TUGAS AKHIR – SS141501

KLASIFIKASI SENYAWA OBAT KANKER UNTUK OPTIMASI TOKSISITAS MENGGUNAKAN *LOGISTIC REGRESSION ENSEMBLES* (LORENS) DAN *ENSEMBLE OF SUPPORT VECTOR MACHINE* DENGAN *FEATURE SELECTION* PADA *HIGH DIMENSIONAL DATA*

Erlin Sukmaputri
NRP 062114 4000 0092

Dosen Pembimbing
Dr.rer.pol. Heri Kuswanto, S.Si., M.Si

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

(Halaman ini sengaja dikosongkan)

TUGAS AKHIR – SS141501

KLASIFIKASI SENYAWA OBAT KANKER UNTUK OPTIMASI TOKSISITAS MENGGUNAKAN *LOGISTIC REGRESSION ENSEMBLES* (LORENS) DAN *ENSEMBLE OF SUPPORT VECTOR MACHINE* DENGAN *FEATURE SELECTION* PADA *HIGH DIMENSIONAL DATA*

Erlin Sukmaputri
NRP 062114 4000 0092

Dosen Pembimbing
Dr.rer.pol. Heri Kuswanto, S.Si., M.Si

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

(Halaman ini sengaja dikosongkan)

FINAL PROJECT – SS141501

CLASSIFICATION OF DRUG CANCER COMPOUND FOR OPTIMIZING TOXICITY USING LOGISTIC REGRESSION ENSEMBLES (LORENS) AND ENSEMBLE OF SUPPORT VECTOR MACHINE WITH FEATURE SELECTION FOR HIGH DIMENSIONAL DATA

Erlin Sukmaputri
SN 062114 4000 0092

Supervisor
Dr.rer.pol. Heri Kuswanto, S.Si., M.Si

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

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

KLASIFIKASI SENYAWA OBAT KANKER UNTUK OPTIMASI TOKSISITAS MENGGUNAKAN *LOGISTIC REGRESSION ENSEMBLES* (LORENS) DAN *ENSEMBLE OF SUPPORT VECTOR MACHINE* DENGAN *FEATURE SELECTION* PADA *HIGH DIMENSIONAL DATA*

TUGAS AKHIR
Diajukan untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Sains
pada
Program Studi Sarjana Departemen Statistika
Fakultas Matematika, Komputasi, dan Sains Data
Institut Teknologi Sepuluh Nopember

Oleh :
Erlin Sukmaputri
NRP. 062114 4000 0092

Disetujui oleh Pembimbing:
Dr.rer.pol. Heri Kuswanto, S.Si., M.Si
NIP. 19820326 200312 1 004

Mengetahui,
Kepala Departemen

Dr. Suhartono
NIP. 19710929 199512 1 001

SURABAYA, JULI 2018

(Halaman ini sengaja dikosongkan)

KLASIFIKASI SENYAWA OBAT KANKER UNTUK OPTIMASI TOKSISITAS MENGGUNAKAN *LOGISTIC REGRESSION ENSEMBLES* (LORENS) DAN *ENSEMBLE OF SUPPORT VECTOR MACHINE* DENGAN *FEATURE SELECTION* PADA *HIGH DIMENSIONAL DATA*

Nama Mahasiswa : Erlin Sukmaputri
NRP : 062114 4000 0092
Departemen : Statistika-FMKSD-ITS
Dosen Pembimbing : Dr.rer.pol. Heri Kuswanto, S.Si., M.Si

Abstrak

Kanker merupakan penyakit akibat pertumbuhan tidak normal akibat sel-sel jaringan tubuh yang berubah menjadi sel kanker. Salah satu pengobatan pada kanker adalah radioterapi, namun memiliki efek samping yaitu membunuh sel-sel normal disekitar sel kanker. Maka dibuat radioprotector dalam mengurangi kematian sel normal dan meningkatkan kematian sel kanker. Data yang digunakan adalah tingkat toksisitas untuk mengklasifikasi senyawa untuk radioprotector dengan 84 senyawa dengan 217 prediktor sehingga data tergolong high dimensional data. Dalam menangani high dimensional data, ensemble dan feature selection digunakan dalam penelitian ini. Penelitian ini menggunakan metode berbasis ensemble yaitu LORENS dan Ensemble of Support Vector Machine (AdaBoost-SVM). Pada LORENS tidak dilakukan feature selection namun, pada AdaBoost-SVM dilakukan feature selection dengan menggunakan Mean Decrease Gini. Penelitian ini membandingkan dari performa klasifikasi dari metode LORENS dan AdaBoost-SVM dengan evaluasi 10-fold cross validation. Performa klasifikasi yang digunakan adalah nilai akurasi karena data yang digunakan sudah balanced. Hasil yang diperoleh adalah metode AdaBoost-SVM mengungguli LORENS dalam hal akurasi. Akurasi optimum yang dihasilkan sebesar 0.7889 dengan jumlah 11 variabel prediktor.

Kata Kunci : AdaBoost, High Dimensional Data, Kanker, LORENS, Support Vector Machine (SVM).

(Halaman ini sengaja dikosongkan)

CLASSIFICATION OF DRUG CANCER COMPOUND FOR OPTIMIZING TOXICITY USING LOGISTIC REGRESSION ENSEMBLES (LORENS) AND ENSEMBLE OF SUPPORT VECTOR MACHINE WITH FEATURE SELECTION FOR HIGH DIMENSIONAL

Name : Erlin Sukmaputri
SN : 062114 4000 0092
Department : Statistics-FMKSD-ITS
Supervisor : Dr.rer.pol. Heri Kuswanto, S.Si., M.Si

Abstract

Cancer is a disease due to abnormal growth of the body tissue cells that turn into cancer cells. One of the treatment for cancer is radiotherapy, but it has side effects of killing normal cells around cancer cells. Thus, radioprotector is made to reducing the death of the normal cells and increase cancer cell death. The data used is the toxicity level to classify the compound for radioprotector with 84 compounds with 217 predictors and the data is classified as high dimensional data. There are several ways for overcome high dimensional data, ensemble and feature selection are used in this study. This research used ensemble based method, which are LORENS and Ensemble of Support Vector Machine (AdaBoost-SVM). In LORENS there is no feature selection but, in AdaBoost-SVM feature selection will be performed using Mean Decrease Gini. This study will compare the classification performance of the LORENS and the AdaBoost-SVM methods with the 10-fold cross validation evaluation. Performance classification used is the value of accuracy because the data is balanced. The result obtained that AdaBoost-SVM method outperformed LORENS in terms of accuracy. The optimum accuracy was 0.7889 with 11 predictor variables.

Keywords : AdaBoost, Cancer, High Dimensional Data, LORENS, Support Vector Machine (SVM).

(Halaman ini sengaja dikosongkan)

KATA PENGANTAR

Puji syukur kepada kehadirat Allah SWT yang telah memberikan rahmat, hidayah, karunia serta pertolongan-Nya yang tak pernah henti diberikan, sehingga penulis dapat menyelesaikan laporan Tugas Akhir dengan judul **"Klasifikasi Senyawa Obat Kanker untuk Optimasi Toksisitas Menggunakan *Logistic Regression Ensemble*s (LORENS) dan *Ensemble* of *Support Vector Machine* dengan *Feature Selection* pada *High Dimensional Data*"** dengan baik, lancar, dan tepat waktu.

Penyusunan laporan Tugas Akhir ini dapat diselesaikan bukan tanpa bantuan serta dukungan dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan rasa terimakasih yang sebesar-besarnya kepada:

1. Ayah saya Harjito, ibu saya Endang Mudjiastutik, adik saya Farid Denhar Baros, yang telah memberikan dukungan, doa, dan nasehat untuk penulis sehingga menajadi motivasi dan penyemangat penulis dalam menghadapi kesulitan dan menyelesaikan laporan Tugas Akhir ini.
2. Dr. Suhartono selaku dosen wali dan Ketua Departemen Statistika, yang telah banyak memberikan saran dan arahan dalam proses belajar di Departemen Statistika.
3. Dr. Sutikno, M.Si. selaku Ketua Program Studi Sarjana yang telah memberikan fasilitas, sarana, dan prasarana dalam proses belajar di Departemen Statistika.
4. Dr. rer. pol. Heri Kuswanto, S.Si., M.Si. selaku dosen pembimbing yang telah meluangkan waktu dan dengan sangat sabar memberikan bimbingan, saran, dukungan serta motivasi selama penyusunan Tugas Akhir.
5. Ibu Santi Puteri Rahayu, M.Si., Ph.D. dan Pak M. Sjahid Akbar, S.Si., M.Si. selaku dosen penguji yang telah memberikan kritikan serta saran demi kesempurnaan tugas akhir ini.
6. Sahabat-sahabat penulis, Drajad Muhammadi Latief, Tanti, Zahrina, Intan, Ajeng, Nikita, Izzan, Icha, Sandra, Desi dan semua teman-teman RESPECT 2014 yang selama ini telah membantu, mendukung, dan mendengarkan keluh kesah

penulis selama masa perkuliahan berlangsung dan juga memberikan saran, dan motivasi dalam proses pengerjaan Tugas Akhir ini.

7. Teman-teman seperjuangan Tugas Akhir, khususnya Dedi, Taufik, Rizky, dan Kiki yang selama ini telah berjuang bersama, saling memberikan semangat, dan membantu dikala ada kesulitan.
8. Teman-teman basket Rana, Ines, Muthia, Muti, Anggi, Uni, Nadia, Putri, dan Ani telah membuat masa kuliah penulis lebih berwarna dan bermakna.
9. Sakinah, Esqy, Tiara, Ikhsan, Adam, Singgih, Andi, Tovan, Dwiki sahabat-sahabat penulis sejak SD yang bersama-sama sedang meraih cita-cita, seling mendukung, dan selalu ada hingga penulis dapat menyelsaikan masa studi.
10. Mas Risky dan Mas Aldi yang telah membantu saat ada kesulitan dalam mengerjakan Tugas Akhir.
11. Semua pihak yang telah membantu dalam penulisan laporan Tugas Akhir ini, yang tidak dapat penulis sebutkan satu per satu.

Penulis menyadari bahwa Tugas Akhir ini masih jauh dari kata sempurna, sehingga saran dan kritik dari segala pihak yang bersifat membangun sangat penulis harapkan demi kesempurnaan penulis selanjutnya.

Surabaya, Juli 2018

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

(Halaman ini sengaja dikosongkan)

DAFTAR LAMPIRAN

Halaman

(*Halaman ini sengaja dikosongkan)*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Penyakit kanker merupakan salah satu penyakit kronis yang memiliki peningkatan cukup tinggi saat ini. Menurut *World Health Organization* atau WHO (2014) kanker merupakan istilah untuk menggambarkan penyakit pada manusia berupa munculnya sel-sel abnormal dalam tubuh yang melampaui batas. Penyakit ini ditandai dengan pertumbuhan sel yang tidak terkendali serta kemampuan sel-sel tersebut untuk menyerang jaringan biologis lainnya, baik dengan pertumbuhan langsung pada jaringan yang bersebelahan (invasi) atau dengan migrasi sel ke tempat yang jauh (metastasis) di dalam tubuh (Meiyanto, Supardjan, & Agustina, 2006).

Berdasarkan Data *Global Burden of Cancer*, *International Agency for Research on Cancer* (IARC), diketahui bahwa pada tahun 2012 kasus kanker mengakibatkan 8.2 juta kematian (58.3%) dari total 14.1 juta kasus baru kanker di seluruh dunia. Data dari Sistem Informasi Rumah Sakit (SIRS) pada tahun 2004-2007 mencatat urutan jenis kanker dengan penderita terbanyak di Indonesia adalah kanker payudara, kanker serviks, kanker hati, leukemia, dan kanker paru-paru (Rusyidi, 2009). Selain itu, *Serikat Pengendalian Kanker Internasional* (UICC) mempredikasi akan terjadi peningkatan jumlah penderita kanker sebesar 300% di seluruh dunia pada tahun 2030. Jumlah tersebut 70% berada di negara berkembang seperti Indonesia (Kartika, 2013).

Kanker terjadi karena kerusakan struktur genetik yang menyebabkan pertumbuhan sel menjadi tidak terkontrol. Pola insiden kanker bervariasi sesuai jenis kelamin, ras, dan letak geografik. Beberapa kanker dapat dipengaruhi oleh faktor genetik keluarga, namun yang paling sering terjadi karena faktor lingkungan dan gaya hidup (Wahyuni, 2015). Terdapat beberapa cara yang dilakukan untuk mengobati penyakit kanker yaitu operasi, radioterapi, dan kemoterapi. Pengobatan ini dilakukan untuk membunuh sel-sel kanker, namun tidak sedikit usaha tersebut justru menimbulkan efek samping (Sukardiman, R. Abdul, & P.N. Fatma, 2004). Salah satu pengobatan kanker yang umum

adalah radioterapi. Radioterapi adalah suatu tindakan pengobatan terapi radiasi pada penyakit kanker dengan menggunakan radiasi, seperti sinar *Gamma*, sinar-x ataupun elektron berenergi tinggi (Syam, Dewang, & Abdullah, 2014). Cara kerja radioterapi adalah dengan memberikan dosis radiasi yang mematikan tumor pada daerah yang telah ditentukan (volume target) sedangkan jaringan normal sekitarnya mendapat dosis seminimal mungkin (Pratiwi, 2016).

Pengobatan menggunakan radioterapi memiliki efek samping yang signifikan yaitu membunuh sel-sel normal disekitar sel kanker. Hal ini karena radioterapi tersebut merusak DNA yang ada pada sel kanker dan membuat DNA memacu p53 untuk apoptosis (kematian sel). p53 merupakan gen supresor tumor yang bertindak menghentikan perkembangan tumor. Saat sel terkena radioterapi, DNA pada sel kanker sudah rusak dan mengakibatkan apoptosis terjadi pada sel yang normal dan membunuh sel-sel tersebut. Dalam menanggulangi efek radioterapi tersebut, Ariyasu, *et al.* (2014) mendesain *radioprotector* atau proteksi radiasi dengan mencari komponen senyawa yang berhubungan dengan protein p53. Didapatkan 84 senyawa yang diduga baik untuk *radioprotector* yang kemudian dari masing-masing senyawa diberi dua percobaan yaitu pada sel normal dan sel yang terkena radiasi *Gamma*. Pada percobaan pertama dilakukan pemberian senyawa pada sel normal, percobaan ini bertujuan untuk mengukur toksisitas. Percobaan kedua dilakukan pemberian senyawa pada sel yang telah terkena radiasi sinar *Gamma* (10 Gy), percobaan ini bertujuan mengukur fungsi proteksi radiasi. Indikator yang digunakan pada kedua percobaan adalah tingkat kematian sel. Senyawa kandiat yang diperlukan untuk *radioprotector* adalah yang memiliki tingkat kematian sel rendah pada toksisitas dan tingkat kematian sel tinggi pada fungsi proteksi radiasi. Penelitian ini akan dilakukan analisis pada tingkat toksisitas dimana toksisitas adalah kemampuan suatu molekul atau senyawa kimia yang dapat menimbulkan kerusakan pada bagian tertentu dalam mahluk hidup (Durham, 1975). Penelitian oleh Matsumoto, *et al.* (2016) yang melanjutkan penelitian Ariyasu, *et al.* (2014) yaitu dengan memprediksi proteksi radiasi dan toksisitas menggunakan *Random*

Forest (RF) dan *Support Vector Machine* (SVM) dan didapatkan hasil bahwa *Random Forest* lebih baik digunakan dalam memprediksi toksisitas dengan nilai AUC 77.8% sedangkan SVM baik digunakan dalam memprediksi proteksi radiasi dengan nilai AUC 64.4%. Penelitian lain dilakukan oleh Kimura, *et al*. (2017) dengan membandingkan hasil ketepatan klasifikasi senyawa untuk optimasi proteksi radiasi dan toksisitas dengan menggunakan *random forest* (RF), *support vector machine* (SVM), *extreme gradient boosting* (XGB), dan K-*nearest neighbor* (kNN).

Data yang digunakan dalam penelitian Matsumoto, *et al.* (2016) dan Kimura, *et al*. (2017) merupakan *high dimensional data* dimana jumlah variabel yang diamati sangat besar dan jumlah pengamatan yang dilakukan jauh lebih kecil dari jumlah variabel. Keduanya melakukan *feature selection* dalam menangani *high dimensional data* dengan menggunakan *feature importance* yaitu pada 5%, 10%, 15%, 20%, 25%, 30%, 35%, dan 100%. Menurut Pappu & Pardalos (2014) tantangan pada kasus *high dimensional data* adalah akurasi yang buruk akibat fenomena *curse of dimensionality* dan model yang *overfitting* pada data *training*, sehingga kemampuan model dalam generalisasi menjadi buruk. Ada dua pendekatan dalam mengatasi tantangan *high dimensional data* yaitu mengurangi dimensi pada dataset atau dengan mengaplikasikan metode yang independen terhadap dimensional data. Cara yang umum adalah dengan melakukan *feature selection* terhadap variabel atau menggunakan klasifikasi berbasis *ensemble*. Setelah penelitian sebelumnya berfokus pada *feature selection*, pada penelitian ini menggunakan *ensemble* sebagai basis dalam kasifikasi. Salah satu metode klasifikasi pada *machine learning* yang berbasis *ensemble* adalah *Logistic Regression Ensembles* (LORENS) dengan keunggulan mengatasi kasus *high dimensional data.* Selain itu, menurut Lim, *et al.* (2010) LORENS juga mampu menangani variabel respon yang tidak seimbang dan mampu meningkatkan akurasi, sesitifitas, dan spesifisitas dibanding metode klasifikasi lainnya. LORENS telah banyak diaplikasikan pada beberapa penelitian sebelumnya yaitu oleh Kuswanto, Asfihani, Sarumaha, & Ohwada (2015), dimana LORENS digunakan dalam mengklasifikasikan kasus pembelotan konsumen

dengan ukuran sampel yang sangat besar. Selanjutnya, Kuswanto & Wedhana (2018) melakukan klasifikasi pada ekspresi gen pada penyakit Alzheimer dengan membandingkannya dengan metode *Naïve Bayes* dan didapatkan hasil bahwa LORENS memiliki nilai AUC yang lebih tinggi yaitu 75.9%.

