sebagai kelas diterima. Adanya kasus *imbalance* pada data menyebabkan rendahnya nilai *sensitivity* dikarenakan fungsi pemisah *random forest* cenderung mengklasifikan pengamatan ke dalam kelas mayoritas, sehingga pengklasifikasian kelas minoritas hanya benar diklasifikasikan kurang dari 1%. Setelah dilakukan penyeimbangan data pada kedua kelas dengan SMOTE dan dilakukan *boosting* dan *bagging* diperoleh hasil yang lebih baik. Hal ini dibuktikan dengan kinerja metode dengan nilai g-mean yang diperoleh menggunakan SMOTE-Boosting *random forest* dan SMOTE-Bagging *random forest* lebih tinggi dibandingkan AdaBoost.M2.

Tabel 4.5 Perbandingan Kinerja (*Performance*) Metode Klasifikasi Data Bidikmisi

Model	Rata-Rata Kelima *Fold*							
	Akurasi	*Presisi*	*Recall*	*F-Value*	*Sensitivity*	*Specificity*	*G-mean*	*AUC*
AdaBoost.M2	0,9648	0.088235	0.008174	0.014963	0.008174	0.999705	0.090274	0.502561
SMOTE-Boosting	0,9074	0.041885	0.100817	0.056587	0.100817	0.937592	0.302161	0.50757
SMOTE-Bagging	0,9167	0.053549	0.119891	0.073504	0.119891	0.947239	0.331291	0.521178

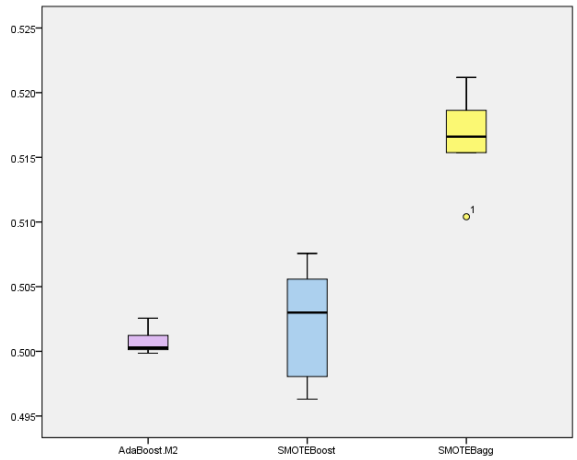
Berdasarkan Tabel 4.5 dapat dilihat pula bahwa AdaBoost.M2 menghasilkan nilai akurasi dan *specificity* yang tinggi. Hal ini dikarenakan pada proses *boosting*-nya, AdaBoost.M2 berhasil mengambil keuntungan dari kesalahan klasifikasi yang dilakukan *random forest* di tiap iterasi *boosting*-nya sehingga dapat meningkatkan ketepatan klasifikasi khususnya klasifikasi pada kelas mayoritas. Sementara SMOTE-Boosting dan SMOTE-Bagging menghasilkan nilai kinerja

metode yang hampir sama pada semua kriteria dikarenakan proses penyeimbangan distribusi kelas *training set* sehingga mengakibatkan peningkatan ketepatan klasifikasi pada kelas minoritas.



Gambar 4.16 *Boxplot* Kinerja (*Performance*) Metode dengan G-Mean

Gambar 4.16 menyajikan *boxplot* dari nilai-nilai g-mean yang dihasilkan pada setiap model. Nilai g-mean yang dihasilkan menggunakan SMOTE-Bagging *random forest* yang ditunjukkan oleh warna kuning pada gambar, cenderung lebih tinggi sedikit dibandingkan dengan SMOTE-Boosting. Nilai g-mean berkisar antara 26% sampai 33%. Nilai g-mean yang dihasilkan menggunakan SMOTE-Boosting *random forest* yang ditunjukkan oleh warna biru pada gambar. Variasi dari g-mean yang dihasilkan oleh algoritma SMOTE-Boosting *random forest* cenderung lebih kecil dibandingkan kedua algoritma lain, nilai g-mean berkisar antara 26% sampai 30%.



Gambar 4.17 *Boxplot* (*Performance*) Metode dengan AUC

Selanjutnya Gambar 4.17 menyajikan *boxplot* dari nilai-nilai AUC yang dihasilkan pada setiap model. Nilai AUC yang dihasilkan menggunakan SMOTE-Bagging *random forest* yang ditunjukkan oleh warna kuning pada gambar, cenderung lebih tinggi dari pada metode lain. Variasi nilai g-mean yang dihasilkan oleh algoritma SMOTE-Bagging *random forest* berkisar antara 51% sampai 52%.

4.8 Hasil Klasifikasi Data Pelamar Bidikmisi

Beasiswa Bidikmisi merupakan beasiswa pemerintahan bagi calon mahasiswa tidak mampu secara ekonomi dan memiliki potensi akademik baik. Dikarenakan beasiswa ini adalah untuk siswa miskin, maka persyaratan utama untuk mendaftar beasiswa jika pendapatan kotor gabungan orangtua/wali dibagi jumlah anggota keluarga sebesar-besarnya Rp 750.000,00 setiap bulannya. Sebelum menuju pembahasan selanjutnya, ada beberapa tahap yang dilakukan untuk mendapatkan nilai-nilai yang disajikan pada Tabel 4.6.

a. Memilih variabel "pendapatan ayah", "pendapatan ibu" dan "jumlah tanggungan keluarga";
b. Membuat variabel baru dengan menghitung pendapatan gabungan orangtua, pada Tabel 4.6 disajikan dengan nama "pendapatan orangtua";
c. Menghitung "pendapatan orangtua" dibagi dengan "jumlah tanggungan keluarga" kemudian menjadi variabel baru "pendapatan per kapita";
d. Koding variabel "pendapatan per kapita" dengan kriteria: Jika pendapatan per kapita > Rp. 750,000.00 maka dikategorikan sebagai keluarga mampu dengan kode prediksi = 0. Jika pendapatan per kapita < Rp. 750,000.00 maka termasuk keluarga miskin dengan kode prediksi = 1;
e. Cocokan dengan variabel respon yaitu "aktual" dengan prediksi pada langkah (d) dan untuk variabel kondisi klasifikasi diisi dengan kriteria: Jika siswa yang menerima beasiswa (Aktual = 1) berasal dari keluarga miskin (Prediksi = 1) maka dikategorikan "Benar". Jika siswa yang tidak menerima beasiswa (Aktual = 0) berasal dari keluarga mampu (Prediksi = 0) maka "Benar". Jika siswa yang menerima beasiswa (Aktual = 1) berasal dari keluarga mampu (Prediksi = 0) maka "Salah".

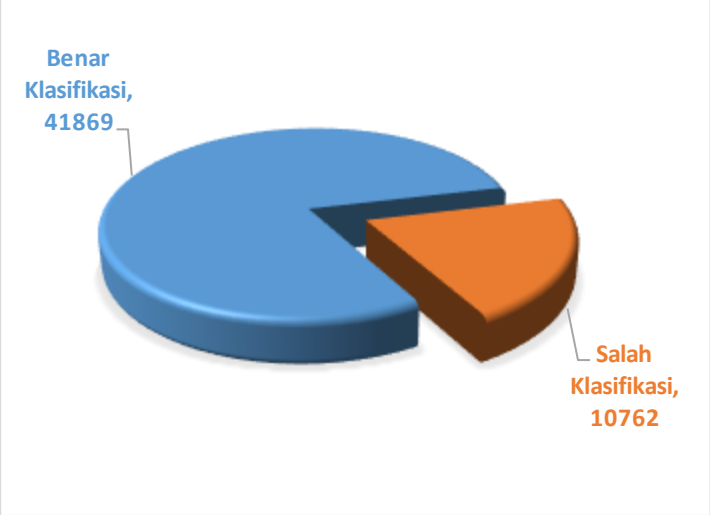
Jika siswa yang tidak menerima beasiswa (Aktual = 0) berasal dari keluarga miskin (Prediksi = 1) maka "Salah".