Selain LORENS, metode *ensemble* lainnya adalah *Ensemble of Support Vector Machine (EnSVM*) dengan menggunakan SVM sebagai *base classifier*. Metode SVM menemukan atau mencari fungsi pemisah atau *hyperplane* terbaik yang memisahkan dua kelas pada ruang input. EnSVM adalah kumpulan *N* model SVM dimana masing-masing telah dilatih (*training*) secara individu terhadap *N* dataset. Selama beberapa tahun terakhir, SVM telah banyak diaplikasikan pada masalah *high dimensional data* yaitu klasifikasi *remote sensing*, web dokumen dan analisis microarray (Pappu & Pardalos, 2014). Metode *Ensemble* pada SVM yang digunakan adalah *Adaptive Boosting*. Adaboost menggunakan SVM sebagai *base classifier* untuk klasifikasi, dan banyak penelitian sebelumnya yang menggunakan *ensemble* SVM yaitu penelitian oleh Pal & Mather (2005) yang membahas klasifikasi *remote sensing* dengan *high dimensional data* menggunakan SVM, ANN, dan Maximum Likelihood, didapatkan hasil bahwa akurasi tertinggi pada metode SVM. Penelitian oleh Dragomir & Bezerianos (2006) membahas tentang prediksi ekspresi gen beberapa jenis kanker dan EnSVM terbukti memiliki akurasi yang optimal dibandingkan dengan SVM tunggal dan kNN. Selain itu, pada penelitian Huang, *et al,* (2017) membahas tentang prediksi kanker payudara pada data skala kecil dan besar dimana didapatkan hasil bahwa untuk dataset skala kecil lebih baik menggunakan EnSVM basis *bagging* dengan kernel linier dan EnSVM basis *boosting* dengan kernel RBF. Penelitian lain terkait kasus *high dimensional data* adalah penelitian oleh Piao *et al* (2015) yaitu klasifikasi pada penyakit kanker menggunakan SVM dengan *ensemble* Ada*Boost*, *bagging*, *random forest,* dan *multiple independent feature subset* didapatkan hasil bahwa meskipun data memiliki dimensi yang tinggi, namun akurasi pada SVM dengan *ensemble boosting* sangat baik yaitu akurasi berkisar antara 91,18% – 97.22%.

Berdasarkan uraian diatas, penelitian ini menggunakan metode LORENS dan *Ensemble of* SVM dalam memprediksi tingkat toksisitas untuk obat kanker. Dalam menemukan metode yang tepat dalam menangani *high dimensional data* dan mampu meningkatkan akurasi dibandingkan dengan metode tunggal, dipilih metode berbasis *ensemble* yang sudah teruraikan diatas. Pemilihan *feature* diguankan pada metode *Ensemble of* SVM, sedangkan metode LORENS tidak dilakukan pemilihan *feature.* Senyawa untuk optimasi toksisitas akan diklasifikasikan secara biner berdasarkan tingkat kematian sel, yaitu sebesar kurang dari 20% dan 20-100%. Evaluasi ketepatan klasifikasi akan membandingkan nilai akurasi total, spesifisitas, dan sensitifitas tertinggi dan dipilih metode terbaik. Senyawa-senyawa dalam klasifikasi yang memiliki toksisitas rendah bisa direkomendasikan untuk *radioprotector*

1.2 Rumusan Masalah

Metode klasifikasi untuk prediksi tingkat toksisitas untuk *radioprotector* dalam menangani penyakit kanker telah menjadi perhatian pakar dibidang penemuan obat. Hal ini karena dalam menemukan senyawa target, ada proses mengidentifikasi gen yang menyebabkan gejala dan menemukan senyawa yang mengikat untuk menghambat fungsi gen. Dimana proses tersebut membutuhkan peneliti profesional dan juga waktu yang lama. Untuk mengatasi masalah tersebut, dalam beberapa tahun terakhir, *machine learning* telah diterapkan di bidang penemuan obat. Kasus pada penelitian ini adalah data yang dimiliki berupa *high dimensional data* sehingga dalam menangani kasus ini digunakan *machine learning* dengan basis *ensemble* dan *feature selection.* Pemilihan fitur digunakan pada metode *Ensemble of* SVM, sedangkan pada LORENS tidak dilakukan pemilihan fitur. Oleh karena itu, permasalahan yang dibahas dalam penelitian ini adalah bagaimana pengklasifikasian senyawa berdasarkan tingkat toksisitas untuk obat kanker dengan menggunakan metode *Logistic Regression Ensembles* (LORENS) dan *Ensemble of Support Vector Machine* serta perbandingan akurasi dari kedua metode tersebut.

1.3 Tujuan

Berdasarkan rumusan masalah yang ada, adapun tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut.

1. Mendapatkan hasil dan ketepatan klasifikasi senyawa obat kanker untuk optimasi toksisitas menggunakan metode *Logistic Regression Ensembles* (LORENS).
2. Mendapatkan hasil dan ketepatan klasifikasi senyawa obat kanker untuk optimasi toksisitas menggunakan metode *Ensemble of Support Vector Machine*.
3. Memilih metode klasifikasi terbaik dari hasil analisis mengunakan metode *Logistic Regression Ensembles* (LORENS) dan *Ensemble of Support Vector Machine.*

1.4 Batasan Masalah

Batasan masalah pada penelitian ini adalah adalah bahwa data yang digunakan tidak dilakukan *pre-processing* data karena sudah dilakukan pada penelitian sebelumnya dan analisis yang digunakan adalah data tingkat toksisitas pada senyawa obat kanker (*radioprotector*).

1.5 Manfaat

Berdasarkan tujuan yang ingin dicapai, adapun manfaat yang diharapkan dari hasil penelitian ini adalah sebagai berikut.

1. Bagi bidang kedokteran, diharapkan penelitian ini dapat memberikan kontribusi dalam prediksi klasifikasi senyawa obat kanker berdasarkan tingkat toksisitas untuk mengurangi kematian sel normal disekitar sel kanker dimana diperoleh senyawa toksisitas rendah dalam penyusunan obat kanker (*radioprotector*).
2. Bagi dunia pendidikan, penelitian ini dapat menjadi referensi dalam melakukan penelitian selanjutnya mengenai klasifikasi senyawa untuk obat kanker metode *Logistic Regression Ensembles* (LORENS) dan *Ensemble of Support Vector Machine.*

BAB II
TINJAUAN PUSTAKA

Pada Bab ini menjelaskan secara rinci teori yang digunakan untuk menganalisis prediksi klasifikasi. Metode klasifikasi yang digunakan adalah LORENS dan *Ensemble of* SVM dengan evaluasi performasi menggunakan *Cross Validation.*

2.1 *Logistic Regression Ensembles* (LORENS)

Pada tahun 2010 Lim, Ahn, Moon dan Chen mengembangkan LORENS dengan menggunakan regresi logistik sebagai *base classifier* dan berdasarkan algoritma LR CERP (*Logistic Regression Classification By Ensembles From Random Partition*). LORENS mengombinasikan hasil model regresi logistik untuk mendapatkan satu *classifier* yang kuat dibanding metode agregasi kompleks lainnya. Hal ini mengakibatkan akurasi dari prediksi yang dilakukan LORENS meningkat. LORENS menggunakan regresi logistic sebagai *base classifer.*

Regresi logistik adalah metode klasifikasi dasar yang digunakan mencari hubungan variabel respon (y) yang bersifat dikotomus maupun polikotomus dengan variabel prediktor (x) yang bersifat polikotomus (Hosmer dan Lemeshow, 2000). Variabel respon (y) dari regresi logistik biner terdiri dari 2 kategori yaitu "sukses" dan "gagal", dimana notasi dari $y = 1$ untuk kategori "sukses" dan $y = 0$ untuk kategori "gagal". Sehingga variabel respon y mengikuti distribusi Bernoulli untuk setiap observasi tunggalnya. Fungsi probabilitas untuk setiap observasinya adalah sebagai berikut:

$$f(y) = \pi^{y}(1-\pi)^{1-y}\,; y = 0,1 \qquad (2.1)$$

dimana apabila $y = 0$ maka $f(y) = 1 - \pi$ dan $y = 1$ maka $f(y) = \pi$, sehinga didapatkan fungsi regresi logistik sebagai berikut:

$$f(z) = \frac{e^{z}}{1+e^{z}};\ z = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p \qquad (2.2)$$

dimana p adalah banyak variabel prediktor. Nilai $f(z)$ terletak antara 0 dan 1 untuk setiap nilai z yang diberikan, karena nilai z sendiri terletak antara $-\infty$ dan ∞. Model regresi logistik tersebut

sebenarnya menggambarkan sebuah probabilitas dari suatu objek. Model regresi logistiknya adalah sebagai berikut:

$$\pi(\boldsymbol{x}) = \frac{e^{\beta_0+\beta_1 x_1+\cdots+\beta_p x_p}}{1+e^{\beta_0+\beta_1 x_1+\cdots+\beta_p x_p}} \tag{2.3}$$

dimana,

$\beta_0 =$ konstanta,

$\beta_j =$ koefisien regresi

$j =$ banyaknya variabel prediktor.

Terdapat suatu bentuk alternatif dari persamaan regresi logistik seperti persamaan (2.4) yang merupakan tranformasi logit dari $\pi(\mathbf{x}_i)$ (Yan & Su, 2009).

$$Logit[\pi(\boldsymbol{x})] = \boldsymbol{x}^T\boldsymbol{\beta} = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p \tag{2.4}$$

Estimasi parameter dalam regresi logistik dilakukan dengan metode *Maximum Likelihood* (MLE). Metode MLE memberikan nilai estimasi β dengan memaksimumkan fungsi likelihood (Hosmer dan Lemeshow, 2000). Metode tersebut mengestimasi parameter β dengan cara memaksimumkan fungsi likelihood dan mensyaratkan bahwa data harus mengikuti suatu distribusi tertentu. Pada regresi logistik, setiap pengamatan mengikuti distribusi bernoulli sehingga dapat ditentukan fungsi likelihoodnya sebagai berikut.

$$L(\boldsymbol{X},\boldsymbol{\beta}) = \prod_{i=1}^{n}[\pi(\boldsymbol{x})]^y\,[1-\pi(\boldsymbol{x})]^{1-y} \tag{2.5}$$

MLE didapatkan dengan cara memaksimumkan logaritma fungsi likelihood pada persamaan (2.5) dengan hasil sebagai berikut.

$$\ln L(\boldsymbol{X},\boldsymbol{\beta}) = \sum_{i=1}^{n}\left[y(\boldsymbol{x}^T\boldsymbol{\beta}) - \ln\left(1+exp(\boldsymbol{x}^T\boldsymbol{\beta})\right)^{-1}\right] \tag{2.6}$$

Sehingga didapatkan hasil estimasi parameter $\widehat{\boldsymbol{\beta}}$ sebagai berikut.

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\,\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{z} \tag{2.7}$$

dimana $\boldsymbol{z}$ merupakan vector berukuran n x 1 dan $\boldsymbol{W}$ merupakan vector pembobot, dengan $\boldsymbol{z}$ adalah sebagai berikut.

$$\boldsymbol{z} = Logit[\hat{\pi}(\boldsymbol{x})] + \frac{y-\hat{\pi}(\boldsymbol{x})}{\hat{\pi}(\boldsymbol{x})[1-\hat{\pi}(\boldsymbol{x})]} \tag{2.8}$$

Matriks varian kovarian untuk $\widehat{\boldsymbol{\beta}}$ ditampilkan pada persamaan berikut.

$$Var(\widehat{\boldsymbol{\beta}}) = (\boldsymbol{X}^T diag[\hat{\pi}(1-\hat{\pi})]\boldsymbol{X})^{-1} \quad (2.9)$$

Berdasarkan *base classifier* regresi logistik diatas, performa CERP sangat tergantung oleh banyaknya variabel prediktor yang digunakan dalam satu partisi. Partisi yang optimal dapat diperoleh dari persamaan berikut ini:

$$K = \frac{6 \times p}{n} \quad (2.10)$$

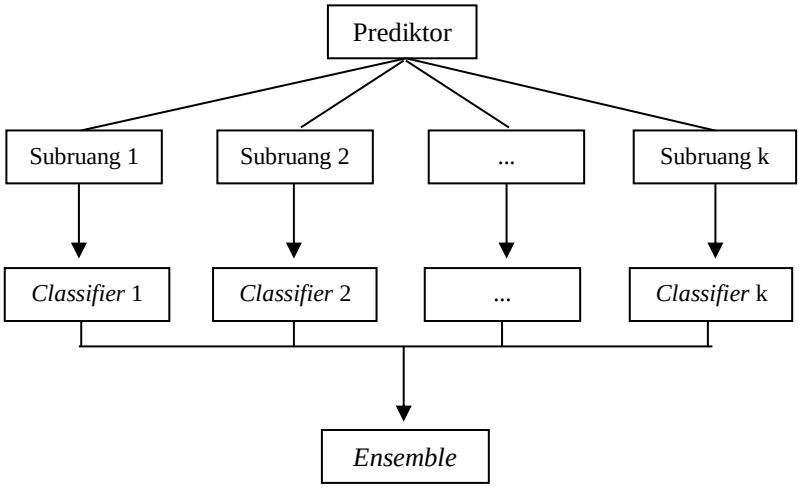
dimana p merupakan banyak variabel prediktor dan n adalah banyaknya observasi. Jika ukuran n lebih besar dari ukuran p, maka partisi yang optimal didapatkan dengan membagi data sebanyak i menjadi $\frac{p}{i}$ dimana i merupakan bilangan integer yang kurang dari n.

LORENS menggunakan prosedur yang sama dengan LR CERP namun, prosedur LR CERP diulang beberapa kali sampai terbentuk beberapa *ensemble.* LORENS mempartisi ruang prediktor Θ yang dipartisi menjadi K subruang $(\theta_1, \theta_2, ..., \theta_k)$ yang sama. Subruang dipilih secara acak pada distribusi yang sama sehingga diasumsikan tidak terdapat bias pada saat pengambilan prediktor pada masing-masing subruang. Masing-masing subruang membentuk model regresi tanpa melalui seleksi variabel. Pengacakan pada variabel ini diharapkan mempunyai probabilitas yang sama pada masing-masing *classifier* pada satu *ensemble* dan juga *error* klasifikasi yang hampir sama, sehingga model yang terbentuk menghasilkan prediksi yang akurat dan tidak bias.

LORENS mengkombinasikan nilai prediksi dari model-model regresi yang terbentuk pada masing-masing partisi dalam satu *ensemble,* sehingga akurasi yang didapatkan meningkat. Dengan mengulangi prosedur LR CERP, LORENS mendapatkan kombinasi rata-rata ataupun nilai terbanyak yang menghasilkan akurasi yang hampir sama. Rata-rata menghasilkan nilai sedikit lebih unggul daripada nilai terbanyak, sehingga LORENS lebih baik menggunakan nilai rata-rata. Dengan menggunakan prosedur LR CERP, LORENS menghasilkan beberapa *ensemble* dengan

patisi acak yang berbeda-beda pula. Dari beberapa *ensemble* yang terbentuk, diambil nilai terbanyak diantaranya. Berdasarkan nilai tersebut didapatkan satu akurasi umum. Nilai akurasi tersebut telah ditingkatkan dengan kontribusi dari beberapa *ensemble* yang dibangun.

Kelebihan LORENS adalah dalam penentuan *threshold*. Pada umumnya *Threshold* yang digunakan dalam klasifikasi dengan respon biner adalah 0,5. Apabila proporsi kelas 0 dan 1 tidak seimbang, akurasi klasifikasi tidak akan baik. *Threshold* yang optimal dibutuhkan untuk menyeimbangkan *sensitifity* dan *spesificity*. Gambar (2.1) merupakan bagan yang menggambarkan konsep satu *ensemble* dari *Logistic Regression Classification By Ensembles From Random Partition*.



Gambar 2.1 Bagan Konsep LR CERP

Berikut merupakan rumus untuk menghitung *threshold* optimal dari LORENS.

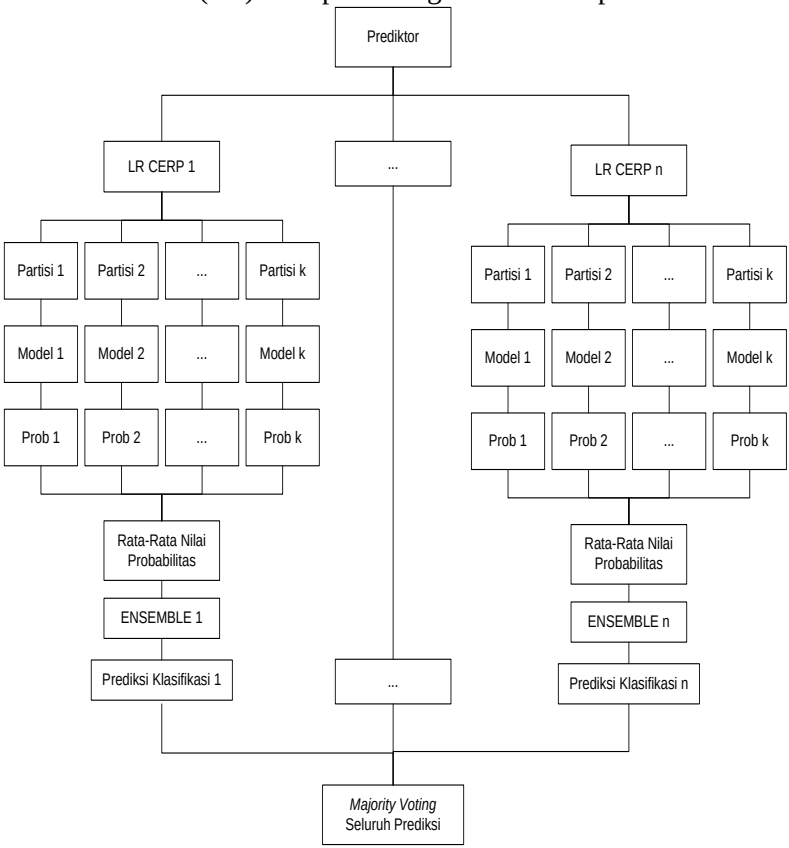
$$Threshold = \frac{p + 0.5}{2} \tag{2.11}$$

p adalah probabilitas pengamatan yang berada di kelas positif. Berikut ini merupakan tahapan dalam proses klasifikasi.

1. Membentuk model logit dari data *training.*

2. Memasukan data *testing* ke dalam model logit, sehingga diperoleh nilai probabilitas.
3. Mengklasifikasikan pengamatan data *testing*. Jika nilai probabilitasnya lebih besar daripada nilai *threshold* maka pengamatan masuk ke dalam kelas positif, sebaliknya jika nilai probabilitasnya lebih kecil daripada nilai *threshold* maka pengamatan masuk ke dalam kelas negatif.
4. Membandingkan kelas aktual dengan prediksi klasifikasi.
5. Menghitung hasil ketepatan klasifikasi.

Berikut Gambar (2.2) merupakan bagan dari konsep LORENS.



Gambar 2.2 Bagan Konsep LORENS dengan prosedur LR CERP

LORENS mempunyai kelebihan bebas dari asumsi dimensi data, karena LORENS melakukan partisi secara acak terhadap prediktornya. Dalam hal komputasi, LORENS lebih unggul daripada LR CERP yang masih menggunakan *tree algorithm* (algoritma pohon). Keakuratan metode dapat menjadi lebih baik dengan dua keunggulan LORENS tersebut diatas (Lee, *et al*., 2013).