Tabel 4.6 Identifikasi Kondisi Klasifikasi Data Bidikmisi

Objek/ Siswa	Pendapatan Gabungan Kotor	Pendapatan Gabungan Bersih	Aktual	Prediksi	Kondisi
1	Rp 625,000.00	Rp 156,250.00	1	1	Benar
2	Rp 1,875,000.00	Rp 937,500.00	1	0	Salah
3	Rp 875,000.00	Rp 875,000.00	1	0	Salah
4	Rp 1,375,000.00	Rp 1,375,000.00	1	0	Salah
5	Rp 875,000.00	Rp 175,000.00	1	1	Benar
6	Rp 1,875,000.00	Rp 625,000.00	1	1	Benar
7	Rp 2,750,000.00	Rp 687,500.00	1	1	Benar
8	Rp 875,000.00	Rp 218,750.00	1	1	Benar
9	Rp 1,000,000.00	Rp 500,000.00	1	1	Benar
10	Rp 1,250,000.00	Rp 178,571.43	1	1	Benar

(Sumber: Iriawan dkk, 2018)

Berdasarkan syarat bidikmisi, bahwa siswa yang berhak mendapatkan beasiswa jika pendapatan per kapita sebesar-besarnya adalah Rp 750.000,00. Hasil "prediksi" status penerimaan beasiswa menunjukkan bahwa terdapat 3 siswa yang seharusnya tidak berhak mendapatkan beasiswa, hasil "aktual" menunjukkan bahwa kesepuluh siswa ternyata mempunyai status diterima beasiswa. Berdasarkan Tabel 4.6 dapat dilihat kolom "kondisi".



Gambar 4.18 *Pie Chart* Identifikasi Kondisi Klasifikasi Data Bidikmisi

Secara keseluruhan terdapat 10.762 siswa yang masuk dalam kondisi "salah diklasifikasikan" dan 41.869 siswa yang masuk dalam kondisi "benar

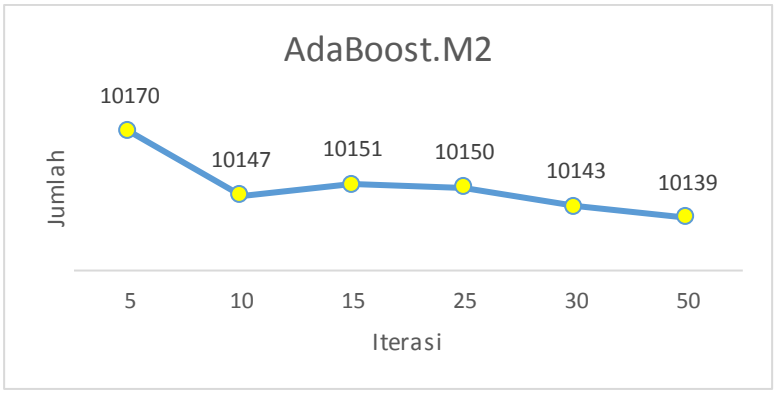
diklasifikasikan" seperti ditunjukkan pada Gambar 4.18 (tabel selengkapnya lihat pada **Lampiran D**).

Penelitian ini menggunakan 3 metode klasifikasi yaitu AdaBoost.M2, SMOTE-Boosting dan SMOTE-Bagging yang digunakan pada data Bidikmisi. Hasil kinerja masing-masing metode telah dijelakan dalam subbab sebelumnya, kemudian hasil prediksi untuk setiap metode ditampilkan lebih jelas pada Tabel 4.7.

Tabel 4.7 Identifikasi Kondisi Klasifikasi Data Bidikmisi dengan Tiga Metode

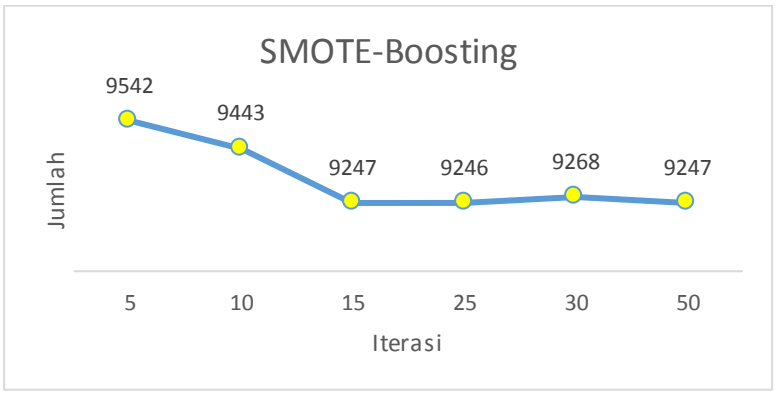
Metode	Iterasi	Jumlah Salah Klasifikasi	Jumlah Benar Klasifikasi	Persentase Benar Klasifikasi
AdaBoost.M2	5	356	10170	96,618
	10	379	10147	96,399
	15	375	10151	96,437
	25	376	10150	96,427
	30	383	10143	96,361
	50	387	10139	96,323
SMOTE-Boosting	5	984	9542	90,652
	10	1083	9443	89,711
	15	1280	9247	87,839
	25	1281	9246	87,830
	30	1259	9268	88,039
	50	1279	9247	87,840
SMOTE-Bagging	5	880	9464	91,639
	10	1192	9334	88,676
	15	1052	9474	90,006
	25	1020	9506	90,309
	30	1123	9403	89,331
	50	1122	9404	89,341

Rata-rata kondisi klasifikasi yang disajikan pada Tabel 4.7 untuk masing-masing metode yaitu AdaBoost.M2, SMOTE-Boosting dan SMOTE-Bagging disajikan selengkapnya di **Lampiran E**. Berdasarkan Tabel 4.7 dapat dilihat perubahan benar klasifikasi untuk masing-masing iterasi yang disajikan pada Gambar 4.19 hingga Gambar 4.21.



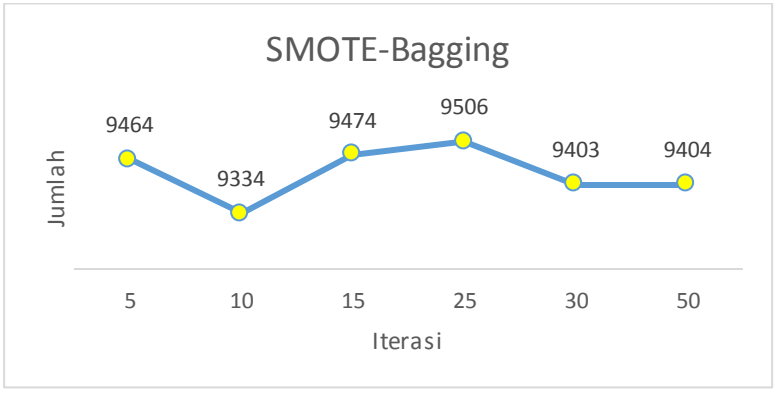
Gambar 4.19 Kondisi "Benar Klasifikasi" Data Bidikmisi Metode AdaBoost.M2

Kondisi metode AdaBoost.M2 yang konvergen pada saat iterasi ke 10 hingga iterasi ke 50, menunjukkan variabilitas data yang kecil untuk metode ini. Dengan rata-rata 10.150 pengamatan masuk ke dalam kelompok benar klasifikasi.



Gambar 4.20 Kondisi "Benar Klasifikasi" Data Bidikmisi Metode SMOTE-Boosting

Gambar 4.20 menunjukkan hasil klasifikasi metode SMOTE-Boosting, dimana kondisi konvergen tercapai pada saat iterasi ke 15 hingga iterasi ke 50, dengan rata-rata 9.332 pengamatan masuk ke dalam kelompok benar klasifikasi.



Gambar 4.21 Kondisi "Benar Klasifikasi" Data Bidikmisi Metode SMOTE-Bagging

Metode SMOTE-Bagging menunjukkan hasil yang berbeda dibandingkan hasil metode sebelumnya. Variabilitas data yang tinggi terlihat berdasarkan fluktuasi plot yang disajikan pada Gambar 4.21 untuk setiap iterasi. Kondisi konvergen tercapai pada saat iterasi terakhir yaitu iterasi ke 30 sampai iterasi ke 50, dengan rata-rata 9.431 pengamatan masuk dalam kelompok benar klasifikasi.

BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Semua model telah dievaluasi dengan menggunakan 5-*fold cross validation*, dan dilihat kriteria kinerja masing-masing metode. Algoritma yang digunakan yaitu *random forest*, SMOTE-Boosting dan SMOTE-Bagging berdasarkan pada integrasi algoritma SMOTE didalam prosedur *boosting* dan *bagging* standar. Kesimpulan yang diperoleh dari penelitian ini sesuai dengan tujuan penelitian adalah sebagai berikut:

1. Akurasi metode *random forest* mencapai optimal ketika prediktor (m) berjumlah 4 dan konvergen pada saat menggunakan 100 pohon (k) klasifikasi.
2. Hasil penelitian dari kelas yang *imbalance* menunjukkan bahwa algoritma *ensemble* SMOTE-Bagging *random forest* dan SMOTE-Boosting *random forest* menunjukkan ketepatan klasifikasi yang cenderung lebih baik dibandingkan metode AdaBoost.M2 *random forest*. Selisih antara kedua metode algoritma SMOTE-Boosting *random forest* dan SMOTE-Bagging *random forest* sangat kecil. Bisa dikatakan bahwa kedua metode tersebut cukup berhasil mengambil keuntungan dari dua algoritma *boosting* dan *bagging* dengan SMOTE. Ketika *boosting* dan *bagging* mempengaruhi akurasi dari *random forest* dengan berfokus pada semua kelas data, algoritma SMOTE merubah nilai kinerja dari *random forest* hanya pada kelas minoritas.

5.2 Saran

Saran yang diberikan untuk penelitian selanjutnya sesuai dengan batasan masalah adalah sebagai berikut:

1. Data yang digunakan adalah data Pelamar Beasiswa Bidikmisi pada tahun 2013 – 2017, namun peneliti mengasumsukan tahun dianggap

sama sehingga untuk penelitian selanjutnya diharapkan tidak mengabaikan tahun yang berbeda.

2. Pada penelitian ini, penanganan data *imbalance* dilakukan dengan SMOTE *Boosting* dan SMOTE *Bagging* yang mana merupakan metode yang berbasis pada *preprocessing* data (*sampling*), sehingga diharapkan pada penelitian selanjutnya dapat dilakukan perbandingan dengan penanganan data *imbalance* menggunakan metode lain yang berbasis pada pendekatan algoritma, seperti fuzzy.

DAFTAR PUSTAKA

Blake, C., dan Merz, C. (1998). *UCI Repository of Machine Learning Databases http://www.ics.uci.edu/~mlearn/~MLRepository.html.* Department of Information and Computer Sciences, University of California, Irvine.

Bolón-Canedo, V., Sánchez-Maroño, N., dan Alonso-Betanzos, A. (2015). *Feature selection for high-dimensional data*. Springer.

Boyd, S., dan Vandenberghe, L. (2004). *Convex Optimization. Cambridge University Press*, Cambridge, USA.

Breiman, L., Friedman, J. H., Olshen, R., dan Stone, C. (1983). *Classification and Regression Trees*. Wadsworth.

Breiman, L. (1996). *Bagging Predictors*. Machine Learning, pp. 123-140.

Breiman, L. (2001). *Random Forests*. Machine Learning, Vol. 45, pp. 5-32.

Burges, C. (1998). *A Tutorial On Support Vector Machine for Pattern Recognition*. Data Mining and Knowledge Discovery, Vol. 2, No. 2, pp. 955-974.

Cahyono, A. B. (2010). *Analisis Pemanfaatan Small Disjunct Pada Decision Tree Dengan Algoritma Genetika*. Jurnal Seminar Nasional Aplikasi Teknologi Informasi.

Chau, V. T. N., dan Phung, N. H. (2013). *Imbalance Educational Data Classification: An Effective Approach wirh resampling and Random Forest*. Computing and Communication Technologies, Research, Innovation, and Vision for The Future (RIVF).

Chawla, N.V., Lazarevic, A., Hall, L.O dan Bowyer, K.W., (2002). *SMOTEBoost: Improving prediction of the minority class in boosting*. Proc. Knowl. Discov. Databases, pp. 107–119.

Chen, P. H., Lin, C.J dan Scholkopf, B. (2005), *A Tutuorial on v-Support Vector Machines*. Applied Stochastic Model in Business and Industry, Vol 21, Hal. 111-136.

Choi, J. (2010). *A* Selective *Sampling Method for Imbalance Data Learning on Support Vector Machines*. Graduate Theses and Dissertations, Paper 11529.

Breiman, L., dan Cutler, A. (2003). *Manual on Setting Up, Using and Understanding Random Forest V4.0*.

Dahri, D., Agus, F., dan Khairina, D. M. (2016). *Metode Naïve Bayes untuk Penentuan* Penerima *Beasiswa Bidikmisi Universitas Mulawarman*. Jurnal Informatika Mulawarman, vol.11, no.2.

Direktorat Jenderal Pembelajaran dan Kemahasiswaan, K.R.T.d.P.T. (2017). *Pedoman Penyelenggaraan Bantuan Biaya Pendidikan Bidikmisi Tahun 2017*. Belmawa, Kemeristek Dikti, Jakarta.

Direktorat Jenderal Pembelajaran dan Kemahasiswaan, K.R.T.d.P.T. (2018). *Panduan Bidikmisi 2018, Kemeristek Dikti*. Jakarta.

Drummond, C., dan Holte, R.C. (2003). *C4.5 Class Imbalance and Cost Sensitivity: Why Undersampling beats Oversampling*. Institute for Information Technology, National Research Council (pp. 1-8). Canada, Ottawa, Ontario: Department of Computing Science, University of Alberta.

Freund, Y., dan Schapire, R. E. (1996). *Experiments with a New Boosting Algorithm*. Machine Learning: Proceedings of the Thirteenth International Conference. Morgan Kaufmann, Italy.

Freund, Y., dan Schapire, R. E. (1995). *A desicion-theoretic generalization of on-line learning and an application to boosting*. In European conference on computational learning theory (pp. 23-37). Springer, Berlin, Heidelberg.

Friedman, J.H. (1999). *Stochastic Gradient Boosting*. North Ryde NSW.

Galar, M., Fernandes, A., Barrenechea, E., dan Bustince, H., (2011). *A Review on Ensemble for the Class Imbalance Problem: Bagging, Boosting, and Hybrid-Based Approaches*. IEEE Transactions on System, Man and Cybernetics, Part C (Applications and Review), Vol. 42, Hal. 463-484.

Gunn, S. (1998). *Support vector Machines for Classification and Regression*. *Technical Report*, ISIS.

Hair. J.F. (1995). *Multivariate Data Analisis, Prentice-Hall International*. Inc, U.S.A-Mexico-Canada.