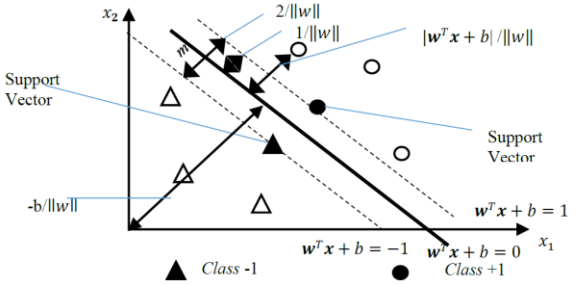
2.2 *Support Vector Machine*

SVM pertama kali diperkenalkan oleh Vapnik tahun 1992 di *Annual Workshop on Computational Learning Theory*. SVM adalah salah satu dari beberapa metode yang dikembangkan untuk mengatasi permasalahan yang tidak bisa diselesaikan dengan metode statistika klasik, terutama pada kasus klasifikasi dan prediksi. SVM dikembangkan dengan prinsip *linier classifier*. Namun dalam kasus nyata sering dijumpai data yang tidak linier sehingga dikembangkan SVM untuk kasus nonlinier dengan memasukkan konsep kernel. Dengan begitu, ada jaminan bahwa klasifikasi menggunakan SVM akan menghasilkan pemetaaan yang sangat akurat (Hsu, Chang, & Lin, 2010).

SVM merupakan suatu teknik untuk menemukan fungsi pemisah dalam pengklasifikasian yang dapat memisahkan dua atau lebih kelompok data yang berbeda. Konsep dari SVM adalah berusaha menemukan *hyperplane* yang optimal pada *input space*. Fungsi dari *hyperplane* itu digunakan sebagai pemisah dua buah kelas pada *input space*. Kelas sering disimbolkan dengan -1 dan +1. SVM dapat menemukan fungsi pemisah (*hyperplane*) terbaik diantara fungsi yang tidak terbatas jumlahnya untuk memisahkan obyek. *Hyperplane* terbaik terletak tepat di tengah antara dua set obyek dari dua kelas. Mencari *hyperplane* terbaik ekuivalen dengan memaksimalkan *margin* atau jarak antara dua set obyek dari dua kelas berbeda. SVM bekerja untuk menemukan suatu fungsi pemisah dengan *margin* yang maksimal (Vapnik, 1999). SVM merupakan teknik klasifikasi dengan proses pelatihan (*supervised learning*).

2.2.1 Klasifikasi Linier *Separable Case* SVM

Klasifikasi linier *separable* data adalah penerapan metode SVM pada data yang dapat dipisahkan secara linier. Misal pada dimensi m, $\boldsymbol{x}_i = \{\boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_m\}$ adalah *dataset* dan $y_i = \{+1, -1\}$ adalah label kategori untuk *dataset*..



Sumber: Kusumaningrum (2014)

Gambar 2.3 Konsep *Hyperplane separable case* pada SVM

Ilustrasi linier *separable case* pada SVM ditunjukkan oleh gambar 2.3. Pada gambar 2.3, garis putus-putus merupakan bidang membatas kelas +1 dan kelas -1, sedangkan garis tegas lurus diantara garis putus-putus merupakan bidang pemisah antara kelas +1 dan kelas -1. Garis pemisah tersebut merupakan *hyperplane* terbaik antara kedua kelas dapat ditemukan dengan mengukur *margin hyperplane* tersebut dan mencari titik maksimalnya. *Hyperplane* terbaik dilihat dari nilai *margin* yang dimaksimalkan dan melewati pertengahan antara kedua kelas. *Hyperplane* sampel yang lokasinya paling dekat dengan *hyperplane* disebut *support vector*, dengan proses dalam SVM adalah mencari *support vector* untuk memperoleh *hyperplane* yang terbaik.

Fungsi pemisah pada linier *separable* memisahkan data dari kelas yang berbeda. Misalnya diberikan data pada dimensi m dan data pengamatan sebanyak n, yaitu $\boldsymbol{x}_i$ dengan $i = 1,2,..,n$ maka fungsi pemisahnya adalah :

$$D(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x}_i + b \qquad (2.12)$$

dimana $\boldsymbol{w}$ merupakan vector bobot dengan dimensi m (ukuran mx1) dan b adalah konstanta yang disebut bias. Penentuan kelas pada masing-masing pengamatan berdasarkan persamaan (2.12)

adalah apabila $\boldsymbol{w}^T\boldsymbol{x}_i + b < 0$ maka $\boldsymbol{x}_i$ masuk kelas 1 atau $y_i = +1$ dan $\boldsymbol{w}^T\boldsymbol{x}_i + b > 0$ maka $\boldsymbol{x}_i$ masuk kelas 0 atau $y_i = -1$. Penentuan kelas tersebut juga merupakan syarat yang harus dipenuhi (konstrain) yang dapat dituliskan menjadi:

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 \geq 0 \qquad i = 1,2,\ldots, n \tag{2.13}$$

Jarak antara data $\boldsymbol{x}$ pada tiap kelas dengan *hyprplane* adalah pada persamaan (2.14).

$$d(\boldsymbol{w}, b; \boldsymbol{x}) = \frac{|\boldsymbol{w}^T\boldsymbol{x}_i + b|}{\|\boldsymbol{w}\|} \tag{2.14}$$

Dengan mengalikan b dan $\boldsymbol{w}$ dengan sebuah konstanta, akan dihasilkan nilai *margin* yang dikalikan dengan konstanta yang sama (Gunn, 1998). Jarak terdekat antara data $\boldsymbol{x}$ dengan *hyperplane* pada kelas 1 dan 2 masing-masing adalah $\frac{1}{\|\boldsymbol{w}\|}$, sehingga nilai *margin* antara bidang pembatas (berdasarkan rumus jarak garis ke titik pusat) yaitu pada persamaan (2.14).

$$\frac{2}{\|\boldsymbol{w}\|} \tag{2.15}$$

Hyperplane yang optimal diperoleh dengan memaksimalkan nilai *margin*. Nilai *margin* akan maksimal bila nilai $\|\boldsymbol{w}\|$ minimal, maka pencarian bidang pemisah terbaik dengan nilai *margin* terbesar dapat dirumuskan menjadi masalah optimasi konstrain yaitu sebagai berikut :

$$\min \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{2.16}$$

dengan fungsi kendala pada persamaan (2.13). Secara umum persoalan optimasi (2.16) akan lebih mudah diselesaikan jika diubah ke dalam fungsi *Lagrange*. Dengan demikian permasalahan optimasi konstrain dapat diubah menjadi persamaan (2.17).

$$\min_{w,b} L_P(\boldsymbol{w}, b) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n} \alpha_i[y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1] \tag{2.17}$$

dengan tambahan konstrain, $\alpha_i \geq 0$ (nilai koefisien *Langrange*). Dengan meminimumkan L_P terhadap $\boldsymbol{w}$ dan b dengan cara $\frac{\partial L_p(\boldsymbol{w},b)}{\partial \boldsymbol{w}} = 0$ dan $\frac{\partial L_p(\boldsymbol{w},b)}{\partial b} = 0$ diperoleh persamaan (2.18).

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \text{ dan } \sum_{i=1}^{n} \alpha_i y_i = 0 \quad (2.18)$$

Persamaan (2.18) didapatkan dari kondisi *Karush-Kuhn-Tucker* (KTT) (Gale, *et al.*, 1951). Dengan mensubstitusikan hasil KTT kedalam persamaan *Lagrange* L_P (*primal problem*) pada persamaan (2.17) sehingga didapatkan persamaan *Lagrange* L_D (*dual problem*) sebagai berikut.

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\boldsymbol{x}_i^T \boldsymbol{x}_j\right) \quad (2.19)$$

Sehingga persoalan pencarian *hyperplane* optimal dapat dirumuskan sebagai berikut.

$$\max_{\alpha} L_D(\alpha) = \max_{\alpha} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\boldsymbol{x}_i^T \boldsymbol{x}_j\right) \right), \quad (2.20)$$

dengan $\sum_{i=1}^{n} \alpha_i y_i = 0$, $\alpha_i \geq 0$.

Formula pencarian *hyperplane* optimal ini adalah permasalahan optimasi kuadrat (*quadratic programming*). Hasil dari optimasi kuadrat pada persamaan (2.20) adalah diperoleh nilai α_i yang nantinya digunakan untuk menemukan nilai $\boldsymbol{w}$. Nilai α_i yang didapatkan untuk setiap data *training* dengan nilai $\alpha_i > 0$ merupakan *support vector*, sedangkan sisanya memiliki nilai $\alpha_i = 0$. Dengan demikian fungsi keputusan yang dihasilkan hanya dipengaruhi oleh *support vector.* Langkah selanjutnya setelah nilai α_i ditemukan adalah menentukan kelas dari prediksi data *testing* ($\boldsymbol{x}_{new}$) yang dapat dicari dengan persamaan (2.21).

$$f(\boldsymbol{x}_{new}) = sign\left(\hat{\boldsymbol{w}}^T \boldsymbol{x}_{new} + \hat{b}\right), \quad (2.21)$$

dimana $\hat{\boldsymbol{w}} = \sum_{i=1}^{n} \hat{\alpha_i} y_i \boldsymbol{x}_i$ dan $b = \frac{1}{n_{sv}} \left(\sum_{i=1}^{n_{sv}} \frac{1}{y_i} - (\hat{\boldsymbol{w}}^T \boldsymbol{x}_{new}) \right)$ dengan $\boldsymbol{x}_i$ merupakan *support vector*, n_{sv} merupakan jumlah *support vector*, dan $\boldsymbol{x}_{new}$ adalah data yang diklasifikasikan.

2.2.2 Kasifikasi Linier *Nonseparable Case* SVM

Formulasi yang telah dijelaskan sebelumnya akan diperluas untuk mengatasi adanya misklasifikasi sehingga dapat digunakan data *non-separable*. Masalah optimasi fungsi obyektif maupun

kendala dimodifikasi dengan mengikuti *slack variable* $\xi < 0$ yang merupakan sebuah ukuran kesalahan klasifikasi. Berikut ini merupakan *constraint* yang sudah dimodifikasi untuk kasus *non-separable*.

$$y_i[(\boldsymbol{w}^T \boldsymbol{x}_i) + b] \geq 1 - \xi_i \tag{2.22}$$

untuk $\xi_i \geq 0$ dan $i = 1,2,\dots,n$. *Hyperplane* atau pemisah yang optimal ditentukan dengan vektor $\boldsymbol{w}$, yaitu dengan memaksimalkan margin dengan rumus:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n} \xi_i \tag{2.23}$$

dengan nilai konstrain pada persamaan (2.22). Nilai C merupakan parameter yang menentukan besar biaya (*cost*) akibat kesalahan klasifikasi dari data *training* dan nilainya ditentukan oleh pengguna. Bentuk fungsi *Lagrange* untuk *primal problem* pada kasus *lineary separable* adalah pada persamaan (2.24).

$$\begin{aligned} L_P(\boldsymbol{w}, b, \mu, \xi) = \frac{1}{2}\|\boldsymbol{w}\|^2 - C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \mu_i \xi_i \\ - \sum_{i=1}^{n} \alpha_i \left(y_i[(\boldsymbol{w}^T \boldsymbol{x}_i) + b] - 1 + \xi_i\right) \end{aligned} \tag{2.24}$$

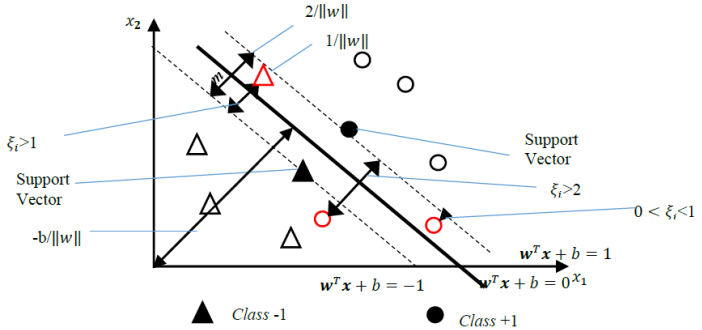
dimana, $\alpha_i \geq 0$ dan $\mu_i \geq 0$ adalah pengali *Lagrange*. Dengan meminimumkan L_P terhadap $\boldsymbol{w}$, b, dan ξ dengan cara $\frac{\partial L_P(\boldsymbol{w},b,\xi)}{\partial \boldsymbol{w}} = 0$, $\frac{\partial L_p(\boldsymbol{w},b,\xi)}{\partial b} = 0$, dan $\frac{\partial L_p(\boldsymbol{w},b,\xi)}{\partial \xi_i} = 0$ diperoleh persamaan (2.25).

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \ , \sum_{i=1}^{n} \alpha_i y_i \ = 0 \text{ , dan } \alpha_i = \mathrm{C} - \mu_i \tag{2.25}$$

Persamaan (2.19) didapatkan dari kondisi KTT dengan syarat $\alpha_i \geq 0$, $\mu_i \geq 0$, $\alpha_i \left(y_i[(\boldsymbol{w}^T \boldsymbol{x}_i) + b] - 1 + \xi_i\right) = 0$, dan $\mu_i \xi_i = 0$. Dengan mensubstitusikan hasil KTT kedalam persamaan *Lagrange* L_P (*primal problem*) pada persamaan (2.25) sehingga didapatkan persamaan *Lagrange* L_D (*dual problem*) sebagai berikut.

$$\max_{\alpha} L_D(\alpha) = \max_{\alpha} \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\boldsymbol{x}_i^T \boldsymbol{x}_j \right) \right), \quad (2.26)$$

dengan konstrain $\sum_{i=1}^{n} \alpha_i y_i = 0$, $\alpha_i \geq 0$. Berikut ini merupakan gambar konsep pada *linier nonseparable case* pada SVM.

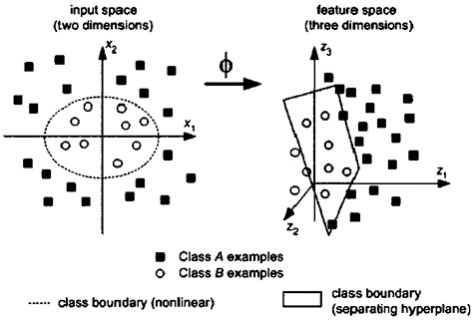


Sumber**:** Kusumaningrum (2014)

Gambar 2.4 *Hyperplane* pada *Linier Nonseparable* SVM

2.2.3 Kasifikasi *Non-linier Case* SVM

Pada umumnya, masalah dalam dunia nyata jarang yang bersifat *linier separable*, kebanyakan bersifat *nonlinier*. Untuk menyelesaikan permasalahan *nonlinier*, SVM dimodifikasi dengan menggunakan fungsi *Kernel*. Dalam *nonlinier SVM*, data dipetakan oleh fungsi $\Phi(\boldsymbol{x})$ ke ruang vektor yang berdimensi lebih tinggi. Pada ruang vektor yang baru ini, *hyperplane* yang memisahkan kedua kelas tersebut.



Sumber: Lessmann, 2004

Gambar 2.5 *Hyperplane* pada Non-Linier SVM

Pada ilustrasi gambar diatas bentuk pemetaan *nonlinier* tidak perlu diketahui secara eksplisit. Pemetaan linier ini dapat diperoleh dengan memanfaatkan sebuah fungsi *Kernel*. Fungsi obyektif SVM pada $\Phi(\boldsymbol{x})$ dalam *feature space F* dengan mengganti *kernel linier* $\boldsymbol{x}\boldsymbol{x}^T$ dengan *kernel nonlinier* $K(\boldsymbol{x}, \boldsymbol{x}^T) \in R^{m \times m}$. Sehingga berdasarkan persamaan (2.22) formulasi hasil dari *problem* menjadi :

$$max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x_i}, \boldsymbol{x_j}) \qquad (2.27)$$

dengan syarat $\sum_{i=1}^{n} \alpha_i y_i = 0$ dengan mendapatkan nilai α_i prediksi dapat dilakukan dengan rumus :

$$f(x) = sign\left(\sum_{i=1}^{n} \alpha_i y_i \, K(\boldsymbol{x_i}, \boldsymbol{x}) + \hat{b} \right) \qquad (2.28)$$

dimana matriks $K(\boldsymbol{x}_i, \boldsymbol{x})$ merupakan matriks *kernel*. Beberapa fungsi pembentuk matriks *kernel* untuk menyelesaikan permasalahan *nonlinier* adalah sebagai berikut :

a. *Kernel Gaussian (RBF)*

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = \exp\left(-\gamma(\|\boldsymbol{x} - \boldsymbol{x}_i\|^2)\right), \gamma > 0 \qquad (2.29)$$

b. *Kernel Linier*

$$K\left(\boldsymbol{x}_{i,} \boldsymbol{x}\right) \qquad (2.30)$$

c. *Kernel Polinomial*

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = (\gamma \boldsymbol{x}, \boldsymbol{x}^T + d)^d \, , \gamma > 0 \qquad (2.31)$$

Bermacam-macam jenis *kernel* lain dapat digunakan untuk pemetaan pada *feature space* namun ketiga *kernel* tersebut yang paling umum digunakan (Santosa, 2007).

2.3 *Ensemble* of *Support Vector Machine*

Sebuah *ensemble* dari *classifier* adalah kumpulan dari beberapa *classifier* yang masing-masing keputusan digabungkan dalam beberapa cara untuk mengklasifikasikan contoh uji. Salah satu metode *ensemble* dalam SVM adalah *boosting* yaitu sebuah keluarga *ensemble* yang meliputi banyak algoritma yang dapat mengubah *weak learner* menjadi *Strong learner* dimana *Adaptive Boosting* (Adaboost) merupakan salah satu algoritma yang populer (Zhou, 2012). Secara umum *boosting* berfokus pada pembuatan

model klasifikasi pada setiap iterasinya. Data yang digunakan dalam menyusun model klasifikasi bergantung pada model klasifikasi sebelumnya dan fokus pada data yang salah prediksi (memberi bobot yang lebih besar untuk data yang salah prediksi atau klasifikasi). Data yang salah prediksi akan diperbaiki terus menerus oleh model klasifikasi selanjutnya. Tujuan akhir dari algoritma *boosting* adalah menyatukan semua model klasifikasi sehingga diperoleh dari *weak learner* menjadi model klasifikasi *strong* yang dapat mengklasifikasikan data secara benar. *Weak learner* pada Adaboost adalah klasifikasi yang menghasilkan akurasi lebih baik dari *random guessing* (akurasi 0.5 atau *error* 0.5). Maka dibutuhkan klasifikasi dengan *error* < 0.5 agar *weak learner* mampu mengambil keuntungan dari Adaboost. Algoritma *boosting* dengan *classifier* SVM sebagai model pengklasifikasian biasanya disebut dengan algoritma *boosting* SVM.