Haerdle, W. K., Prastyo, D. D., dan Hafner, C. M. (2014). *Support Vector Machines with Evolutionary Model Selection for Default Prediction in The Oxford Handbook of Applied Nonparametric and Semiparametric Econometrics and Statistics, eds. Racine*. JS, Su, L, and Ullah, A, Oxford University Press, 346-373.

Han, J., Kamber, dan M., Pei, J. (2006). *Data Mining Concepts and Techniques 2nd Edition*. Kaufman Publisher, USA.

Han, J., Kamber, dan M., Pei, J. (2012). *Data Mining Concepts and Techniques 3rd Edition*. Kaufman Publisher, USA.

Hand, D. J., dan Till, R. J. 2001. *A Simple Generalisation of the Area Under the ROC Curve for Multiple Class* Classification *Problems*. Machine Learning 45:171–186.

Hastie, T.J., Tibshirani, R.J., dan Friedman, J.H. (2008). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Second Edirion. New York: Springer – Verleg.

Hsu, C.W, Chang, C.C., dan Lin, C.J. (2004). *A Practical Guide to Support Vector Classification*. Department of Computer Scinece an Information Engineering, National Taiwan University.

Iriawan, N., Fihriasari, K., Ulama, B, S, S., Suryaningtyas, W., Susanto, I., dan Pravitasari, A. A. (2018). *Bayesian Bernoulli mixture regression model for Bidikmisi scholarship classification*. Jurnal Ilmu Komputer dan Informasi. Vol. 11. Pp. 67 – 76. UI Depok.

Japkowicz, N., dan Stephan, S. (2002). *The Class Imbalance Problem: A Systematic Study*. Intelligent Data Analysis, pp. 203-231.

Kim, M. J., Kang, D. K., dan Kim, H. B. (2015). *Geometric mean based boosting algorithm with over-sampling to resolve data imbalance problem for bankruptcy prediction*. Expert Systems with Applications, 42(3), pp. 1074-1082.

Kotsianis, S. B., Kanellopoulos, D., dan Pintelas, P. (2006). *Handling Imbalance Dataset: A Review*. GESTS International Transaction on Computer Science ang Engineering, 25-36.

Kubat, M., dan Matwin, S., (1997). *Addressing the curse of imbalanced training sets: one-sided selection*. Fourteenth Internation Conference on Machine Learning, Conference Proceedings. pp. 179–186.

Kumar, R., dan Sharma, A., (2016). *Comparative Analysis of SVM & kNN for Academic Prediction of Students*. International Journal of IT & Knowledge Management, vol.10, no.1, pp. 15-19.

Li, X., Wang, L., dan Sung, E. (2008). *AdaBoost with SVM-based component classifiers*. Engineering Applications of Artificial Intelligence, 21(5), 785-795.

Liaw, A., dan Wiener, M. (2002). *Classification and regression by Random Forest, R News*. Vol 2, pp. 18 – 22.

Liu, P., Cai, L., Wang, Y., dan Zhan, L. (2010). *Classifying Skewed Data Streams Based on Reusing Data*. International Conference on Computer Application and System Modeling (ICCASM).

Malohlava, M., dan Candel, A. (2018). *A desicion-theoretic generalization of on-line learning and an application to boosting*. In European conference on computational learning theory (pp. 23-37). Springer, Berlin, Heidelberg.

Marquez-Vera, C., Cano, A., Romero, C., dan Ventura, S. (2013). *Predicting student failure at school using genetic programming and different data mining approaches with high dimensional and imbalance data*. Applied Intelligence, 315-330.

Nitesh, V. C., Japkowicz, N., dan Kolcz, A. (2004). *Special Issue on Learning from Imbalance Data Sets*. SIGKDD Explorations, Volume 6, Issue 1, 1-6.

Quinlan, J. R. (1993). *C4.5 : Programs For Machine Learning*. San Mateo, California: Morgan Kaufmann.

Quinlan, J. R. (1996). *Bagging, Boosting, and C4.5*. In Proceedings of the Thirteenth National Conference on Artificial Intelligence, 725-730.

Robandi, I dan Prasetyo G, R.A. (2008). *Peramalan Beban Jangka Pendek Untuk Hari-hari Libur Dengan Metode Support Vector Machine*. Tugas Akhir. ITS, Surabaya.

Sain, H., dan Purnami, S.W. (2015). *Combine Sampling Support Vector Machine for Imbalance Data Classification*. Procedia Computer Science, 72, 771-780.

Santosa, B. (2007). *Data Mining*: Teknik Pemanfaatan Data Untuk Keperluan Bisnis, Teori dan Aplikasi, Graha Ilmu.

Schapire,R.E., dan Freund, Y. (1999). *Gradient Boosting Machine with H20 7th Edition*. H2O.ai, Inc. USA.

Schapire, R.E., dan Freund, Y. (2012). *Boosing : Foundations and Algorithms.* London: MIT Press.

Secholkopf, B dan A. Simola. (2002). *Learning with Kernel : Support Vector Machines, Regulerizatiom, Optimization, and Beyond*. Cambridge, MA : MIT Press.

Sokolova, M., dan Lapalme, G. (2009). *A systematic analysis of peformance measures for classification task*. Information Processing Management 45:427 – 437.

Susilala, R. M., Kuswanto, H., dan Fithriasari, K. (2015). *The Comparison of Classical and Bayesian Bivariate Binary Logistic Regression Prediction for Unbalanced Response: Case Study Customers of Antivirus Software Perusahaan 'X'*. International Conference on Science, Technology and Humanity, 206-216.

Tan, P. N., Steinbach, M., dan Kumar, V. (2006). *Introduction to Data Mining* (4th ed.). Pearson Addison Wesley, Boston.

Tang, K., Wang, R., dan Chen, T. (2011). *Towards Maximizing the Area Under the ROC Curve for Multi-Class Classification Problems*. Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence.

Vapnik, V., dan Cortes, C. (1995). *Support Vector Networks*. Machine Learning, Volume 20, 273-297.

Virag, M., dan Nyitrai, T. (2014). *The Application of Ensemble Methods in Forecasting Bankruptcy*. Financial and Economic Review, Volume 13, Issue 4, 178 – 193.

Wang, S dan Yao, X., (2009). *Diversity analysis on imbalanced data sets by using ensemble models*. IEEE Symp. Comput. Intell. Data Mining, pp. 324–331.

Wickramaratna, J., Holden, S., dan Buxton, B. (2001). *Performance degradation in boosting*. Multiple Classifier Systems, 11-21.

Zhou, Z. H., dan Liu, X. Y. (2006). *Training cost-sensitive neural networks with methods addressing* the *class imbalance problem*. IEEE Transaction Knowledge Data Engineering, vol.18, no.1, pp. 63–77.

Zhu, M. (2008). *Kernels and Ensembles: Perspective on Statistical Learning*. The American Statistician, pp. 97 – 109.