Metode *boosting* yang digunakan adalah *Adaptive Boosting* atau biasa disebut AdaBoost yang pertama kali diperkenalkan Schapire (1999). Pada prisipnya AdaBoost memberikan bobot yang sama kepada tiap pengamatan di awal iterasi, kemudian bobot-bobot tersebut berubah selama proses iterasi. Bobot berubah-ubah sesuai dengan tingat kesalahan klasifikasi dari masing-masing pengamatan. Apabila pengamatan pada data *training* salah terklasifikasi, maka bobotnya akan lebih besar sehingga peluang pengamatan untuk menjadi data *training* selanjutnya menjadi besar. Hal ini bertujuan untuk memaksa *base classifier* untuk lebih fokus pada pengamatan yang sulit untuk dipelajari. AdaBoost menyusun *final classifier* dengan mengombinasikan sekumpulan *weak classifier* diakhir iterasi *boosting*-nya. Distribusi bobot setiap pengamatan D_1 pada data *training* ditentukan dengan rumus berikut (Schapire, 1999):

$$D_1 : D_1(i) = \frac{1}{n}, \qquad i = 1,2,\dots,n \tag{2.32}$$

dengan n merupakan banyaknya pengamatan. Apabila r merupakan nomor iterasi, maka untuk setiap $r = 1,2,\dots,R$ dilakukan penyusunan *base classifier* yaitu model SVM tunggal h_r dengan bobot D_r. Kemudian menghitung tingkat kesalahan klasifikasi (*error*) data *training* dengan menggunakan formula:

$$h_r: \varepsilon_r = \sum_{i=1}^{n} D_r(i)\ , \left(y_i \neq h_r(\boldsymbol{x_i})\right) \tag{2.33}$$

h_r adalah komponen klasifikasi SVM. Selanjutnya, menghitung nilai α_r yang merupakan pembobot *classifier* f_r dengan rumus:

$$\alpha_r = \frac{1}{2} ln\left(\frac{1-\varepsilon_r}{\varepsilon_r}\right) \tag{2.34}$$

Bobot baru dihitung untuk setiap pengamatan yang mengalami salah klasifikasi dengan rumus pembobot baru yaitu:

$$D_{r+1}(i) = \frac{D_r(i)\ e^{\{-\alpha_r y_i h_r(\boldsymbol{x_i})\}}}{Z_t} \tag{2.35}$$

dimana $\sum_{i=1}^{n} D_{r+1}(i) = 1$ dan Z_t adalah konstanta normalisasi. Data yang pengamatan pengamatan yang mengalami tepat klasifikasi maka bobotnya tetap. Hastie et al. (2008) menjelaskan bahwa untuk kasus peubah respon adalah peubah dengan hanya dua kelas dan tarafnya dinotasikan -1 dan +1, maka dugaan kelas akhir dapat dinyatakan sebagai fungsi:

$$H(\boldsymbol{x}) = sign\left(\sum_{r=1}^{R} \alpha_r h_r(\boldsymbol{x})\right) \tag{2.36}$$

Dengan α_r merupakan pembobot *classifier* h iterasi ke-r dan $h_r(\boldsymbol{x})$ adalah fungsi klasifikasi iterasi ke-r. Persamaan terakhir yang berupa total terboboti hasil dugaan dari setiap pengamatan ini memperjelas pengelompokan metode *boosting* sebagai salah satu metode penggabungan (*ensemble*).

2.4 *Feature Selection*

Pemilihan *feature* atau variabel digunakan sebagai salah satu cara untuk mengatasi *high dimensional data*. Pemilihan *feature* diguanakan pada metode *Ensemble of* SVM dan tidak diguankan pada LORENS. Salah satu cara untuk memilih variabel adalah dengan menghitung tingkat kepentingan variabel atau *feature importance*. *Mean decrease gini* (MDG) merupakan salah satu ukuran tingkat kepentingan *feature* yang dihasilkan dari menggunakan metode *random forest*. Berdasarkan penelitian yang dilakukan oleh Calle & Urrea (2010), metode MDG lebih baik jika dibandingkan dengan metode *mean decrease accuracy* (MDA) dan *gini index* untuk mengukur tingkat kepentingan *feature*, karena

MDG lebih stabil dan menghasilkan hasil yang lebih *robust*. Berikut merupakan rumus untuk menghitung MDG (Marco & Zuccolotto, 2006).

$$MDG_h = \frac{1}{k}\sum_{t=1}^{k}[d(h,t)\, I(h,t)] \tag{2.37}$$

Keterangan:

$d(h,t)$ = Besar penurunan indeks Gini untuk variabel X_h pada simpul t.

$I(h,t)$ = Bernilai 1 jika X_h memilah simpul t, bernilai 0 jika selainnya.

k = Banyaknya pohon dalam *random forest*.

2.5 K-*Fold Cross validation*

Metode *cross validation* merupakan salah satu cara yang digunakan untuk mengevaluasi performa sebuah model dalam melakukan prediksi melalui data *testing* dan data *training* (Witten, Frank, & Hall, 2001). Metode *cross validation* membagi data menjadi *k folds* atau partisi yang memiliki jumlah yang seimbang. Masing-masing partisi berperan sebagai data *traning* sekaligus data *testing* pada saat gilirannya. Jelasnya, metode *cross validation* menggunakan satu partisi data sebagai data *testing* dan k-1 sisanya sebagai data *training*. Prosedur ini terus berulang sampai semua partisi data telah menjadi data *testing*. Metode atau prosedur ini dikenal dengan nama *k fold cross validation*. Namun apabila prosedur stratifikasi juga dilakukan, metode ini disebut *stratified k fold cross validation*. Misalnya digunakan 10 *folds* untuk metode *cross validation*. Pertama, data dibagi secara acak menjadi 10 bagian dengan proporsi sama. Selanjutnya metode *cross validation* ini dijalankan sebanyak 10 kali dengan data *training* yang berbeda. Dimana setiap set data memiliki jumlah yang sama dengan set data yang lainnya. Pengujian telah dilakukan dengan menggunakan data yang berbeda dan teknik belajar yang berbeda pula, kesimpulanya 10 *folds* merupakan *folds* terbaik untuk mendapatkan kesalahan yang terbaik. Metode 10 *folds cross validation* telah menjadi metode standar dalam *machine learning* dan *data mining.* Metode evaluasi ini juga menunjukan dengan penggunaan stratifikasi dapat

meningkatkan akurasi prediksi. Berikut merupakan langkah dalam melakukan metode *Cross validation* (Witten, *et al.*, 2011).

1. Memisahkan variabel respon berdasarkan kelasnya.
2. Membagi keseluruhan pengamatan menjadi 10 partisi pada masing-masing kelas.
3. Menggabungkan kedua kelas pada bagian yang sama.
4. Mengunakan salah satu partisi sebagai data *testing* dan menggunakan bagian kedua sampai ke sepuluh menjadi data *training* pada *folds* yang pertama. Terus berlanjut sampai *folds* ke sepuluh menjadi data *testing*.

2.6 Evaluasi Performa Klasifikasi

Ukuran dasar yang digunakan mengukur dan mengevaluasi performa klasifikasi adalah sensitivitas, spesifitas dan akurasi. Akurasi menunjukkan efektifitas *classifier* secara menyeluruh. Akurasi biasa digunakan untuk data yang *balanced.* Semakin besar akurasi, maka kinerja *classifier* semakin baik. Berikut ini merupakan tabel tabulasi silang dan rumus untuk menghitung ukuran ketepatan klasifikasi (Okun, 2011).

Tabel 2.1 Tabel Tabulasi Silang Klasifikasi Aktual dan Klasifikasi Prediksi

		Kelas Aktual	
		p (+)	*n* (-)
Kelas Prediksi	*p* (+)	*True Positive*	*False Positive*
	n (-)	*False Negative*	*True Negative*

Akurasi yang digunakan adalah akurasi total dari keseluruhan ketepatan *classifier* dalam mengidentifikasi ke kelas positif maupun negatif. Berikut ini merupakan rumus dari akurasi:

$$Total\ Akurasi = \frac{TP + TN}{(TP + TN + FP + FN)} \tag{2.38}$$

Selain itu, akurasi setiap kelas ada sensitifitas dan spesifisitas. Sensitifitas digunakan untuk mengukur efektifitas sebuah *classifier* untuk mengidentifikasi kelas positif, sedangkan spesifisitas menggukur efektifitas *classifier* dalam mengidentifikasi kelas negatif. Berikut ini merupakan rumus dari sensitifitas dan spesifisitas:

$$Sensitifitas = \frac{TP}{(TP + FN)} \tag{2.39}$$

$$Spesifisitas = \frac{TN}{(TN + FP)} \quad (2.40)$$

TP (*True Positive*) adalah total senyawa toksisitas rendah yang tepat terprediksi ke dalam kelas positif. TN (*True Negatif*) adalah total senyawa toksisitas tinggi yang tepat terprediksi ke dalam kelas negatif. FP (*False Positif*) adalah total toksisitas tinggi yang terprediksi ke dalam kelas positif. FN (*False Negatif*) adalah total senyawa toksisitas rendah yang terprediksi ke dalam kelas negatif.

Perhitungan dari akurasi adalah dengan mengidentifikasi data *testing* dengan data prediksi dari model. Data *testing* didapatkan dari 10-*fold cross validation* dengan data *testing* yaitu 1 set data pada setiap iterasi dengan iterasi sebanyak 10 kali. Hasil rata-rata akurasi adalah rata-rata dari hasil perhitungan akurasi setiap data *training* dan *testing*.

2.7 Toksisitas

Kata racun ”toxic” adalah bersaral dari bahasa Yunani, yaitu dari akar kata *tox*, dimana dalam bahasa Yunani berarti panah. Dimana panah pada saat itu digunakan sebagai senjata dalam peperangan, yang selalu pada anak panahnya terdapat racun. Secara sederhana dan ringkas, toksikologi dapat didefinisikan sebagai kajian tentang hakikat dan mekanisme efek berbahaya (efek toksik) berbagai bahan kimia terhadap makhluk hidup dan sistem biologik lainnya. Pada umumnya efek berbahaya / efek farmakologik timbul apabila terjadi interaksi antara zat kimia (tokson atau zat aktif biologis) dengan reseptor. Terdapat dua aspek yang harus diperhatikan dalam mempelajari interakasi antara zat kimia dengan organisme hidup, yaitu kerja farmakon pada suatu organisme (aspek farmakodinamik / toksodinamik) dan pengaruh organisme terhadap zat aktif (aspek farmakokinetik / toksokinetik) aspek ini akan lebih detail dibahas pada sub bahasan kerja toksik (Wirasuta & Niruri, 2006).

Sifat toksik suatu tokson sangat ditentukan oleh dosis (konsentrasi tokson pada reseptornya). Artinya kehadiran suatu zat yang berpotensial toksik di dalam suatu organisme belum tentu menghasilkan juga keracunan. Sehingga apabila batas konsentrasi

toksiknya terlampaui, barulah akan muncul efek toksik. Efek atau kerja toksik seperti ini lebih dikenal dengan efek toksik yang bersifat kronis.Bidang yang paling berkaitan dengan toksikologi adalah farmakologi, karena ahli farmakologi harus memahami tidak hanya efek bermanfaat zat kimia, tetapi juga efek berbahayanya yang mungkin diterapkan pada penggunaan terapi. Farmakologi pada umumnya menelaah efek toksik, mekanisme kerja toksik, hubungan dosis respon, dari suatu tokson.

Kerja utama dari toksikologi forensik adalah analisis racun baik kualitatif maupun kuantitatif sebagai bukti dalam tindak kriminal (forensik) di pengadilan.Toksikologi juga berperan dalam pengembangan obat baru, sudah menjadi prasat dalam pengembangan obat baru harus dibarengi baik uji toksisitas akut maupun toksisitas krinis, dengan persyaratan uji yang ketat. Penilaian tentang keamanannya merupakan tantangan dan tunggung jawab toksikologi.

BAB III
METODOLOGI PENELITIAN

Penjelasan mengenai data yang digunakan dalam analisis dijelaskan pada Bab ini. Sub-bab yang dibahas merupakan sumber data, variabel penelitian, dan langkah analisis.

3.1 Sumber Data dan Variabel Penelitian

Data yang digunakan dalam penelitian ini adalah data sekunder yang berasal dari penelitian yang dilakukan oleh Ariyasu et al, (2014) yaitu data senyawa pelindung sel normal pada *Radioprotector*. Variabel prediktor pada penelitian ini berjumlah 217 variabel dan variabel respon berjumlah 84.

Terdapat dua jenis tingkat toksisitas yaitu toksisitas rendah dan tinggi yang dihitung berdasarkan *death rate cell.* Observasi yang dilakukan berupa komponen senyawa yang telah diuji toksisitasnya berdasarkan tingkat kematian sel (*death rate cell*). Berikut Tabel 3.1 merupakan variabel respon yang digunakan dalam penelitian.

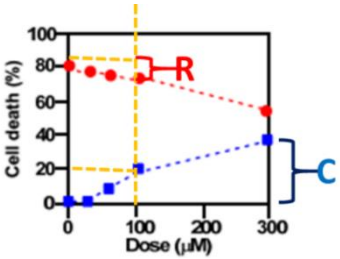
Tabel 3. 1 Variabel Respon

Variabel	Target Kelas	Skala
Respon (Y)	$Y(0)$ = Toksisitas dengan tingkat kematian sel normal 20-100% (toksisitas tinggi) $Y(1)$ = Toksisitas dengan tingkat kematian sel normal <20% (toksisitas rendah)	Nominal

Variabel respon pada Tabel 3.1 merupakan target kelas yang terdiri dari 2 kelas (biner). Klasifikasi didasarkan pada tingkat kematian sel, kelas 1 untuk toksisitas dengan tingkat kematian sel kurang dari 20% dan kelas 0 untuk toksisitas dengan tingkat kematian sel 20-100%. Penentuan *threshold* pada toksisitas dilakukan dengan perhitungan persentase maksimum tingkat kematian sel (*cell death (%))* yaitu pada dosis $300\mu M$ dikurangi dengan persentase minimum tingkat kematian yaitu pada dosis $100\mu M$, sehingga didapatkan *threshold* tingkat kematian sel untuk toksisitas yaitu 20%. Berikut ini merupakan rumus untuk mendapatkan *threshold.*

$$C = cell\ death\%\ (maximum) - cell\ death\%(minimum)$$

Nilai C merupakan *threshold* pada toksisitas dan R merupakan *threshold* pada radiasi proteksi. Berikut ini adalah gambar pada penelitian Ariyasu *et al* (2014) dalam menentukan *threshold* untuk kelas klasifikasi pada toksisitas pada Gambar 3.1



Sumber : Ariyasu, *et al* (2014)

Gambar 3.1 Penentuan *threshold* dalam kelas toksisitas

Berikut ini merupakan struktur data dan variabel prediktor yang digunakan dalam penelitian ini. Struktur data yang digunakan pada pelitian dapat dilihat pada Tabel 3.2.

Tabel 3. 2 Struktur Data Penelitian

No	Variabel Respon	Variabel Prediktor					
		x_1	x_2	x_3	x_4	$\cdots$	x_{217}
1	0	$x_{1,1}$	$x_{2,1}$	$x_{3,1}$	$x_{4,1}$	$\cdots$	$x_{217,1}$
2	0	$x_{1,2}$	$x_{2,2}$	$x_{3,2}$	$x_{4,2}$	$\cdots$	$x_{217,2}$
3	1	$x_{1,3}$	$x_{2,3}$	$x_{3,3}$	$x_{4,3}$	$\cdots$	$x_{217,3}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
84	0	$x_{1,84}$	$x_{2,84}$	$x_{3,84}$	$x_{4,84}$	$\cdots$	$x_{217,84}$

Tabel 3. 3 Variabel Prediktor

Nama Variabel	Skala
x_1 = pKa (tingkat keasaman)	Rasio
x_2 = Jumlah atom Br (Br_Count)	Rasio
x_3 = Jumlah atom C (C_Count)	Rasio
x_4 = Jumlah atom CI (Cl_Count)	Rasio
x_5 = Jumlah atom F (F_Count)	Rasio
x_6 = Jumlah atom H (H_Count)	Rasio
x_7 = Jumlah atom I (I_Count)	Rasio
x_8 = Jumlah atom N (N_Count)	Rasio
$\vdots$	
x_{212} = Luas area molekul di bidang YZ (Shadow_YZfrac)	Rasio

Tabel 3. 3 Variabel Prediktor (Lanjutan)

Nama Variabel	Skala
x_{213} = Panjang molekul pada dimensi Z (Shadow_Zlength)	Rasio
x_{214} = Jumlah molekul 3D Polar SASA (Molecular_3D_PolarSASA)	Rasio
x_{215} = Jumlah molekul 3D SASA (Molecular_3D_SASA)	Rasio
x_{216} = Jumlah molekul 3D SAVol (Molecular_3D_SAVoL)	Rasio
x_{217} = Volume molekul (Molecular_Volume)	Rasio

Pada Tabel 3.3 merupakan variabel prediktor yang digunakan yaitu berjumlah 217 prediktor. Variabel prediktor merupakan karakteristik senyawa obat kanker yang didapatkan dari penelitian Ariyasu et al, (2014) dengan menghitung melalui Discovery Studio (*software* pemodelan 3D yang dapat menghitung sifat kimia).

3.2 Langkah Analisis

Dalam penelitian ini analisis klasifikasi dilakukan dengan metode LORENS dan AdaBoost-SVM. Kedua metode klasifikasi tersebut dievaluasi menggunakan metode *Cross validation*. Langkah-langkah analisis dalam penelitian ini adalah sebagai berikut:

1. Melakukan analisis klasifikasi menggunakan metode *Logistic Regression Ensembles* dengan prosedur evaluasi *10-folds Cross validation* dengan langkah sebagai berikut.
   a. Membuat analisa deskriptif terhadap data yaitu diagram lingkaran untuk melihat perbandingan kelas toksisitas rendah dan tinggi dan diagram batang pada setiap variabel terhadap kelas.
   b. Membagi data menjadi 10 bagian yang sama.
   c. Mengambil sampel satu bagian data sebagai data *testing* dan menggunakan 9 bagian data lainnya sebagai data *training* sesuai pada pembahasan sub-bab 2.5.
   d. Menentukan banyak partisi (k) dimana k = 5, 7, 10, 12, 15, 17, 20, 22, 25, 30, 40, dan 50 yang dilakukan dengan rata secara *random sampling.* Penentuan jumlah partisi yang digunakan menggunakan rumus pada persamaan (2.10). Variabel yang berlebih pada partisi

secara otomatis masuk kedalam partisi secara berurut kedalam partisi pertama, kedua dan seterusnya.

e. Menentukan banyak *ensemble* (*j*) dan nilai *threshold*, dimana *j*=11 dan *threshold* 0,5 dan *threshold* optimal. Jumlah *ensemble* yang ditentukan berdasarkan Lim *et al*, (2010) yaitu 10 atau lebih, namun agar menghindari seri dalam *majority voting* maka *ensemble* berjumlah 11.
f. Menyusun model *Logistic Regression* masing-masing subruang partisi dari data *training*. Variabel prediktor berupa hasil dari langkah (2e) untuk masing-masing partisi. Variabel respon berupa data *training* hasil dari langkah (2d).
g. Mendapatkan model LR dari data *training* dan memprediksi data *testing* berdasarkan model LR *training.*
h. Mendapatkan nilai probabilitas klasifikasi untuk setiap pengamatan pada masing-masing model LR. Menghitung rata-rata nilai probabilitas setiap partisi.
i. Mengklasifikasikan pengamatan berdasarkan rata-rata probabilitas, apabila probabilitasnya lebih dari *threshold* (0,5) maka masuk kelas positif, sebaliknya jika probabilitas kurang dari 0,5 maka masuk kelas negatif.
j. Mengulangi langkah (b) hingga (j) sampai terbentuk 11 *ensemble*.
k. Mencari nilai prediksi terbanyak (*majority voting*) masing-masing pengamatan diantara semua *ensemble*.
l. Menghitung nilai *threshold* optimal sesuai pada persamaan (2.11).
m. Membandingkan hasil dari langkah g dengan nilai *threshold* 0,5 dan *threshold* optimal.
n. Mengulangi semua langkah hingga semua data telah diperlakukan sebagai data *training* dan data *testing*

2. Melakukan analisis klasifikasi menggunakan metode *Support Vector Machine Ensemble* (AdaBoost-SVM)

berdasarkan Schapire (1999) dengan langkah sebagai berikut.