DAFTAR LAMPIRAN

LAMPIRAN A1. *Confusion Matrix* AdaBoost.M2

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 5	Tidak Diterima	0	367	367
	Diterima	3	10156	10159
	Total	3	10523	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 10	Tidak Diterima	1	366	367
	Diterima	25	10134	10159
	Total	26	10500	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 15	Tidak Diterima	1	367	368
	Diterima	22	10137	10159
	Total	23	10504	10527

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 25	Tidak Diterima	1	366	367
	Diterima	22	10137	10159
	Total	23	10503	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 30	Tidak Diterima	2	367	369
	Diterima	30	10129	10159
	Total	32	10496	10528

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 50	Tidak Diterima	3	364	367
	Diterima	31	10128	10159
	Total	34	10492	10526

LAMPIRAN A2. *Confusion Matrix* SMOTE-Boosting

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 5	Tidak Diterima	27	340	367
	Diterima	634	9525	10159
	Total	661	9865	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 10	Tidak Diterima	32	335	367
	Diterima	732	9427	10159
	Total	764	9762	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 15	Tidak Diterima	32	335	367
	Diterima	961	9199	10160
	Total	993	9534	10527

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 25	Tidak Diterima	37	330	367
	Diterima	959	9201	10160
	Total	996	9531	10527

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 30	Tidak Diterima	36	331	367
	Diterima	940	9220	10160
	Total	976	9551	10527

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 25	Tidak Diterima	33	334	367
	Diterima	953	9206	10159
	Total	986	9540	10526

LAMPIRAN A3. *Confusion Matrix* SMOTE-Bagging

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 5	Tidak Diterima	27	340	367
	Diterima	536	9623	10159
	Total	563	9963	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 10	Tidak Diterima	44	323	367
	Diterima	859	9300	10159
	Total	903	9623	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 15	Tidak Diterima	37	330	367
	Diterima	712	9447	10159
	Total	749	9777	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 25	Tidak Diterima	36	331	367
	Diterima	681	9478	10159
	Total	717	9809	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 30	Tidak Diterima	42	325	367
	Diterima	784	9375	10159
	Total	826	9700	10526

	Actual Class	Prediction Class		Total
		Tidak Diterima	Diterima	
Iterasi 50	Tidak Diterima	43	324	367
	Diterima	760	9399	10159
	Total	803	9723	10526

LAMPIRAN B1. Perhitungan kinerja (*performamce*) metode AdaBoost.M2

$$Presisi = \frac{TP}{(TP + FP)} \qquad Recall = \frac{TP}{(TP + FN)}$$

$$F\text{-}Value = \frac{2(\mathrm{Re}call \times \mathrm{Pr}esisi)}{(\mathrm{Re}call + \mathrm{Pr}esisi)}$$

$$Specificity = \frac{TN}{(TN + FP)} \times 100\% \qquad Sensitivity = \frac{TP}{(TP + FN)} \times 100\%$$

$$G\text{-}Mean = \sqrt{Sensitivity \times Specificity}$$

$$TPR = \frac{TP}{(TP + FN)} \qquad FPR = 1 - \frac{TN}{(TN + FP)}$$

$$\mathrm{AUC} = \frac{1 + TPR - FPR}{2}$$

Kinerja Metode	Iterasi					
	5	10	15	25	30	50
Presisi	0,000	0,038462	0,043478	0,043478	0,0625	0,088235
Recall	0,000	0,002725	0,002717	0,002725	0,00542	0,008174
F-Value	0,000	0,005089	0,005115	0,005128	0,009975	0,014963
Specificity	0,999	0,997539	0,997834	0,997834	0,997047	0,996949
Sensitivity	0,000	0,002725	0,002717	0,002725	0,00542	0,008174
G-Mean	0,000	0,052135	0,052072	0,052143	0,073512	0,090274
TPR	0,000	0,002725	0,002717	0,002725	0,00542	0,008174
FPR	0,0002	0,002461	0,002166	0,002166	0,002953	0,003051
AUC	0,499	0,500132	0,500276	0,50028	0,501234	0,502561

LAMPIRAN B2. Perhitungan manual kinerja (*performance*) metode SMOTE-Boosting

$$Presisi = \frac{TP}{(TP + FP)} \qquad Recall = \frac{TP}{(TP + FN)}$$

$$F\text{-}Value = \frac{2(\mathrm{Re}call \times \mathrm{Pr}esisi)}{(\mathrm{Re}call + \mathrm{Pr}esisi)}$$

$$Specificity = \frac{TN}{(TN + FP)} \times 100\% \qquad Sensitivity = \frac{TP}{(TP + FN)} \times 100\%$$

$$G\text{-}Mean = \sqrt{Sensitivity \times Specificity}$$

$$TPR = \frac{TP}{(TP + FN)} \qquad FPR = 1 - \frac{TN}{(TN + FP)}$$

$$\mathrm{AUC} = \frac{1 + TPR - FPR}{2}$$

Kinerja Metode	Iterasi					
	5	10	15	25	30	50
Presisi	0.040847	0.041885	0.032226	0.037149	0.036885	0.033469
Recall	0.073569	0.087193	0.087193	0.100817	0.098093	0.089918
F-Value	0.052529	0.056587	0.047059	0.054292	0.053611	0.04878
Specificity	0.937592	0.927946	0.905413	0.90561	0.90748	0.906192
Sensitivity	0.073569	0.087193	0.087193	0.100817	0.098093	0.089918
G-Mean	0.262637	0.284448	0.280974	0.302161	0.298357	0.285453
TPR	0.073569	0.087193	0.087193	0.100817	0.098093	0.089918
FPR	0.062408	0.072054	0.094587	0.09439	0.09252	0.093808
AUC	0.505581	0.50757	0.496303	0.503214	0.502786	0.498055

LAMPIRAN B3. Perhitungan manual kinerja (*performance*) metode SMOTE-Bagging

$$Presisi = \frac{TP}{(TP + FP)} \qquad Recall = \frac{TP}{(TP + FN)}$$

$$F\text{-}Value = \frac{2(\mathrm{Re}\,call \times \mathrm{Pr}\,esisi)}{(\mathrm{Re}\,call + \mathrm{Pr}\,esisi)}$$

$$Specificity = \frac{TN}{(TN + FP)} \times 100\% \qquad Sensitivity = \frac{TP}{(TP + FN)} \times 100\%$$

$$G\text{-}Mean = \sqrt{Sensitivity \times Specificity}$$

$$TPR = \frac{TP}{(TP + FN)} \qquad FPR = 1 - \frac{TN}{(TN + FP)}$$

$$\mathrm{AUC} = \frac{1 + TPR - FPR}{2}$$

Kinerja Metode	Iterasi					
	5	10	15	25	30	50
Presisi	0.047957	0.048726	0.049399	0.050209	0.050847	0.053549
Recall	0.073569	0.119891	0.100817	0.098093	0.114441	0.117166
F-Value	0.058065	0.069291	0.066308	0.066421	0.070411	0.073504
Specificity	0.947239	0.915444	0.929914	0.932966	0.922827	0.925189
Sensitivity	0.073569	0.119891	0.100817	0.098093	0.114441	0.117166
G-Mean	0.263985	0.331291	0.306189	0.302518	0.324976	0.329243
TPR	0.073569	0.119891	0.100817	0.098093	0.114441	0.117166
FPR	0.052761	0.084556	0.070086	0.067034	0.077173	0.074811
AUC	0.510404	0.517668	0.515366	0.515529	0.518634	0.521178

LAMPIRAN C1. Syntax AdaBoost.M2 package ebmc di R

```
adam2=function (formula, data, size, alg, rf.ntree = 100)
{
    target <- gsub(" ", "", unlist(strsplit(format(formula),
        split = "~"))[1])
    list_model <- list()
    a <- 0
    w <- rep(1/nrow(data), nrow(data))
    label <- data[, target]
    for (i in 1:size) {
        if (i == 1) {
          samp <- sample(nrow(data), nrow(data), replace = FALSE)
        }
        else if (i != 1) {
          samp <- sample(nrow(data), nrow(data), replace = TRUE,
                prob = w)
        }
        train <- data[samp, ]
        if (alg == "rf") {
          list_model[[i]] <- randomForest::randomForest(formula,
                data = train, ntree = rf.ntree)
            prob <- as.data.frame(predict(list_model[[i]], data,
                type = "prob"))
        }
        pred <- as.factor(ifelse(prob[, "1"] >= 0.5, 1, 0))
        new <- .wt.update(probability = prob, prediction = pred,
            actual = label, wt = w, smooth = 1/nrow(data))
        w <- new[[1]]
        a[i] <- new[[2]]
    }
    result <- list(weakLearners = list_model, errorEstimation=a)
    attr(result, "class") <- "modelBst"
    return(result)
}
```