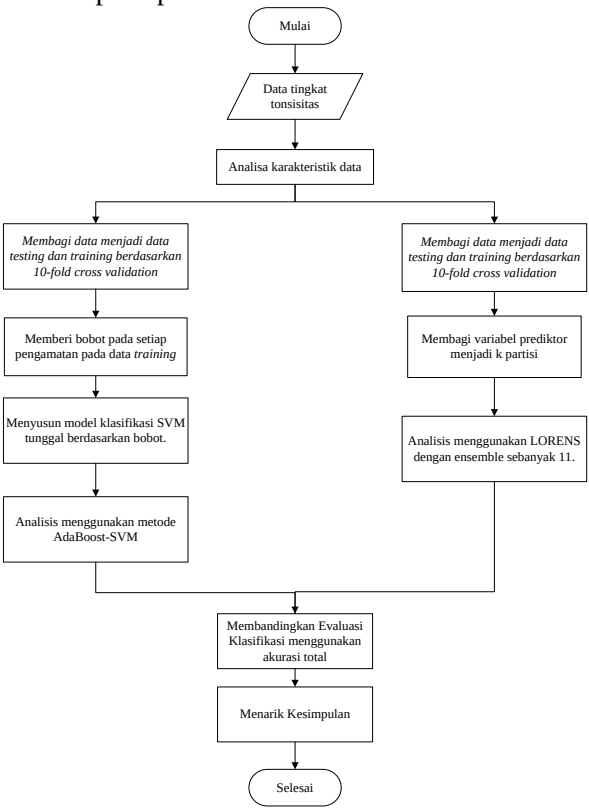
A. Melakukan pemilihan variabel berdasarkan tingkat kepentingan variabel menggunakan nilai *Mean Decrease Gini* untuk klasifikasi AdaBoost-SVM dengan cara sebagai berikut.
   i. Menghitung nilai kepentingan variabel berdasarkan *mean decrease gini* dengan rumus pada persamaan (2.37)
   ii. Memilih jumlah varibel yang akan digunakan dalam penelitian yaitu variabel prediktor dengan peringkat tertinggi yaitu 5%, 10%, 25%, dan 35% dari keseluruhan variabel.
B. Melakukan stratifikasi pada data.
C. Membagi data menjadi 10 bagian yang sama.
D. Mengambil sampel satu bagian data sebagai data *testing* dan menggunakan 9 bagian data lainnya sebagai data *training* sesuai pada pembahasan sub-bab 2.5.
E. Menentukan bobot awal setiap pengamatan sesuai persamaan (2.32).
F. Untuk setiap iterasi $r = 1,2,\dots,R$, melakukan proses berikut.
   i. Menyusun model klasifikasi SVM tunggal seperti pada langkah berikut ini.
      a. Menentukan fungsi kernel yang digunakan yaitu kernel RBF.
      b. Menentukan nilai parameter *C* dan *Gamma* optimal yang digunakan dengan *grid search*.
      c. Melakukan klasifikasi SVM tunggal menggunakan parameter optimal dengan memperhatikan bobot awal sehingga didapatkan *classifier* $h_r(\boldsymbol{x})$.
   ii. Menghitung tingkat kesalahan klasifikasi model SVM tunggal yang ditulis pada persamaan (2.33).
   iii. Menghitung pembobot *classifier* α_r yang ditulis pada persamaan (2.34).

iv. Menentukan bobot baru untuk setiap pengamatan dengan mengalikan bobot dengan α_r yang ditulis pada persamaan (2.35).
v. Melakukan proses (i) sampai (iv) hingga didapatkan model klasifikasi terbaik.

G. Mendapatkan prediksi dari klasifikasi *boosting* SVM.

3. Menghitung akurasi total dari model klasifikasi LORENS dan AdaBoost-SVM yang terbentuk sesuai dengan pesamaan (2.36) dan memilih metode terbaik berdasarkan akurasi terbaik.

Berdasarkan langkah analisis diatas, diagram alir (*flow chart*) analisis dalam penelitian ini secara umum dapat diilustrasikan seperti pada Gambar 3.2 berikut ini.



Gambar 3.2 Diagram Alir Penelitian

BAB IV
ANALISIS DAN PEMBAHASAN

Pada bab ini membahas hasil analisis berdasarkan pengolahan data yang telah dilakukan. Metode yang digunakan dalam analisis adalah klasifikasi dengan LORENS dan *Support Vector Machine Ensemble* menggunakan data Senyawa pelindung sel normal pada *Radioprotector*.

4.1 Klasifikasi Menggunakan *Logistic Regression Ensembles* (LORENS)

Metode LORENS merupakan salah satu modifikasi metode logistic dengan konsep *ensemble*. Pada dasarnya, regresi logistic mengansumsikan variabel-variabel prediktor yang digunakan saling independen atau tidak ada multikolinearitas antar variabel. Oleh karena itu, dengan menggunakan LORENS, dilakukan pengembangan regresi logistik agar menurunkan besarnya kemungkinan multikolinieritas variabel karena variabel dibagi ke dalam beberapa partisi.

4.1.1 Ilustrasi Metode LORENS

Bagian ini menjelaskan bagaimana LORENS melakukan klasifikasi terhadap data. Data yang digunakan adalah data yang berdimensi tinggi. Langkah-langkah yang dijelaskan sesuai pada langkah analisis pada sub-bab 3.2. Berikut ini merupakan perhitungan analisis menggunakan metode LORENS.

1. Data sampel yang digunakan merupakan data penelitian ini namun hanya menggunakan beberapa variabel dan pengamatan yang sedikit. Jumlah variabel prediktor (p) yang digunakan sebanyak 15 dan jumlah data (n) sebanyak 9. Tabel 4.1 merupakan data yang digunakan dimana y merupakan variabel respon, sedangkan $x_1, x_2, \dots, x_{15}$ merupakan variabel prediktor. Data tersebut tergolong dalam data berdimensi tinggi karena jumlah prediktor lebih banyak dari jumlah pengamatan atau bisa dituliskan $p > n$.

Tabel 4. 1 Data Ilustrasi LORENS

No	Variabel Respon	Variabel Prediktor							
	Y	**X_1**	**X_2**	**X_3**	**X_4**	**X_5**	**X_6**	**…**	**X_{15}**
1	0	20	0.58	0.04	55.85	6.17	49.68	…	0.7
2	0	11.4	0.59	0.04	65.25	1.95	63.3	…	0.69
3	1	7.9	0.81	0.06	29.36	-0.17	29.53	…	0.73
4	1	8.3	0.84	0.07	31.01	-1	32.01	…	0.74
5	1	20	0.7	0.03	7.59	3.06	4.53	…	0.79
6	0	20	0.88	0.09	29.85	2.76	27.09	…	0.78
7	1	6.4	0.83	0.07	24.46	-1.04	25.5	…	0.77
8	0	20	0.66	0.06	80.86	3.93	76.93	…	0.74
9	1	20	0.55	0.04	83.97	5.5	78.47	…	0.71

2. Jumlah partisi yang digunakan adalah 5. Jumlah ini dipertimbangkan dari perhitungkan jumlah minimal partisi yang bisa digunakan. Data yang digunakan dalam *training* adalah sebanyak n=9 dan p=15, sehingga jumlah variabel maksimal yang bisa digunakan dalam satu partisi adalah $n-1=9-1=8$. Hal ini karena dalam satu partisi tidak boleh sama dengan atau lebih dari jumlah pengamatan. Jumlah partisi minimal yang bisa digunakan agar memenuhi syarat $p<n$ adalah $m=\frac{p}{n-1}=1.87\approx 2$ partisi, sedangkan jumlah partisi maksimal yang bisa digunakan adalah $m=p=15$ partisi yaitu menghasilkan 1 variabel prediktor untuk masing-masing partisi. Pilihan jumlah partisi yang digunakan adalah 5 partisi sehingga memenuhi syarat karena berada pada rentang partisi 2 hingga 15.
3. Menentukan threshold berdasarkan persamaan (2.11) yaitu sebagai berikut.

$$threshold=\frac{\frac{1}{10}\sum_{i=1}^{9}y_i+0.5}{2}=\frac{0.556+0.5}{2}=0.527$$

Dengan demikian nilai threshold yang digunakan adalah 0.527 dimana apabila probabilitas prediksi ≥ 0.527 maka masuk kelas 1 dan sebaliknya masuk kelas 0.

4. Analisis untuk *ensemble* pertama dilakukan seperti langlah berikut.

a. Variabel prediktor sebanyak 15 dibagi menjadi 5 partisi yaitu masing-masing partisi memiliki 3 variabel prediktor. Pemilihan variabel prediktor ini dilakukan secara *random sampling.* Pada Tabel 4.2 menunjukkan bahwa pembagian variabel prediktor kedalam 5 partisi secara *random* pada *ensemble* pertama. Partisi pertama terdiri dari variabel prediktor X7, X8 dan X14, partisi kedua terdiri dari X3, X10 dan X11, hingga seterusnya. Berikut ini adalah hasil pembagian variabel prediktor kedalam partisi.

Tabel 4. 2 Pembagian Variabel dalam Partisi pada *Ensemble* Pertama

Partisi ke-	Variabel
1	X7, X8, X14
2	X3, X10, X11
3	X5, X9, X12
4	X2, X4, X6
5	X1, X13, X15

b. Tahap kedua yaitu memodelkan data *training* menggunakan metode *Logistic Regression* (LR) sesuai pada persamaan (2.3) untuk setiap partisi, sehingga didapatkan 5 model LR pada satu *ensemble*. Kelima model tersebut menghasilkan nilai parameter $\widehat{\boldsymbol{\beta}}$ untuk masing-masing partisi dengan hasil koefisien regresi pada Tabel 4.4 dan nilai *intercept* pada Tabel 4.3.

Tabel 4. 3 Koefisien Regresi Intercept Setiap Partisi pada *Ensemble* Pertama

Partisi ke-	*Intercept*
1	-7.38
2	732.01
3	9.93
4	10.41
5	-957.61

Tabel 4. 4 Koefisien Regresi pada *Ensemble* Pertama

Variabel Prediktor	Koefisien Regresi	Partisi
X1	-30.65	5
X2	-9.55	4
X3	-5299.61	2
X4	-0.58	4
X5	-0.15	3
X6	0.53	4
X7	-2.39	1
X8	3.21	1
X9	-0.04	3
X10	11.17	2
X11	2.07	2
X12	-0.58	3
X13	1298.01	5
X14	16.25	1
X15	451.01	5

Tabel 4.3 dan Tabel 4.4 merupakan koefisien model regresi yang terbentuk pada kedua ruang partisi untuk *ensemble* pertama.

c. Langkah selanjutnya adalah mendapatkan probabilitas untuk masing-masing pengamatan setiap partisi. Berikut ini merupakan probabilitas dan klasifikasi dari *ensemble* pertama.

Tabel 4. 5 Probabilitas pada *Ensemble* Pertama

Pengamatan ke-	Probabilitas	Kelas
1	0.0819	0
2	0.2963	0
3	0.8201	1
4	0.9309	1
5	0.9571	1
6	0.3127	0
7	0.8943	1
8	0.0630	0
9	0.6437	1

Probabilitas pada Tabel 4.5 didapatkan dengan cara mensubtitusi nilai parameter pada Tabel 4.3 dan Tabel 4.4 kedalam persamaan regresi pada persamaan (2.3) untuk setiap

partisi. Sehingga didapatkan lima probabilitas untuk masing-masing pengamatan. Kelima probabilitas tersebut dirata-rata sehingga didapatkan satu nilai rata-rata probabilitas sebanyak pengamatan (n). Selanjutnya dilakukan klasifikasi berdasarkan rata-rata probabilitas, yaitu apabila nilai probabilitas $\geq$ *threshold* (0.527) maka pengamatan masuk kelas 1 (positif) dan apabila probabilitas < 0.527 maka pengamatan masuk kelas 0 (negatif).

5. Langkah 4 merupakan analisis LORENS menggunakan satu *ensemble*. Selanjutnya dilakukan berulang sampai terbentuk *j ensemble* dimana digunakan jumlah 3 *ensemble*. Pada *ensemble* ke-2 dan ke-3 mengulangi prosedur pada langkah 4 sampai didapatkan klasifikasi kelas pada masing-masing pengamatan. Dalam mengevaluasi dar performa klasifikasi, digunakan nilai akurasi. Nilai akurasi didapatkan dari *majority voting* antara ketiga *ensemble*. Kelas pada *ensemble* 1, 2 dan 3 untuk setiap pengamatan dihitung berdasarkan suara terbanyak. Berikut ini merupakan hasil dari keseluruhan analisis LORENS dengan 3 *ensemble*.

Tabel 4. 6 Hasil Klasifikasi LORENS dengan 3 *Ensemble*

Pengamatan ke-	Probabilitas *Ensemble* ke-			Kelas Akhir
	1	2	3	
1	0.0819 (0)	0.2731 (0)	0.2046 (0)	0
2	0.2963 (0)	0.3992 (0)	0.2199 (0)	0
3	0.8201 (1)	0.7366 (1)	0.8357 (1)	1
4	0.9309 (1)	0.7769 (1)	0.8494 (1)	1
5	0.9571 (1)	0.8708 (1)	0.8785 (1)	1
6	0.3127 (0)	0.4161 (0)	0.3279 (0)	0
7	0.8943 (1)	0.8352 (1)	0.9136 (1)	1
8	0.0630 (0)	0.2043 (0)	0.1702 (0)	0
9	0.6437 (1)	0.4878 (0)	0.6004 (1)	1

6. Evaluasi klasifikasi yang digunakan adalah akurasi dengan rumus pada persamaan (2.40). Hasil kelas akhir pada Tabel 4.6 merupakan hasil akhir dari LORENS berdasarkan *majority voting*. Hasil klasifikasi pada data ilustrasi dapat ditampilkan dalam tabulasi silang seperti berikut.

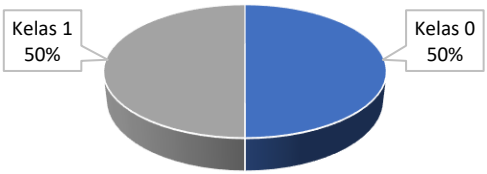
Tabel 4. 7 Tabulasi Silang Metode LORENS pada Data Ilustrasi

		Kelas Aktual	
		+	-
Kelas Prediksi	+	5	0
	-	0	4

Didapatkan hasil bahwa terdapat 5 pengamatan yang tepat terklasifikasi pada kelas positif atau kelas 1 dan terdapat 4 pengamatan yang tepat terklasifikasi pada kelas negatif atau kelas 0. Nilai akurasi yang dihasilkan pada tabel diatas berdasarkan persamaan (2.38) adalah 100%. Akurasi tersebut merupakan akurasi yang dihasilkan dari data *training*. Prediksi menggunakan data baru (data *testing*), bisa dilakukan dengan menggunakan koefisien regresi yang dihasilkan oleh model data *training*.

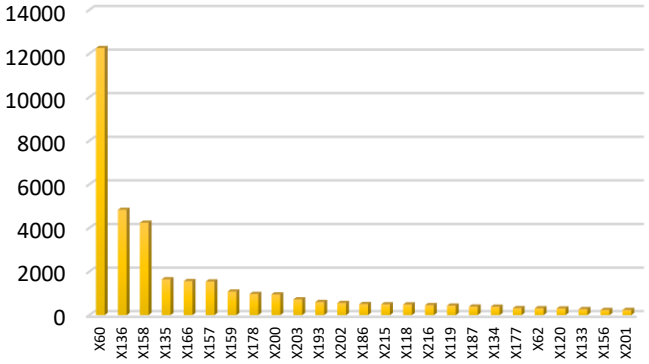
4.1.2 Klasifikasi Senyawa Penyusun Obat Kanker Menggunakan LORENS

Sebelum melakukan analisis klasifikasi senyawa obat kanker berdasarkan toksisitas menggunakan LORENS, dilakukan Analisa deskriptif terlebih dahulu. Analisa statistika deskriptif digunakan untuk mengetahui karakteristik data senyawa pelindung sel normal pada *radioprotector*. Penelitian mengenai klasifikasi senyawa ini digunakan untuk mengetahui senyawa-senyawa yang baik untuk pelindung sel normal pada *radioprotector* berdasarkan tingkat toksisitas. Penelitian ini menggunakan variabel respon yang bersifat biner, yaitu senyawa dengan toksisitas rendah dan tinggi. Gambar 4.1 berikut menunjukan proporsi kelas respon yang digunakan dalam penelitian ini.



Gambar 4.1 Perbandingan Kelas Respon Senyawa pelindung sel normal pada *Radioprotector*

Gambar 4.1 menunjukkan bahwa terdapat dua kategori senyawa pelindung sel normal pada *Radioprotector* berdasarkan tingkat toksisitas yaitu toksisitas rendah (kelas 0) dan toksisitas tinggi (kelas 1) yang memiliki proporsi yang sama (*balanced*). Jumlah senyawa keseluruhan sebanyak 84 senyawa dengan proporsi masing-masing kelas 0 dan 1 sama yaitu 50%. Hal ini menunjukkan bahwa data sudah *balanced*, sehingga nilai batas *threshold* yang digunakan adalah 0.5 namun digunakan *threshold* yang optimal untuk membandingkan hasil rata-rata akurasi yang didapat dari kedua jenis *threshold*. Klasifikasi dilakukan dengan menggunakan 217 variabel prediktor yang merupakan komponen penyusun senyawa berupa karakteristik dari senyawa tersebut.

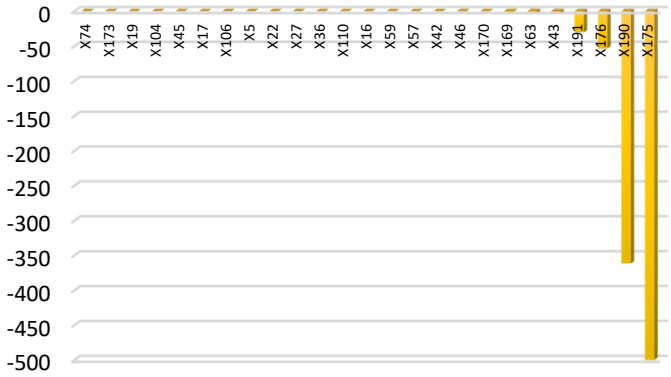


Gambar 4.2 Rata-rata 25 Variabel Tertinggi

Komponen penyusun senyawa karakteristik dari Senyawa pelindung sel normal pada *radioprotector* yang berupa rata-rata dari masing-masing variabel dapat ditunjukkan pada gambar 4.2. Gambar 4.2 merupakan diagram batang yang berupa 25 variabel rata-rata tertinggi dari variabel karakteristik penyusun senyawa pelindung sel normal pada *Radioprotector*. Gambar tersebut menunjukkan perbedaan signifikan pada nilai setiap variabel, dibuktikan dengan rata-rata variabel X60 memiliki nilai terbesar namun variabel terbesar kedua dan ketiga yaitu X136 dan X158 hanya sepertiga dari nilai variabel X60. Pada variabel X135 sampai

X201 memiliki tinggi batang yang hampir sama sehingga memiliki nilai rata-rata yang tidak jauh berbeda.