LAMPIRAN C2. Syntax SMOTE-Boosting package ebmc di R

```
Sbo=function (formula, data, size, alg, over, rf.ntree = 100)
{
    target <- gsub(" ", "", unlist(strsplit(format(formula),
        split = "~"))[1])
    list_model <- list()
    a <- 0
    n <- data[which(data[, target] == "0"), ]
    p <- data[which(data[, target] == "1"), ]
    data$w <- rep(1/nrow(data), nrow(data))
    label <- data[, target]
    for (i in 1:size) {
    n <- data[which(data[, target] == "0"), ]
    f <- reformulate(paste(colnames(data)[which(colnames(data) !=
            target & colnames(data) != "w")], collapse = "+"),
            response = target)
        smote <- DMwR::SMOTE(f, data = data, perc.over = over,
            perc.under = 0)
        train <- rbind(n, smote)
        train$w <- train$w/sum(train$w)
        train <- train[sample(nrow(train), nrow(train), replace =
        TRUE, prob = train$w), ]
        train$w <- NULL
        if (alg == "rf") {
           list_model[[i]] <- randomForest::randomForest(formula,
                data = train, ntree = rf.ntree)
            prob <- as.data.frame(predict(list_model[[i]], data,
                type = "prob"))
        }
        pred <- as.factor(ifelse(prob[, "1"] >= 0.5, 1, 0))
        new <- .wt.update(probability = prob, prediction = pred,
            actual = label, wt = data$w, smooth = 1/nrow(data))
        data$w <- new[[1]]
        a[i] <- new[[2]]

    result <-list(weakLearners = list_model, errorEstimation = a)
    attr(result, "class") <- "modelBst"
    return(result)
}
```

LAMPIRAN C3. Syntax SMOTE-Bagging package ebmc di R

```
Sbag=function (formula, data, size, alg, over, rf.ntree = 100)
{
    target <- gsub(" ", "", unlist(strsplit(format(formula),
        split = "~"))[1])
    list_train <- list()
    list_model <- list()
    b <- rep(seq(10, 100, 10), (size%/%10 + 1))
    b <- b[1:size]
    n <- data[which(data[, target] == "0"), ]
    p <- data[which(data[, target] == "1"), ]
    smote <- DMwR::SMOTE(reformulate(".", response = target),
        data = data, perc.over = over, perc.under = 0)
    smote <- rbind(smote, p)
    smote <- smote[which(!(duplicated(smote) | duplicated(smote,
        fromLast = TRUE) == TRUE)), ]
    for (i in 1:size) {
        resamp_rate <- nrow(n)/nrow(p) * (b[i]/100)
        resamp <- p[sample(nrow(p), round(nrow(p) * resamp_rate),
            replace = TRUE), ]
        smt <-smote[sample(nrow(smote), (nrow(n) - nrow(resamp)),
            replace = TRUE), ]
        list_train[[i]] <- rbind(n, resamp, smt)
    }
    for (i in 1:size) {
        if (alg == "rf") {
        list_model[[i]] <- randomForest::randomForest(formula,
                data = list_train[[i]], ntree = rf.ntree)
        }
    attr(list_model, "class") <- "modelBag"
    return(list_model)
}
```

LAMPIRAN C4. Syntax klasifikasi pada data Bidikmisi

```
######### membaca data #############
datasv=read.csv("H:Mydata Edu 2 Class.csv",header=TRUE)
datasv$Status <- factor(datasv$Status, levels = c("1", "0"),
labels = c("0", "1"))
datasv[,'Status']=as.factor(datasv[,'Status'])
for(i in 1:10){
  datasv[,i]=as.factor(datasv[,i])}
str(datasv)
summary(datasv)

####### library yang digunakan ###########
library(caret)
library(ebmc)
library(rminer)
library(data.table)
library(MLmetrics)
library(dplyr)
library(randomForest)
library(pROC)
library(ROCR)
library(AUCRF)

############################ adam2 rf ############################
set.seed(123)
folds <- createFolds(factor(datasv$Status), k = 5, list = FALSE)
time.adam2rf=rep(0,5)
fbm.adam2rf=rep(0,5)
aucbm.adam2rf=rep(0,5)
gmeanbm.adam2rf=rep(0,5)
tprbm.adam2rf=rep(0,5)
tnrbm.adam2rf=rep(0,5)
akurasibm.adam2rf=rep(0,5)
for(i in 1:5)
{
  print(i)
  testIndexes <- which(folds==i,arr.ind=TRUE)
  testData <- datasv[testIndexes, ]
  trainData <- datasv[-testIndexes, ]
  start.time = Sys.time()
```

```
  modelbm.adam2rf <- adam2(Status ~ ., data = trainData, size =
5, alg = "rf", rf.ntree=100)
  end.time = Sys.time()
  probbm.adam2rf <- predict(modelbm.adam2rf, newdata = testData,
type = "prob")
  predbm.adam2rf <- predict(modelbm.adam2rf, newdata = testData,
type = "class")
  aucbm.adam2rf[i] <- measure(label = testData$Status,
probability = probbm.adam2rf, metric = "auc")
  gmeanbm.adam2rf[i] <- measure(label = testData$Status,
probability = probbm.adam2rf, metric = "gmean", threshold = 0.5)
  tprbm.adam2rf[i] <- measure(label = testData$Status,
probability = probbm.adam2rf, metric = "tpr", threshold = 0.5)
  tnrbm.adam2rf[i] <- measure(label = testData$Status,
probability = probbm.adam2rf, metric = "tnr", threshold = 0.5)
  fbm.adam2rf[i] <- measure(label = testData$Status, probability
= probbm.adam2rf, metric = "f", threshold = 0.5)
  akurasibm.adam2rf[i] <-
Accuracy(testData$Status,predbm.adam2rf)
  time.adam2rf[i]=end.time-start.time
  conf.mbm.adam2rf <- table(testData$Status,predbm.adam2rf)
  error.adam2rf <- 1-
sum(diag(conf.mbm.adam2rf)/sum(conf.mbm.adam2rf))
}


######################## smoteboost_rf ##########################
folds <- createFolds(factor(datasv$Status), k = 5, list = FALSE)
time.sborf=rep(0,5)
fbm.sborf=rep(0,5)
aucbm.sborf=rep(0,5)
gmeanbm.sborf=rep(0,5)
tprbm.sborf=rep(0,5)
tnrbm.sborf=rep(0,5)
akurasibm.sborf=rep(0,5)
akurasi.sborf=rep(0,5)
set.seed(1235)
for(i in 1:5)
{
  print(i)
  testIndexes <- which(folds==i,arr.ind=TRUE)
  testData <- datasv[testIndexes, ]
  trainData <- datasv[-testIndexes, ]
  start.time = Sys.time()
```

```
  modelbm.sborf <- sbo(Status ~ ., data = trainData, size = 5,
over = 2400, alg = "rf", rf.ntree = 100)
  end.time = Sys.time()
  probbm.sborf <- predict(modelbm.sborf, newdata = testData, type
= "prob")
  predbm.sborf <- predict(modelbm.sborf, newdata = testData, type
= "class")
  aucbm.sborf[i] <- measure(label = testData$Status, probability
= probbm.sborf, metric = "auc", threshold = 0.5)
  gmeanbm.sborf[i] <- measure(label = testData$Status,
probability = probbm.sborf, metric = "gmean", threshold = 0.5)
  tprbm.sborf[i] <- measure(label = testData$Status, probability
= probbm.sborf, metric = "tpr", threshold = 0.5)
  tnrbm.sborf[i] <- measure(label = testData$Status, probability
= probbm.sborf, metric = "tnr", threshold = 0.5)
  fbm.sborf[i] <- measure(label = testData$Status, probability =
probbm.sborf, metric = "f", threshold = 0.5)
  akurasi.sborf[i] <- measure(label = testData$Status,
probability = probbm.sborf, metric = "acc", threshold = 0.5)
  akurasibm.sborf[i] <- Accuracy(testData$Status,predbm.sborf)
  time.sborf[i]=end.time-start.time
  conf.mbm.sborf<- table(testData$Status,predbm.sborf)
  error.sborf <- 1-sum(diag(conf.mbm.sborf)/sum(conf.mbm.sborf))
}

######################## smotebagging_rf ########################
folds <- createFolds(factor(datasv$Status), k = 5, list = FALSE)
time.sbgrf=rep(0,5)
fbm.sbgrf=rep(0,5)
aucbm.sbgrf=rep(0,5)
gmeanbm.sbgrf=rep(0,5)
tprbm.sbgrf=rep(0,5)
tnrbm.sbgrf=rep(0,5)
akurasibm.sbgrf=rep(0,5)
akurasi.sbgrf=rep(0,5)
set.seed(123)
for(i in 1:5)
{
  print(1)
  testIndexes <- which(folds==1,arr.ind=TRUE)
  testData <- datasv[testIndexes, ]
  trainData <- datasv[-testIndexes, ]
  start.time = Sys.time()
```

```
  modelbm.sbgrf <- sbag(Status ~ ., data = trainData, size = 5,
alg = "rf", over= 2400, rf.ntree = 100)
  end.time = Sys.time()
  probbm.sbgrf <- predict(modelbm.sbgrf, newdata = testData, type
= "prob")
  predbm.sbgrf <- predict(modelbm.sbgrf, newdata = testData, type
= "class")
  aucbm.sbgrf[1] <- measure(testData$Status, probbm.sbgrf, metric
= "auc", threshold = 0.5)
  gmeanbm.sbgrf[1] <- measure(label = testData$Status,
probability = probbm.sbgrf, metric = "gmean", threshold = 0.5)
  tprbm.sbgrf[i] <- measure(label = testData$Status, probability
= probbm.sbgrf, metric = "tpr", threshold = 0.5)
  tnrbm.sbgrf[i] <- measure(label = testData$Status, probability
= probbm.sbgrf, metric = "tnr", threshold = 0.5)
  fbm.sbgrf[i] <- measure(label = testData$Status, probability =
probbm.sbgrf, metric = "f", threshold = 0.5)
  akurasi.sbgrf[i] <- measure(label = testData$Status,
probability = probbm.sbgrf, metric = "acc", threshold = 0.5)
  akurasibm.sbgrf[i] <- Accuracy(testData$Status,predbm.sbgrf)
  time.sbgrf[i]=end.time-start.time
  conf.mbm.sbgrf<- table(testData$Status,predbm.sbgrf)
  error.sbgrf <- 1-sum(diag(conf.mbm.sbgrf)/sum(conf.mbm.sbgrf))
}
```