Diagram batang pada gambar 4.3 menunjukkan diagram berupa rata-rata dari variabel dengan 25 variabel rata-rata terendah. Dapat dilihat bahwa terdapat perbedaan rata-rata pada beberapa variabel yaitu variabel X190 dan X175.



Gambar 4.3 Rata-rata 25 Variabel Terendah

Analisis deskriptif diatas merupakan langkah awal untuk mengetahui karakteristik dari data yang akan dianalisis, selanjutnya dilakukan analisis klasifikasi menggunakan metode LORENS. Analisis *Logistic Regression Ensembles* (LORENS) merupakan analisis yang menggunakan pendekatan komputasional dalam klasifikasi. Variabel prediktor pada analisis LORENS dialokasikan ke dalam ruang-ruang partisi yang terbentuk. Pengalokasian sampel dilakukan dengan cara *random sampling*. Proses tersebut berulang terus sampai dengan jumlah *ensemble* yang ditentukan. Dalam penelitian ini, 217 variabel prediktor yang digunakan akan dialokasikan ke dalam ruang-ruang partisi yang terbentuk. Prediksi variabel respon dapat dilakukan setelah model didapatkan, dimana prediksi yang dihasilkan menggunakan *majority voting*.

Metode evaluasi yang digunakan dalam klasifikasi adalah *Cross validation*. Metode *Cross validation* ini diharapkan mampu meningkatkan performa dari klasifikasi. Penelitian ini

menggunakan 10-*folds cross validation* dengan besar partisi yang berbeda yaitu partisi sebesar 5, 7, 10, 12, 15, 17, 20, 22, 25, 30, 40, dan 50. *Threshold* yang digunakan pada penelitian ini menggunakan *threshold* 0.5 dan *threshold* optimal. Jumlah *ensemble* yang umum digunakan adalah 10, namun penelitian ini menggunakan *ensemble* sebesar 11 untuk menghindari persamaan *vote* saat *majority voting.*

Dalam analisis menggunakan metode LORENS, digunakan 10-*fold cross validation* yaitu data akan dibagi menjadi 10 bagian (*fold*) dengan jumlah yang sama. Masing-masing bagian akan diperlakukan sebagai data *testing* dan data *training*. Data pada *fold* ke-1 digunakan sebagai data *testing*, sedangkan *fold* ke-2 sampai ke-10 digunakan sebagai data *training*. Data pada *fold* ke-2 digunakan sebagai data *testing* dan sisa *fold* digunakan sebagai data *training*, begitu seterusnya sampai *fold* ke-10. Jumlah model yang terbentuk pada jumlah partisi 5, ensemble 11, dan 10 fold adalah sebanyak 550 model regresi logistik. Secara keseluruhan jumlah model yang terbentuk dengan jumlah *ensemble* 11, 10 *fold* dan 12 jenis partisi adalah 27830 model *binary logistic regression.* Tabel 4.8 merupakan *threshold* optimal setiap *fold* pada setiap jenis partisi dengan nilai yang berbeda-beda.

Tabel 4. 8 *Threshold* Optimal Tiap *Fold* pada Setiap Partisi

***Fold* ke-**	**Partisi 5**	**Partisi 7**	**Partisi 10**	**Partisi 12**	**Partisi 15**	**Partisi 17**
1	0.5100	0.5033	0.5033	0.4833	0.5100	0.4967
2	0.4967	0.5033	0.5033	0.5033	0.5100	0.5100
3	0.5033	0.4900	0.4967	0.5033	0.4967	0.4900
4	0.5033	0.4900	0.4833	0.5167	0.4833	0.5033
5	0.4934	0.5000	0.5000	0.4934	0.5197	0.5066
6	0.5000	0.5066	0.4934	0.5132	0.5000	0.4934
7	0.5000	0.4934	0.5000	0.4934	0.4868	0.4934
8	0.4934	0.5132	0.5132	0.5000	0.5000	0.5132
9	0.4868	0.5000	0.5066	0.5000	0.5066	0.5000
10	0.5132	0.5000	0.5000	0.4934	0.4868	0.4934

Tabel 4.8 *Threshold* Optimal Tiap *Fold* pada Setiap Partisi (Lanjutan)

***Fold* ke-**	**Partisi 20**	**Partisi 22**	**Partisi 25**	**Partisi 30**	**Partisi 40**	**Partisi 50**
1	0.4967	0.5033	0.4900	0.5100	0.4967	0.4967
2	0.4833	0.4900	0.5033	0.4900	0.4767	0.4967
3	0.4833	0.4900	0.5100	0.5033	0.5100	0.4900
4	0.5167	0.5100	0.5167	0.4900	0.5100	0.5033
5	0.5000	0.4934	0.5000	0.5066	0.4934	0.5066
6	0.5000	0.4934	0.4868	0.5066	0.5000	0.4934
7	0.5066	0.4934	0.5000	0.4934	0.5132	0.5132
8	0.5066	0.5132	0.5000	0.5000	0.5000	0.4934
9	0.5000	0.5066	0.4868	0.5132	0.5000	0.5000
10	0.5066	0.5066	0.5066	0.4868	0.5000	0.5066

Tabel 4.8 menunjukkan enam jenis partisi yaitu 5, 7, 10, 12, 15, dan 17 partisi. Pada jumlah partisi 5, *fold* pertama didapatkan *threshold* optimal 0.51 sampai dengan *fold* ke-10 adalah 0.5132. Pada partisi 7, *threshold* optimal pada *fold* ke-1 and ke-2 adalah 0.5033, selanjutnya sampai dengan *fold* ke-10 sebesar 0.5. Nilai *threshold* pada partisi 20 dengan *fold* ke-1 adalah sebesar 0.4967 hingga *fold* ke-10 yaitu 0.5066 sampai dengan nilai *threshold* optimal pada 50 partisi dengan *fold* ke-10. Pembagian data pada 22 partisi memiliki nilai *threshold* optimal tertinggi terletak pada *fold* ke-8 yaitu 0.5132 dan terkecil pada *fold* ke-2 dan ke-3 yaitu 0.49. Berikutnya pada jumlah partisi sampai dengan partisi 50 memiliki nilai *threshold* yang berbeda-beda pada setiap *fold*. Nilai *threshold* optimal tersebut akan digunakan sebagai pembanding dengan nilai probabilitas pada masing-masing prediksi setiap model. Apabila nilai probabilitasnya lebih dari nilai *threshold* optimal, maka senyawa akan diklasifikasikan sebagai senyawa dengan toksisitas tinggi, jika kurang dari nilai *threshold* maka diklasifikasikan sebagai senyawa dengan toksisitas rendah. Dalam menghitung nilai total *accuracy*, perlu dilakukan perhitungan tabel tabulasi silang pada masing-masing partisi dan *threshold* yang digunakan. Berikut merupakan hasil klasifikasi senyawa obat kanker menggunakan LORENS dengan *threshold* optimum ditunjukkan pada Tabel 4.9.

Tabel 4. 9 Tabulasi Silang Hasil Klasifikasi pada Setiap Partisi

			Kelas Prediksi			
			***Threshold* 0,5**		***Threshold* Optimal**	
			+	-	+	-
Kelas Aktual	5 partisi	+	24	18	27	15
		-	16	26	22	20
	7 partisi	+	25	17	24	18
		-	16	26	17	25
	10 partisi	+	27	15	25	17
		-	25	25	18	24
	12 partisi	+	28	14	23	29
		-	16	26	18	24
	15 partisi	+	26	16	25	17
		-	19	23	21	21
	17 partisi	+	29	13	27	15
		-	21	21	21	21
	20 partisi	+	34	8	29	13
		-	21	21	20	22
	22 partisi	+	32	10	30	12
		-	19	23	19	23
	25 partisi	+	28	14	33	9
		-	18	24	21	21
	30 partisi	+	34	8	34	8
		-	18	24	18	24
	40 partisi	+	34	8	32	10
		-	19	23	19	23
	50 partisi	+	33	9	34	8
		-	19	23	19	23

Hasil dari tabulasi silang pada Tabel 4.9 diatas dapat dihitung nilai total *accuracy*, *specifity*, dan *sensitifity*. Pemilihan model maupun metode terbaik dalam masalah klasifikasi pada penelitian ini menggunakan total *accuracy*. Berikut ini merupakan hasil perhitungan total *accuracy*, *specifity*, dan *sensitifity* pada klasifikasi LORENS dengan *threshold* 0.5 dan *threshold* optimal ditunjukkan pada Tabel 4.10 dan Tabel 4.11.

Tabel 4. 10 Hasil rata-rata akurasi, *Sensitifity*, dan *Specificity* pada *Threshold* 0.5

Jumlah Partisi	***Sensitifity* (%)**	***Specificity* (%)**	***Accuracy* (%)**
5	0.5714	0.6190	0.5952
7	0.5952	0.6190	0.6071
10	0.6429	0.5000	0.6190
12	0.6667	0.6190	0.6429

Tabel 4. 11 Hasil rata-rata akurasi, *Sensitifity*, dan *Specificity* pada *Threshold* 0.5 (Lanjutan)

Jumlah Partisi	*Sensitifity* (%)	*Specificity* (%)	*Accuracy* (%)
15	0.6190	0.5476	0.5833
17	0.6905	0.5000	0.5952
20	0.8095	0.5000	0.6548
22	0.7619	0.5476	0.6548
25	0.6667	0.5714	0.6190
30	**0.8095**	**0.5714**	**0.6905**
40	0.8095	0.5476	0.6786
50	0.7857	0.5476	0.6667

Angka bercetak tebal merupakan akurasi optimal

Tabel 4. 12 Hasil rata-rata akurasi, *Sensitifity*, dan *Specificity* pada *Threshold* Optimal

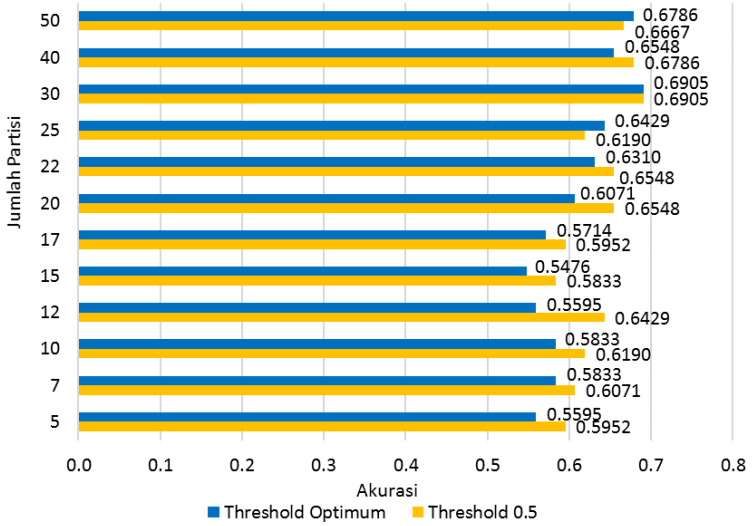
Jumlah Partisi	*Sensitifity* (%)	*Specificity* (%)	*Accuracy* (%)
5	0.6429	0.4762	0.5595
7	0.5714	0.5952	0.5833
10	0.5952	0.5714	0.5833
12	0.4423	0.5714	0.5595
15	0.5952	0.5000	0.5476
17	0.6429	0.5000	0.5714
20	0.6905	0.5238	0.6071
22	0.7143	0.5476	0.6310
25	0.7857	0.5000	0.6429
30	**0.8095**	**0.5714**	**0.6905**
40	0.7619	0.5476	0.6548
50	0.8095	0.5476	0.6786

Angka bercetak tebal merupakan akurasi optimal

Tabel 4.10 menunjukkan bahwa ukuran partisi optimal analisis LORENS pada *threshold* 0.5 untuk klasifikasi Senyawa pelindung sel normal pada *Radioprotector* adalah 30 partisi dengan *accuracy* 69.0476%. Pada Akurasi yang didapatkan pada *threshold* optimal adalah pada 30 partisi dengan *accuracy* 69.0476%. Nilai *accuracy* yang didapatkan pada *threshold* optimal dan *threshold* 0.5 adalah sama yaitu 69.0476%. *Sensitifity* dan *specifity* yang didapatkan pada keduanya juga sama yaitu *sensitifity* 80.95% dan

specifity 57.1429%. Ukuran partisi yang optimal dalam analisis LORENS adalah 30 partisi dengan *threshold* 0.5 maupun optimal.

Pada pembagian variabel prediktor, terdapat 217 variabel yang dialokasikan kedalam 30 ruang partisi, sehingga masing-masing partisi terdiri dari 7 dan 8 variabel prediktor. Didapatkan pembagian variabel sebanyak 7 partisi dengan masing-masing 8 variabel prediktor dan 23 partisi dengan masing-masing 7 variabel prediktor.



Gambar 4.4 Hasil rata-rata akurasi *Threshold* 0.5 dan *Threshold* Optimal

Berdasarkan grafik yang ditunjukkan pada Gambar 4.4 dapat dilihat bahwa semakin tinggi jumlah partisi, nilai akurasi akan cenderung naik. Pada setiap jumlah partisi menunjukkan kecenderungan memiliki nilai akurasi yang lebih tinggi pada *threshold* 0.5 dibanding *threshold* optimal, hal ini ditujukkan dengan diagram batang berwarna kuning lebih panjang dari batang dengan warna biru. Akurasi tertinggi terdapat pada jumlah partisi 30 untuk *threshold* 0.5 maupun *threshold* optimal yaitu sebesar 69.05% sedangkan akurasi terendah terdapat pada jumlah partisi 15 untuk *threshold* 0.5 maupun *threshold* optimal yaitu sebesar 54.76% dan 58.33%.

4.2 Klasifikasi Menggunakan AdaBoost - SVM dengan Seleksi Variabel

Klasifikasi Senyawa pelindung sel normal pada *Radioprotector* selain menggunakan metode LORENS, digunakan metode AdaBoost-SVM juga untuk membandingkan Hasil rata-rata akurasi yang didapatkan. *Ensemble* yang digunakan adalah *boosting* yaitu metode AdaBoost Original. SVM-*Boosting* menggunakan 10-*fold Cross validation* dalam pembagian data *testing* dan *training*. Setiap *fold* akan berperan sebagai data *testing* dan *training* sehingga akan didapat sebanyak 10 akurasi yang kemudian digunakan nilai rata-rata dari kesepuluh akurasi tersebut.

Metode ini menggunakan SVM sebagai *base learning* dengan menyusun model SVM tunggal sebanyak iterasi yang kemudian dihitung nilai bobot dari masing-masing pengamatan dan nilai *error* dari model. Model SVM tunggal pertama akan dimodelkan tanpa bobot, kemudian setelah didapatkan prediksi dari pengamatan maka akan dihitung ketepatan klasifikasi dari prediksi terhadap data *testing*. Selanjutnya, model tersebut akan diklasifikasi lagi dengan menggunakan bobot yang sama pada semua pengamatan dan dihitung nilai *error* dari model. Bobot tersebut digunakan sebagai peluang pengamatan untuk menjadi data *training*. Apabila pengamatan salah terprediksi, maka bobot pada pengamatan tersebut akan bertambah dan apabila pengamatan tepat terprediksi maka bobotnya akan menjadi tetap. Bobot pada setiap pengamatan akan terus berubah sesuai dengan ketepatan dari model dalam memprediksi. Nilai *error* dari model adalah tingkat kesalahan klasifikasi secara keseluruhan. Apabila dilakukan percobaan sebanyak n iterasi maka akan didapatkan sebanyak n nilai *error*.

Dalam memperjelas penerapan AdaBoost-SVM dalam klasifikasi, maka dilakukan ilustrasi terkait penerapan metode AdaBoost-SVM. Bagian ini akan menjelaskan ilustrasi bagaimana proses *boosting* pada AdaBoost-SVM melakukan klasifikasi terhadap data. Langkah-langkah yang dijelaskan sesuai pada langkah analisis pada Bab.III. Berikut ini merupakan perhitungan analisis menggunakan metode AdaBoost-SVM. Data *training* yang digunakan ditampilkan sebagai berikut.

Tabel 4. 13 Data *Training* Ilustrasi Metode AdaBoost-SVM

Pengamatan ke-	Y	X1	X2
1	-1	20	0.58
2	-1	11.4	0.59
3	1	7.9	0.81
4	1	8.3	0.84
5	1	20	0.7
6	-1	20	0.88
7	1	6.4	0.83
8	-1	20	0.66
9	1	20	0.55

Data *training* digunakan sebagai klasifikasi awal yang akan digunakan yaitu berjumlah n=9. Data *training* digunakan untuk memprediksi data *testing*. Jumlah iterasi yang digunakan adalah sebanyak 3.

Tabel 4. 14 Data *Testing* Ilustrasi Metode AdaBoost-SVM

Pengamatan ke-	Y	X1	X2
10	1	7.1	0.75
11	-1	20	0.15
12	1	4.8	0.48
13	1	20	0.4
14	-1	20	0.88

Langkah pertama dalam klasifikasi AdaBoost-SVM adalah masing-masing data *training* diberi bobot sesuai dengan rumus pada persamaan (2.32) yaitu :

$$D_1(i) = \frac{1}{n} = \frac{1}{10}$$

Masing-masing pengamatan memiliki bobot awal yang sama yaitu 0.1. Selanjutnya untuk iterasi pertama hingga ketiga dijelaskan sebagai berikut.

Iterasi ke-1

1. Pada iterasi pertama, data yang digunakan adalah data *training* dengan bobot yang sama untuk setiap pengamatan. Data *training* ini kemudian membentuk data *training* baru dengan cara *sampling* probabilitas tanpa pengembalian sebanyak n pengamatan berdasarkan bobot awal $(D_1(i))$

sebagai probabilitas terpilihnya sampel. Data *training* awal pada iterasi pertama *boosting* yaitu sebagai berikut.

Tabel 4. 15 Data *Training* Baru Iterasi ke-1

Pengamatan ke-	Y	X1	X2	Bobot
1	-1	20	0.58	0.1111
2	-1	11.4	0.59	0.1111
3	1	7.9	0.81	0.1111
4	1	8.3	0.84	0.1111
5	1	20	0.7	0.1111
6	-1	20	0.88	0.1111
7	1	6.4	0.83	0.1111
8	-1	20	0.66	0.1111
9	1	20	0.55	0.1111

Pada iterasi pertama, data *training* baru sama dengan data *training* awal, hal ini karena bobot sebagai probabilitas terpilihnya sampel memiliki nilai yang sama. Pada iterasi selanjutnya, komposisi data *training* akan berubah sesuai berubahnya bobot.

2. Melakukan *training* SVM tunggal pada data *training* baru

Data *training* baru pada Tabel 4.10 digunakan untuk melatih *weak learner* (SVM) dengan kernel RBF, parameter C=1 dan Gamma=0.1. Hasil klasifikasi data *training* yang diperoleh yaitu sebagai berikut.

Tabel 4. 16 Hasil Klasifikasi SVM Iterasi ke-1

Pengamatan ke-	Y awal	Y *training*
1	-1	-1
2	-1	1
3	1	1
4	1	1
5	1	-1
6	-1	1
7	1	1
8	-1	-1
9	1	-1

Angka yang di blok adalah pengamatan yang salah terklasifikasi.

Dari hasil klasifikasi yang diperoleh, diketahui bahwa terdapat empat kesalahan klasifikasi yaitu pada pengamatan ke 2, 5, 6, dan 9.

3. Menghitung *error classifier boosting*
   Perhitungan *error classifier* dilakukan menggunakan rumus pada persamaan (2.33) yaitu didapatkan hasil sebagai berikut.

$$\varepsilon_1 = \sum_{i:h_1(\boldsymbol{x_i})\neq y_i}^{n} D_1(i) = 4(0.1111) = 0.4444$$

   Karena terdapat empat pengamatan yang salah terklasifikasi maka nilai *error* sebesar empat dikali bobot awal yaitu sebesar 0.444.
4. Menghitung bobot *classifier*
   Bobot *classifier* dihitung berdasarkan rumus pada persamaan (2.34) yaitu sebagai berikut.

$$\alpha_1 = \frac{1}{2} ln\left(\frac{1-\varepsilon_1}{\varepsilon_1}\right) = \frac{1}{2} ln\left(\frac{1-0.4444}{0.4444}\right) = 0.1116$$

   Bobot *classifier* pada iterasi pertama sebesari 0.1116.
5. Menghitung bobot pengamatan baru
   Bobot pengamatan selanjutnya berubah sesuai pada ketepatan klasifikasi. Apabila salah terklasifikasi, maka bobot akan lebih besar, dan apabila benar terklasifikasi, maka bobot akan berkurang. Bobot baru ditentukan berdasarkan rumus pada persamaan (2.35). Hasil bobot baru dapat dilihat pada tabel berikut.

Tabel 4. 17 Bobot Baru Iterasi ke-1

Pengamatan ke-	Y awal	Y *training*	Bobot awal	Bobot baru
1	-1	-1	0.1111	0.1
2	-1	1	0.1111	0.125
3	1	1	0.1111	0.1
4	1	1	0.1111	0.1
5	1	-1	0.1111	0.125
6	-1	1	0.1111	0.125
7	1	1	0.1111	0.1
8	-1	-1	0.1111	0.1
9	1	-1	0.1111	0.125

Angka yang di blok adalah pengamatan yang salah terklasifikasi.

Bobot pada Tabel 4.16 tersebut akan digunakan pada iterasi selanjutnya sebagai probabilitas terpilihnya pengamatan sebagai data *training*.

Iterasi ke-2

1. Alur yang digunakan pada iterasi ke-2 sama seperti iterasi ke-1. Pembentukan data *training* pada iterasi ke-2 menggunakan proses sampling dengan pengembalian sebanyak jumlah pengamatan (n). Pengamatan yang memiliki bobot besar menunjukkan semakin besar peluang pengamatan tersebut terpilih menjadi data *training*. Data *training* yang terpilih pada iterasi kedua adalah sebagai berikut.

Tabel 4. 18 Data *Training* Iterasi Kedua

Pengamatan ke-	Y	X1	X2
3	1	7.9	0.81
6	-1	20	0.88
6	-1	20	0.88
2	-1	11.4	0.59
5	1	20	0.7
7	1	6.4	0.83
4	1	8.3	0.84
5	1	20	0.7
3	1	7.9	0.81

2. Melakukan klasifikasi SVM tunggal pada data *training* baru Selanjutnya dilatih SVM dengan parameter yang sama yaitu kernel RBF dengan parameter C=1 dan Gamma=0.1 sehingga diperoleh hasik klasifikasi data *training* sebagai berikut.

Tabel 4. 19 Hasil Klasifikasi SVM Iterasi ke-2

Pengamatan ke-	Y awal	Y *training*
1	-1	1
2	-1	1
3	1	1
4	1	1
5	1	1
6	-1	1
7	1	1
8	-1	1
9	1	1

Angka yang di blok adalah pengamatan yang salah terklasifikasi.

Dari hasil klasifikasi yang diperoleh, diketahui bahwa kesalahan klasifikasi terdapat pada pengamatan ke 1, 2, 6, dan 8.

3. Menghitung *error classifier boosting*. Dengan rumus yang sama pada iterasi pertama, didapatkan nilai *error classifier* sebesar $\varepsilon_2 = 0.45$.
4. Menghitung bobot *classifier*. Bobot *classifier* yang didapatkan adalah sebesar $\alpha_2 = 0.1$.
5. Menghitung bobot pengamatan baru. Bobot baru yang didapatkan ditunjukkan pada Tabel 4.19.

Tabel 4. 20 Bobot Baru Iterasi ke-2

Pengamatan ke-	Y awal	Y *training*	Bobot baru
1	-1	1	0.1111
2	-1	1	0.1389
3	1	1	0.0909
4	1	1	0.0909
5	1	1	0.1136
6	-1	1	0.1389
7	1	1	0.0909
8	-1	1	0.1111
9	1	1	0.1111

Iterasi ke-3

Dengan proses yang sama, dipilih data *training* baru menggunakan bobot pada iterasi kedua. Sampel yang terpilih adalah pengamatan ke-1, 4, 7, 5, 6, 1, 3, 1, dan 8.

Tabel 4. 21 Data *Training* Iterasi Ketiga

Pengamatan ke-	Y	X1	X2
1	-1	20	0.58
4	1	8.3	0.84
7	1	6.4	0.83
5	1	20	0.7
6	-1	20	0.88
1	-1	20	0.58
3	1	7.9	0.81
1	-1	20	0.58
8	-1	20	0.66

Selanjutnya dilatih SVM dengan parameter yang sama yaitu kernel RBF dengan parameter C=1 dan Gamma=0.1 sehingga diperoleh hasik klasifikasi data *training* sebagai berikut.

Tabel 4. 22 Hasil Klasifikasi SVM Iterasi ke-3

Pengamatan ke-	Y awal	Y *training*
1	-1	-1
2	-1	-1
3	1	1
4	1	1
5	1	-1
6	-1	-1
7	1	1
8	-1	-1
9	1	-1

Angka yang di blok adalah pengamatan yang salah terklasifikasi

Dari hasil klasifikasi yang diperoleh, diketahui bahwa terdapat satu kesalahan klasifikasi yaitu pada pengamatan ke 5 dan 9. Dengan rumus yang sama pada iterasi pertama, didapatkan nilai *error classifier* sebesar $\varepsilon_3 = 0.227$. Bobot *classifier* yang didapatkan adalah sebesar $\alpha_3 = 0.6119$. Bobot baru setiap pengamatan yang didapatkan adalah sebagai berikut.

Tabel 4. 23 Bobot Baru Iterasi ke-3

Pengamatan ke-	Y awal	Y *training*	Bobot baru
1	-1	-1	0.0719
2	-1	-1	0.0899
3	1	1	0.0588
4	1	1	0.0588
5	1	-1	0.2500
6	-1	-1	0.0899
7	1	1	0.0588
8	-1	-1	0.0719
9	1	-1	0.0719

Setelah dilakukan proses iterasi sebanyak 3 iterasi diperoleh vector bobot *classifier* $\boldsymbol{\alpha} = [\alpha_1\ \alpha_2\ \alpha_3]^T = [0.1116\ \ 0.1\ \ 0.6119]^T$ kemudian bobot *classifier* tersebut dinormalisasi sehingga

diperoleh $\boldsymbol{\alpha} = [0.1355\ 0.1214\ 0.743]^T$. Setelah didapatkan bobot *classifier*, dilakukan prediksi terhadap data *testing*. Data *testing* yang digunakan sesuai pada Tabel 4.9. Sesuai pada persamaan (2.30), dugaan kelas akhir menggunakan bobot *classifier* dengan model klasifikasi SVM didapatkan hasil sebagai berikut.

Tabel 4. 24 Prediksi Data *Testing*

Pengamatan ke-	Y *testing*	Prediksi	Y prediksi
1	1	0.813214	1
2	-1	-0.31654	-1
3	1	0.058451	1
4	1	-0.70371	-1
5	-1	-0.11891	-1

Selanjutnya dihitung nilai ketepatan klasifikasi menggunakan nilai akurasi dengan rumus pada persamaan (2.40) dengan tabel *confusion matrix* sebagai berikut.

Tabel 4. 25 *Confusion Matrix* Adaboost-SVM Data Ilustrasi

Aktual	Prediksi	
	-1	1
-1	2	1
1	0	2

Nilai akurasi yang didapatkan berdasarkan tabel diatas adalah sebagai berikut.

$$Akurasi = \frac{2+2}{2+2+1} = \frac{4}{5} = 0.8$$

Akurasi yang didapatkan dari data *testing* adalah sebesar 0.8 atau 80%. Nilai akurasi dari testing merupakan tujuan akhir dari ilustrasi klasifikasi.

Penelitian dengan metode AdaBoost-SVM menggunakan *feature selection* dengan melihat tingkat kepentingan variabel berdasarkan hasil dari *mean decrease gini.* Setelah didapatkan variabel yang digunakan, dilanjutkan dengan mencari parameter yang optimal untuk SVM. Kernel yang digunakan adalah RBF berdasarkan penelitian Huang, *et al.* (2007) menyatakan SVM

kernel RBF baik digunakan dengan *boosting.* Sehingga parameter yang digunakan adalah *C* dan *Gamma.*

4.2.1 *Feature Selection*

Data yang digunakan pada penelitian ini memiliki jumlah variabel prediktor lebih banyak daripada jumlah pengamatan (*high dimensional data*), maka AdaBoost-SVM dalam menangani kasus tersebut dilakukan pemilihan fitur (*fitur selection*). Pemilihan fitur digunakan dalam menentukan tingkat kepentingan dari variabel-variabel prediktor dengan metode *Mean Decrease Gini* (MDG). Metode ini merupakan merupakan salah satu ukuran tingkat kepentingan *feature* yang dihasilkan dari menggunakan metode *random forest.* Hasil dari *feature selection* adalah peringkat setiap variabel prediktor dimana semakin tinggi nilai MDG maka variabel terserbut semakin penting. Parameter yang digunakan untuk menghitung MDG dari *random forest* adalah mtry (banyaknya variabel prediktor yang terpilih) dan ntree (banyaknya pohon yang terbentuk) dengan nilai sebagai berikut.

Tabel 4.26 Parameter untuk MDG

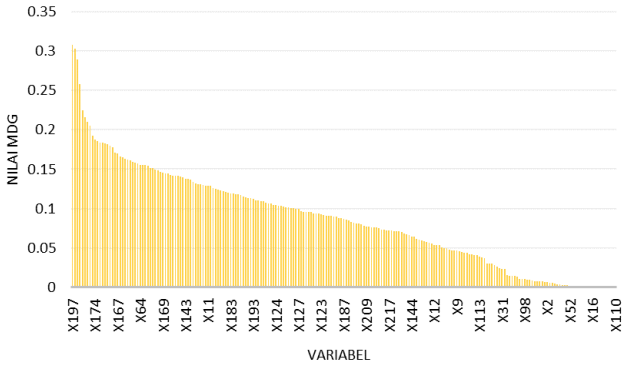
Parameter	Nilai
mtry	1, 2, 3, 4, 5, …, 19, 20
ntree	100, 200, 300, …, 900, 1000

Berdasarkan nilai-nilai parameter mtry dan ntree yang digunakan, didapatkan kombinasi untuk setiap nilai parameter. Dalam menentukan nilai mtry dan ntree yang optimal dilihat dari rata-rata akurasi tertinggi. Berikut ini adalah rata-rata akurasi yang dihasilkan.

Tabel 4.27 Hasil Rata-rata Akurasi Setiap Nilai Parameter

mtry	ntree	Rata-rata Akurasi
1	100	0.5196
1	200	0.5121
1	300	0.5231
1	400	0.5157
…	…	…
10	100	0.6548
…	…	…
20	1000	0.6338

Berdasarkan hasil rata-rata akurasi pada Tabel 4.26 didapatkan akurasi yang optimal adalah 0.6749 dengan parameter pada mtry sebanyak 5 dan ntree sebanyak 800 pohon. Dengan menggunakan parameter tersebut, selanjutnya dilakukan perhitungan MDG untuk mengetahui kepentingan dari setiap variabel. Hasil dari perhitungan MDG ditunjukkan pada Gambar 4.27 sebagai berikut.



Gambar 4.5 Nilai *Mean Decrease Gini* dari terbesar hingga terkecil

Diagram batang pada Gambar 4.5 adalah hasil dari pemilihan fitur menggunakan metode *Mean Decrease Gini* (MDG) dari seluruh variabel yaitu sebanyak 217. Variabel prediktor dengan nilai MDG tertinggi adalah X197 yaitu 0.3075 dan terendah adalah X110 dengan nilai nol. Hal tersebut menunjukkan bahwa berdasarkan metode MDG, variabel yang paling berpengaruh dalam menentukan klasfikasi adalah X197 dan variabel yang tidak memiliki pengaruh adalah X110. Penyebab variabel X110 memiliki nilai MDG nol adalah karena pengamatan pada variabel tersebut memiliki nilai yang sama. Berdasarkan pemilihan fitur tersebut, dilakukan pembagian fitur yang digunakan yaitu 5%, 10%, 25%, dan 35% variabel teratas yang ditunjukkan pada Tabel 4.9. Pemilihan jumlah fitur yang digunakan pada penelitian ini dengan tujuan agar tidak *high dimensional data* atau agar jumlah prediktor tidak lebih banyak dari jumlah pengamatan, sehingga variabel prediktor yang digunakan memiliki jumlah antara 11 – 76 variabel yaitu kurang dari jumlah pengamatan sebesar 84 senyawa.

Tabel 4.28 Pemilihan Variabel berdasarkan Nilai MDG

No	Persentase kepentingan variabel	Jumlah Variabel	Variabel
1	5%	11 variabel	X1, X73, X74, X162, X63, X174, X176, X196, X197, X199, dan X214
2	10%	22 variabel	X1, X56, X66, X71, X73, X74, ..., X176, X196, X197, X199, X214
3	25%	55 variabel	X1, X11, X33, X37, X39, X43, X53, … , X196, X197, X199, X201, X204, X214
4	35%	76 variabel	X1, X6, X11, X33, X37, X38, X39, X43, ..., X199, X201, X204, X211, X214

Pemilihan fitur pada metode AdaBoost-SVM ada 4 jenis berdasarkan peringkat tertinggi yaitu 5%, 10%, 25%, dan 35%. Pemilihan ini digunakan untuk membandingkan nilai akurasi dari keempat jenis fitur tersebut. Jumlah variabel prediktor paling sedikit adalah 11 variabel dengan 5% variabel terpenting dan jumlah variabel terbanyak adalah 76 variabel dengan 35% variabel terpenting.

Pada Adaboost dengan SVM sebagai dasar klasifikasi, terdapat parameter pada SVM untuk membentuk model SVM tunggal yaitu kernel, C, dan *Gamma*, sedangkan pada Adaboost mempunyai parameter banyaknya iterasi. Dalam klasifikasi SVM, perlu diketahui bahwa untuk setiap data akan memiliki parameter optimal yang berbeda-beda. Kernel yang digunakan pada penelitian adalah RBF dan untuk nilai *C* dan *Gamma* digunakan *grid search*. Algoritma *grid search* membagi jangkauan parameter yang akan dioptimalkan ke dalam *grid* dan melintasi semua titik untuk mendapatkan parameter yang optimal (Yasi, Prahutama, & Utami, 2014). Algoritma *grid search* pada penelitian menggunakan 10-*fold cross validation* agar mendapatkan *range* yang optimal dilihat nilai rata-rata akurasi dari data *testing cross validation*. Nilai *C* dan *Gamma* yang masing-masing dicobakan adalah pada *range*

10^{-2} – 10^{4} (Huang, Lee, Lin, & Huang, 2007), namun *range* diperlebar menjadi 10^{-4} – 10^{4} untuk mendapatkan hasil yang optimal. *Range* yang optimal dilihat dari rata-rata akurasi yang tinggi. Parameter optimal yang sudah didapatkan kemudian digunakan untuk memprediksi klasifikasi data menggunakan metode AdaBoost-SVM. Kombinasi dari nilai *C* dan *Gamma* yang optimal akan dibahas pada masing-masing sub-bab yaitu pemilihan variabel berdasarkan pemilihan fitur 5%, 10%, 25%, dan 35% variabel.

4.2.2 AdaBoost-SVM dengan 5% Variabel Prediktor Terpenting

Klasifikasi senyawa pelindung sel normal pada *Radioprotector* menggunakan metode AdaBoost-SVM dilakukan *grid search* untuk memperoleh parameter yang membentuk model optimal. Dalam mencari parameter SVM yang optimal, metode *grid search* yang dicobakan adalah pada klasifikasi SVM tunggal, karena pada metode AdaBoost-SVM, *base learner* yang digunakan adalah SVM. Apabila pada SVM tunggal yang dibuat adalah parameter yang optimal dan menghasilkan akurasi yang optimal, maka pada AdaBoost-SVM juga akan menghasilkan akurasi yang optimal. *Range* parameter *C* dan *Gamma* yang digunakan adalah 10^{-4} – 10^{4}. *Range* parameter yang terpilih merupakan rata-rata kinerja terbaik dari setiap kombinasi *range* parameter.

Tabel 4.29 Kombinasi *Range* Parameter untuk *Grid search* SVM pada Data 5% Variabel Terpenting

C	***GAMMA***	**Rata-rata Akurasi Total**
10^{-4} - 10^{-1}	10^{-4} - 10^{0}	0.5508
	10^{1} - 10^{4}	0.5122
10^{0} - 10^{4}	**10^{-4} - 10^{0}**	**0.6783**
	10^{1} - 10^{4}	0.5217

Angka bercetak tebal merupakan range parameter akurasi optimal.

Berdasarkan Tabel 4.29, dapat disimpulkan bahwa nilai parameter *C* dan *Gamma* yang optimal terdapat pada *range* 10^{0} - 10^{4} dan 10^{-4} - 10^{0}. Hal ini ditunjukkan dari rata-rata nilai akurasi yang diperoleh pada *range* tersebut merupakan rata-rata nilai

akurasi yang paling tinggi dibandingkan dengan rata-rata nilai akurasi pada *range* yang lain yaitu sebesar 0.6783.