LAMPIRAN D. Identifikasi Kondisi Klasifikasi Data Bidikmisi

Objek/ Siswa	Pendapatan Gabungan Kotor	Pendapatan Gabungan Bersih	Prediksi	Aktual	Kondisi
1	Rp 625,000.00	Rp 156,250.00	1	1	Benar
2	Rp 1,875,000.00	Rp 937,500.00	0	1	Salah
3	Rp 875,000.00	Rp 875,000.00	0	1	Salah
4	Rp 1,375,000.00	Rp 1,375,000.00	0	1	Salah
5	Rp 875,000.00	Rp 175,000.00	1	1	Benar
6	Rp 1,875,000.00	Rp 625,000.00	1	1	Benar
7	Rp 2,750,000.00	Rp 687,500.00	1	1	Benar
8	Rp 875,000.00	Rp 218,750.00	1	1	Benar
9	Rp 1,000,000.00	Rp 500,000.00	1	1	Benar
10	Rp 1,250,000.00	Rp 178,571.43	1	1	Benar
11	Rp 750,000.00	Rp 375,000.00	1	1	Benar
12	Rp 750,000.00	Rp 750,000.00	1	1	Benar
13	Rp 1,125,000.00	Rp 225,000.00	1	1	Benar
14	Rp 875,000.00	Rp 291,666.67	1	1	Benar
15	Rp 1,000,000.00	Rp 333,333.33	1	1	Benar
16	Rp 1,375,000.00	Rp 687,500.00	1	1	Benar
17	Rp 625,000.00	Rp 625,000.00	1	1	Benar
18	Rp 1,500,000.00	Rp 750,000.00	1	1	Benar
19	Rp 1,375,000.00	Rp 343,750.00	1	1	Benar
20	Rp 375,000.00	Rp 375,000.00	1	1	Benar
21	Rp 125,000.00	Rp 31,250.00	1	1	Benar
22	Rp 125,000.00	Rp 41,666.67	1	1	Benar
23	Rp 1,875,000.00	Rp 625,000.00	1	1	Benar
24	Rp 875,000.00	Rp 175,000.00	1	1	Benar
25	Rp 875,000.00	Rp 437,500.00	1	1	Benar
26	Rp 375,000.00	Rp 375,000.00	1	1	Benar
27	Rp 1,250,000.00	Rp 312,500.00	1	1	Benar
28	Rp 1,000,000.00	Rp 250,000.00	1	1	Benar
29	Rp 1,750,000.00	Rp 875,000.00	0	1	Salah
30	Rp 875,000.00	Rp 875,000.00	0	1	Salah
⋮	⋮	⋮	⋮	⋮	⋮
52589	Rp 2,125,000.00	Rp 708,333.33	1	0	Salah
52590	Rp 125,000.00	Rp 15,620.00	1	1	Benar
⋮	⋮	⋮	⋮	⋮	⋮
52628	Rp -	Rp -	1	1	Benar
52629	Rp 2,125,000.00	Rp 531,250.00	1	1	Benar
52630	Rp 1,875,000.00	Rp 937,500.00	1	0	Salah
52631	Rp 1,375,000.00	Rp 687,500.00	0	1	Salah

LAMPIRAN D1. Tabel hasil prediksi metode AdaBoost.M2 Iterasi ke-5

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
356
Jumlah benar klasifikasi
10170
Persentase benar klasifikasi
$\frac{10170}{356+10170} * 100 = 96{,}618\%$

LAMPIRAN D2. Lanjutan Tabel hasil prediksi metode AdaBoost.M2 Iterasi ke-10

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
379
Jumlah benar klasifikasi
10147
Persentase benar klasifikasi
$\frac{10147}{379 + 10147} * 100 = 96{,}399\%$

LAMPIRAN D3. Tabel hasil prediksi metode AdaBoost.M2 Iterasi ke-15

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
375
Jumlah benar klasifikasi
10151
Persentase benar klasifikasi
$\frac{10151}{375 + 10151} * 100 = 96,437\%$

LAMPIRAN D4. Tabel hasil prediksi metode AdaBoost.M2 Iterasi ke-25

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
376
Jumlah benar klasifikasi
10150
Persentase benar klasifikasi
$\frac{10150}{376+10150} * 100 = 96{,}427\%$

LAMPIRAN D5. Tabel hasil prediksi metode AdaBoost.M2 Iterasi ke-30

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
383
Jumlah benar klasifikasi
10143
Persentase benar klasifikasi
$\frac{10143}{383 + 10143} * 100 = 96{,}361\%$

LAMPIRAN D6. Tabel hasil prediksi metode AdaBoost.M2 Iterasi ke-50

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
387
Jumlah benar klasifikasi
10139
Persentase benar klasifikasi
$\frac{10143}{383 + 10143} * 100 = 96,323\%$

LAMPIRAN D7. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-5

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
984
Jumlah benar klasifikasi
9542
Persentase benar klasifikasi
$\frac{9542}{984 + 9542} * 100 = 90{,}652\%$

LAMPIRAN D8. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-10

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1083
Jumlah benar klasifikasi
9443
Persentase benar klasifikasi
$\frac{9443}{1083 + 9443} * 100 = 89{,}711\%$

LAMPIRAN D9. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-15

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1280
Jumlah benar klasifikasi
9247
Persentase benar klasifikasi
$\frac{9247}{1280+9247} * 100 = 87{,}839\%$

LAMPIRAN D10. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-25

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1281
Jumlah benar klasifikasi
9246
Persentase benar klasifikasi
$\frac{9246}{1281+9246} * 100 = 87{,}830\%$

LAMPIRAN D11. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-30

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1259
Jumlah benar klasifikasi
9268
Persentase benar klasifikasi
$\frac{9268}{1259+9268} * 100 = 88{,}039\%$

LAMPIRAN D12. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-50

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1279
Jumlah benar klasifikasi
9247
Persentase benar klasifikasi
$\frac{9247}{1279+9247} * 100 = 87{,}849\%$

LAMPIRAN D13. Tabel hasil prediksi metode SMOTE-Bagging Iterasi ke-5

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
880
Jumlah benar klasifikasi
9646
Persentase benar klasifikasi
$\frac{9247}{1279+9247} * 100 = 91{,}639\%$

LAMPIRAN D14. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-10

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Salah Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1192
Jumlah benar klasifikasi
9334
Persentase benar klasifikasi
$\frac{9334}{1192+9334} * 100 = 88{,}676\%$

LAMPIRAN D15. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-15

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1052
Jumlah benar klasifikasi
9474
Persentase benar klasifikasi
$\frac{9474}{1052+9474} * 100 = 90{,}006\%$

LAMPIRAN D16. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-25

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1020
Jumlah benar klasifikasi
9506
Persentase benar klasifikasi
$\frac{9506}{1020 + 9506} * 100 = 90{,}309\%$

LAMPIRAN D17. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-30

Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1123
Jumlah benar klasifikasi
9403
Persentase benar klasifikasi
$\frac{9403}{1123+9403} * 100 = 89{,}331\%$

LAMPIRAN D18. Tabel hasil prediksi metode SMOTE-Boost Iterasi ke-50

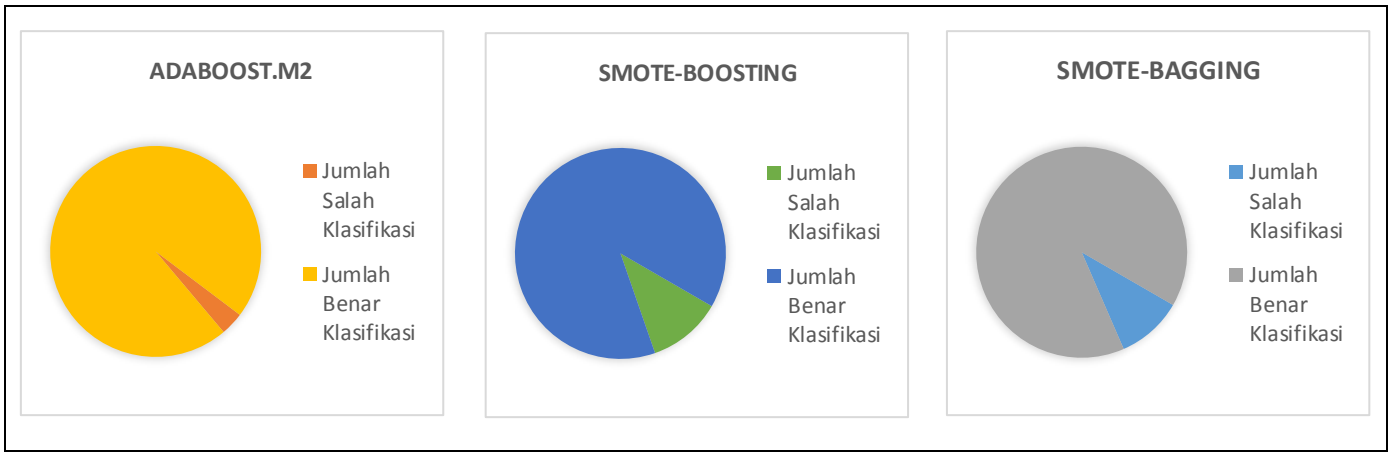
Prediksi	Aktual	Y-hat	Keterangan
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
⋮	⋮	⋮	⋮
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
0	1	0	Salah klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi
1	1	1	Benar Klasifikasi

Jumlah salah klasifikasi
1122
Jumlah benar klasifikasi
9404
Persentase benar klasifikasi
$\frac{9404}{1122+9404} * 100 = 89{,}341\%$

LAMPIRAN E1. Kondisi Klasifikasi Data Bidikmisi dengan Tiga Metode

Metode	Iterasi	Jumlah Salah Klasifikasi	Jumlah Benar Klasifikasi	Persentase Benar Klasifikasi
AdaBoost.M2	5	356	10170	96,618
	10	379	10147	96,399
	15	375	10151	96,437
	25	376	10150	96,427
	30	383	10143	96,361
	50	387	10139	96,323
Rata-rata		376	10150	
SMOTE-Boosting	5	984	9542	90,652
	10	1083	9443	89,711
	15	1280	9247	87,839
	25	1281	9246	87,830
	30	1259	9268	88,039
	50	1279	9247	87,840
Rata-rata		1194	9332	
SMOTE-Bagging	5	880	9464	91,639
	10	1192	9334	88,676
	15	1052	9474	90,006
	25	1020	9506	90,309
	30	1123	9403	89,331
	50	1122	9404	89,341
Rata-rata		1065	9431	

LAMPIRAN E2. Kondisi Rata-Rata Klasifikasi Data Bidikmisi dengan Tiga Metode



BIODATA PENULIS

Penulis memiliki nama lengkap Sinta Septi Pangastuti lahir di Magetan, 22 September 1993 yang merupakan anak pertama dari 1 bersaudara. Pada tahun 2011 lulus dari SMAN 2 Magetan dan diterima melalui jalur reguler D3 Jurusan Statistika FMIPA ITS, kemudian melanjutkan Lintas Jalur (LJ) S1 dengan jurusan yang sama selama 2 tahun. Tanpa jeda, pada bulan Juli 2016 penulis mendapatkan kesempatan melanjutkan ke jenjang S2 di jurusan dan kampus yang sama dan terdaftar sebagai mahasiswa ITS dengan NRP 06211650010038. Penulis lulus S2 dengan Tesis yang berjudul "***Perbandingan Metode Ensemble Random Forest* dengan *SMOTE-Boosting* dan SMOTE-Bagging pada *Klasifikasi Data Mining untuk Kelas Imbalance (Studi Kasus : Data Beasiswa Bidikmisi Tahun 2017 di Jawa Timur)***". Selama menempuh bangku perkuliahan, penulis memberikan les privat kepada anak SD untuk semua mata pelajaran dan bergabung dengan Himpunan Mahasiswa Pascasarjana. Semoga penulis dapat mengamalkan ilmu yang telah didapat sekaligus menjadi amal ibadah yang akan dicatat oleh Allah SWT. Aamiin. Akhir kata apabila pembaca memiliki saran, kritik, dan masukan mengenai tugas akhir dapat menghubungi melalui email ke sintaseptip@gmail.com.

" *It Always Seems Imposible Until It's Done* "

-Nelson Mandela-