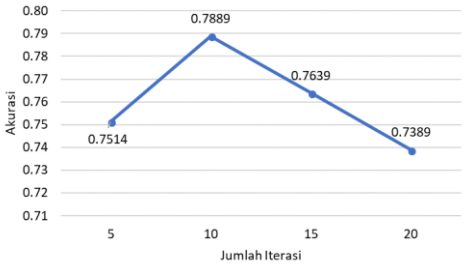
Langkah selanjutnya setelah mendapatkan kombinasi *range* yang optimal dari parameter *C* dan *Gamma* adalah melakukan prediksi klasifikasi menggunakan metode AdaBoost-SVM. Parameter pada AdaBoost-SVM adalah iterasi, sehingga dilakukan beberapa percobaan jumlah iterasi berbeda yaitu 5, 10, 15, dan 20. Akurasi pada penelitian diperoleh dengan metode validasi 10-*fold Cross-validation* pada data. Tabel 4.30 menunjukkan hasil dari klasifikasi AdaBoost-SVM pada 5% variabel prediktor.

Tabel 4.30 Hasil rata-rata akurasi Optimal AdaBoost-SVM dengan 5% Variabel Terpenting

Parameter SVM		**Jumlah**	**Rata-rata**
C	***Gamma***	**Iterasi**	**Akurasi Total**
1	0.1	5	0.7514
1	0.1	10	**0.7889**
10	0.01	15	0.7639
1000	0.001	20	0.7389

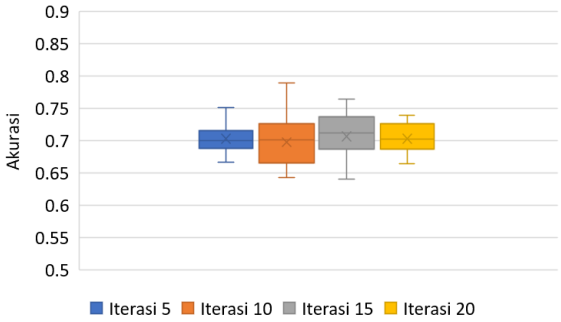
Angka bercetak tebal merupakan akurasi optimal.

Hasil analisis pada Tabel 4.30 menunjukkan hasil klasifikasi dengan akurasi yang optimal dari setiap jumlah iterasi yang dicobakan. Akurasi terendah terdapat pada jumlah iterasi 20 yaitu 0.7389 dengan nilai parameter *C* dan *Gamma* adalah 1000 dan 0.001, sedangkan akurasi tertinggi diperoleh pada jumlah iterasi 10 dengan akurasi 0.7889 dengan nilai parameter C dan *Gamma* yang optimal adalah 1 dan 0.1. Secara grafik hasil rata-rata akurasi lebih jelas dapat dilihat pada Gambar 4.6.



Gambar 4.6 Hasil rata-rata akurasi dari Klasifikasi AdaBoost-SVM dengan 5% Variabel Terpenting

Hasil rata-rata akurasi tertinggi pada data 5% variabel terpenting adalah 0.7889 dengan jumlah iterasi 10 dan parameter *C* dan *Gamma* 1 dan 0.1. Berikut merupakan *Box plot* dari akurasi setiap jenis jumlah iterasi dengan nilai parameter C dan *Gamma* yang optimal yaitu pada *range* 10^0 - 10^4 dan 10^{-4} - 10^0.



Gambar 4.7 *Box Plot* Nilai Akurasi Seluruh Model pada 5% Variabel Terpenting

Gambar 4.7 menyajikan *Box Plot* dari akurasi setiap jenis jumlah iterasi dengan nilai parameter C dan *Gamma* yang optimal yaitu pada *range* 10^0 - 10^4 dan 10^{-4} - 10^0. Nilai-nilai akurasi yang dihasilkan pada jumlah iterasi 10 yang ditunjukkan warna jingga pada gambar cenderung lebih tinggi dibandingkan dengan jumlah iterasi yang lain, namun variasi yang dihasilkan lebih besar. Variasi dari nilai akurasi terkecil ditunjukkan pada warna biru yaitu jumlah iterasi 5, dimana variasi kecil menunjukkan nilai akurasi yang dihasilkan pada setiap nilai *C* dan *Gamma* yang berbeda menghasilkan akurasi yang tidak jauh berbeda satu sama lain.

Tabel 4.31 Akurasi Setiap *Fold* Data *Testing* Pada 5% Variabel Terpenting Jumlah Iterasi 10

Data	***Fold* ke-**	**Akurasi *Testing***
5%	1	0.5556
	2	0.6667
	3	0.8889
	4	0.7778
	5	0.6250
	6	0.8750
	7	0.7500

Tabel 4.32 Akurasi Setiap *Fold* Data *Testing* Pada 5% Variabel Terpenting Jumlah Iterasi 10 (Lanjutan)

Data	*Fold* ke-	Akurasi *Testing*
5%	8	1.0000
	9	0.8750
	10	0.8750
Rata-rata		**0.7889**

Pada Tabel 4.31 menunjukkan akurasi pada masing-masing *fold* pada akurasi tertinggi yaitu pada jumlah iterasi 10. Akurasi data *testing* tertinggi terdapat pada *fold* ke-8 yaitu akurasi 1 dan terkecil pada *fold* ke-1 yaitu 0.5556.

4.2.3 AdaBoost-SVM dengan 10% Variabel Prediktor Terpenting

Tahapan yang dilakukan dalam prediksi klasifikasi mengguanakan metode AdaBoost-SVM dengan 10% variabel terpenting sama seperti pada subbab 4.2.2. Pertama dilakukan *grid search* pada SVM tunggal pada *range* parameter *C* dan *Gamma* yaitu $10^{-4} – 10^{4}$. *Range* parameter yang terpilih merupakan rata-rata kinerja terbaik dari setiap kombinasi *range* parameter.

Tabel 4.33 Kombinasi *Range* Parameter untuk *Grid search* SVM pada Data 10% Variabel Terpenting

C	*GAMMA*	Rata-rata Akurasi Total
10^{-4} - 10^{-1}	10^{-4} - 10^{0}	0.5453
	10^{1} - 10^{4}	0.5028
10^{0} - 10^{4}	**10^{-4} - 10^{0}**	**0.6969**
	10^{1} - 10^{4}	0.5156

Angka bercetak tebal merupakan range parameter akurasi optimal.

Berdasarkan Tabel 4.32, dapat disimpulkan bahwa nilai parameter C dan *Gamma* yang optimal terdapat pada *range* 10^{0} - 10^{4} dan 10^{-4} - 10^{0}. Hal ini ditunjukkan dari rata-rata nilai akurasi yang diperoleh pada *range* tersebut merupakan rata-rata nilai akurasi yang paling tinggi dibandingkan dengan rata-rata nilai akurasi pada *range* yang lain yaitu sebesar 0.6969. Hasil *range* parameter tersebut digunakan untuk klasifikasi AdaBoost-SVM.

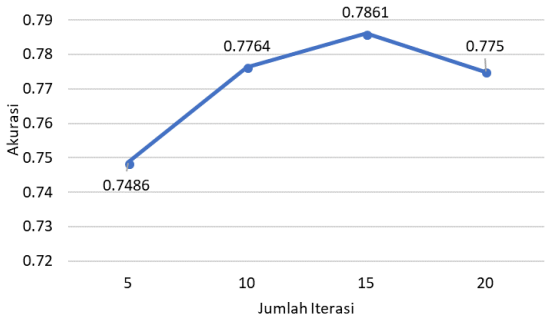
Hasil klsifikasi AdaBoost-SVM dengan beberapa jumlah iterasi yaitu 5, 10, 15, dan 20 ditunjukkan pada Tabel 4.33 berikut.

Tabel 4.34 Hasil rata-rata akurasi Optimal AdaBoost-SVM dengan 10% Variabel Terpenting

Parameter SVM		Jumlah Iterasi	Rata-rata Akurasi Total
C	***Gamma***		
100	0.001	5	0.7486
10	0.01	10	0.7764
10	0.001	15	**0.7861**
100	0.0001	20	0.775

Angka bercetak tebal merupakan akurasi optimal.

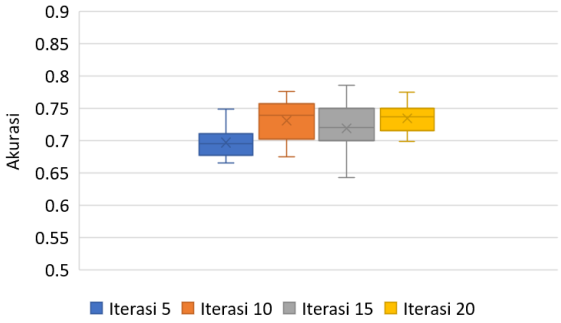
Hasil rata-rata akurasi yang ditunjukkan pada Tabel 4.33 menunjukkan hasil klasifikasi dengan akurasi yang optimal dari setiap jumlah iterasi yang dicobakan. Akurasi terendah terdapat pada jumlah iterasi 5 yaitu 0.7486 dengan nilai *C* dan *Gamma* adalah 100 dan 0.001, sedangkan akurasi tertinggi diperoleh pada jumlah iterasi 10 dengan akurasi 0.7861 dengan nilai parameter C dan *Gamma* yang optimal adalah 10 dan 0.001.



Gambar 4.8 Hasil rata-rata akurasi dari Klasifikasi AdaBoost-SVM dengan 10% Variabel Terpenting

Perbedaan Hasil rata-rata akurasi lebih jelas dapat dilihat pada Gambar 4.8. Hasil rata-rata akurasi tertinggi pada data dengan 10% variabel terpenting adalah 0.7861 dengan jumlah iterasi 10 dan parameter *C* dan *Gamma* 10 dan 0.001. Berikut merupakan *Box plot* dari akurasi setiap jenis jumlah iterasi dengan nilai parameter

C dan *Gamma* yang optimal yaitu pada *range* 10^0 - 10^4 dan 10^{-4} - 10^0.



Gambar 4.9 *Box Plot* Nilai Akurasi Seluruh Model pada 10% Variabel Terpenting

Box Plot pada Gambar 4.9 menunjukkan dari akurasi setiap jenis jumlah iterasi dengan nilai parameter C dan *Gamma* yang optimal yaitu pada *range* 10^0 - 10^4 dan 10^{-4} - 10^0. Dari gambar terlihat bahwa akurasi tertinggi dan terendah terletak pada jumlah iterasi 15 yang ditunjukkan dengan warna abu-abu. Variasi nilai akurasi terbesar terdapat pada jumlah iterasi 10 ditunjukkan pada warna jingga yaitu dibuktikan dengan luasan box yang lebih panjang dibanding yang lain, sedankan variasi terkecil terdapat pada jumlah iterasi 5 yaitu pada warna biru. Secara keseluruhan menunjukkan semakin banyak iterasi pada AdaBoost-SVM, variasi akurasi yang dihasilkan semakin kecil.

Tabel 4.35 Akurasi Setiap *Fold* Data *Testing* Pada 10% Variabel Terpenting Jumlah Iterasi 15

Data	***Fold***	**Akurasi *Testing***
10%	1	0.5556
	2	0.8889
	3	0.7778
	4	0.8889
	5	0.3750
	6	0.8750
	7	0.8750
	8	1.0000
	9	0.7500
	10	0.8750
Rata-rata		**0.7861**

Tabel 4.34 merupakan nilai akurasi pada masing-masing *fold* pada akurasi tertinggi yaitu pada jumlah iterasi 15. Akurasi data *testing* tertinggi terdapat pada *fold* ke-8 yaitu akurasi 1 dan terkecil pada *fold* ke-5 yaitu 0.375.

4.2.4 AdaBoost-SVM dengan 25% Variabel Prediktor Terpenting

Data pada klasifikasi AdaBoost-SVM dengan 25% variabel terpenting adalah variabel prediktor yang digunakan sebanyak 55 variabel dengan menggunakan *cross validation* dalam pembagian data *testing* dan *training*. Tahapan yang dilakukan dalam prediksi klasifikasi mengguanakan metode AdaBoost-SVM dengan 25% variabel terpenting sama seperti pada subbab 4.2.2. Pertama dilakukan *grid search* pada SVM tunggal pada *range* parameter *C* dan *Gamma* yaitu $10^{-4} – 10^{4}$ dengan hasil pada Tabel 4.35.

Tabel 4.36 Kombinasi *Range* Parameter untuk *Grid search* SVM pada Data 25% Variabel Terpenting

C	***GAMMA***	**Rata-rata Akurasi Total**
10^{-4} - 10^{-1}	10^{-4} - 10^{0}	0.5442
	10^{1} - 10^{4}	0.5028
10^{0} - 10^{4}	**10^{-4} - 10^{0}**	**0.6652**
	10^{1} - 10^{4}	0.4840

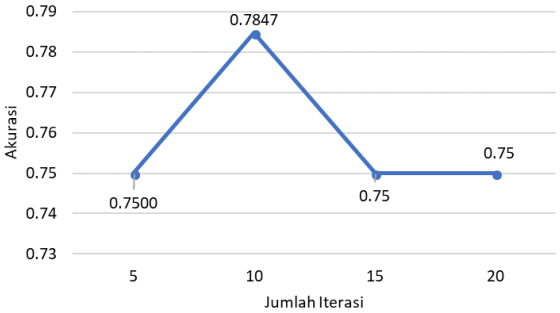
Angka bercetak tebal merupakan range parameter akurasi optimal.

Nilai parameter C dan *Gamma* yang optimal pada data 25% variabel terpenting terdapat pada *range* 10^{0} - 10^{4} dan 10^{-4} - 10^{0} sesuai pada Tabel 4.35. Hal ini ditunjukkan dari rata-rata nilai akurasi yang diperoleh pada *range* tersebut merupakan rata-rata nilai akurasi yang paling tinggi dibandingkan dengan rata-rata nilai akurasi pada *range* yang lain yaitu sebesar 0.6652.

Tabel 4.37 Hasil rata-rata akurasi Optimal AdaBoost-SVM dengan 25% Variabel Terpenting

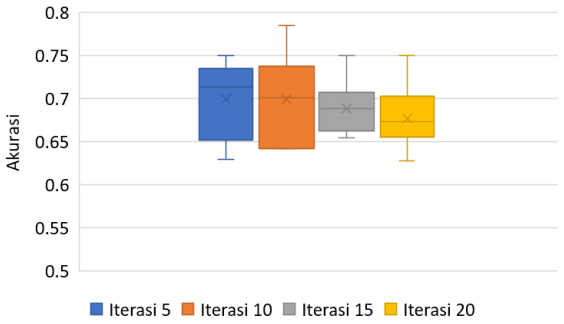
Parameter SVM		**Jumlah Iterasi**	**Rata-rata Akurasi Total**
C	***Gamma***		
1	0.1	5	0.75
10	0.001	10	**0.7847**
1000	0.0001	15	0.75
1	0.1	20	0.75

Tabel 4.36 menunjukkan hasil klsifikasi AdaBoost-SVM dengan beberapa jumlah iterasi yaitu 5, 10, 15, dan 20. Berdasarkan nilai akurasi pada Tabel 4.36, didapatkan akurasi yang optimal dari setiap jumlah iterasi yang dicobakan. Akurasi terendah terdapat pada jumlah iterasi 5, 15, dan 20 yaitu 0.75, sedangkan akurasi tertinggi diperoleh pada jumlah iterasi 10 dengan akurasi 0.7847 dengan nilai parameter C dan *Gamma* yang optimal adalah 10 dan 0.001. Secara grafik hasil rata-rata akurasi lebih jelas dapat dilihat pada Gambar 4.10.



Gambar 4.10 Hasil rata-rata akurasi dari Klasifikasi AdaBoost-SVM dengan 25% Variabel Terpenting

Berikut merupakan *Box plot* dari akurasi setiap jenis jumlah iterasi dengan nilai parameter C dan *Gamma* yang optimal yaitu pada *range* 10^0 - 10^4 dan 10^{-4} - 10^0.



Gambar 4.11 *Box Plot* Nilai Akurasi Seluruh Model pada 25% Variabel Terpenting

Gambar 4.11 menyajikan *Box Plot* dari akurasi yang dihasilkan pada setiap percobaan dengan parameter yang berbeda pada keempat jenis iterasi. Nilai-nilai akurasi yang dihasilkan pada jumlah iterasi 10 yang ditunjukkan warna jingga pada gambar cenderung lebih tinggi dibandingkan dengan jumlah iterasi yang lain, namun variasi dari nilai akurasi yang dihasilkan juga lebih besar yaitu ditunjukkan pada ukuran batang yang lebih panjang dibandingkan dengan yang lain. Variasi nilai akurasi terkecil ditunjukkan pada warna abu-abu yaitu jumlah iterasi 15, hal ini menunjukkan nilai akurasi yang dihasilkan pada setiap nilai *C* dan *Gamma* yang berbeda menghasilkan akurasi yang tidak jauh berbeda satu sama lain. Secara visual, variasi akurasi yang dihasilkan menunjukkan semakin banyak iterasi pada AdaBoost-SVM, variasi akurasi yang dihasilkan semakin kecil.

Nilai akurasi pada Tabel 4.18 merupakan nilai akurasi pada masing-masing *fold* pada akurasi tertinggi yaitu pada jumlah iterasi 10. Akurasi data *testing* tertinggi terdapat pada *fold* ke-8 yaitu akurasi 1 dan terkecil pada *fold* ke-6 dan ke-9 yaitu 0.625.

Tabel 4.38 Akurasi Setiap *Fold* Data *Testing* Pada 25% Variabel Terpenting Jumlah Iterasi 10

Data	***Fold***	**Akurasi *Testing***
25%	1	0.6667
	2	0.8889
	3	0.7778
	4	0.8889
	5	0.7500
	6	0.6250
	7	0.8750
	8	1.0000
	9	0.6250
	10	0.7500
Rata-rata		**0.7847**

4.2.5 AdaBoost-SVM dengan 35% Variabel Prediktor Terpenting

Tahapan yang dilakukan dalam prediksi klasifikasi mengguanakan metode AdaBoost-SVM dengan 35% variabel terpenting sama seperti pada subbab 4.2.2. Jumlah variabel yang

digunakan adalah 76 variabel prediktor. Langkah pertama yang dilakukan adalah mendapatkan *range* parameter *C* dan *Gamma* yang optimal. *Range* parameter *C* dan *Gamma* yang digunakan yaitu $10^{-4} – 10^{4}$.

Berdasarkan Tabel 4.38, nilai parameter C dan *Gamma* yang optimal pada data 35% variabel terpenting terdapat pada *range* 10^{0} - 10^{4} dan 10^{-4} - 10^{0} dengan akurasi 0.6378.

Tabel 4.39 Kombinasi *Range* Parameter untuk *Grid search* SVM pada Data 25% Variabel Terpenting

C	***GAMMA***	**Rata-rata Akurasi Total**
10^{-4} - 10^{-1}	10^{-4} - 10^{0}	0.5364
	10^{1} - 10^{4}	0.5090
10^{0} - 10^{4}	**10^{-4} - 10^{0}**	**0.6378**
	10^{1} - 10^{4}	0.4922

Angka bercetak tebal merupakan range parameter akurasi optimal.

Setelah didapatkan *range* parameter yang optimal, selanjutnya dilakukan prediksi klasifikasi menggunakan metode AdaBoost-SVM. Klasifikasi dilakukan dengan beberapa percobaan dengan menggunakan beberapa jumlah iterasi yaitu 5, 10, 15, dan 20. Akurasi pada penelitian diperoleh dengan metode validasi 10-*fold Cross-validation* pada data. Berikut ini merupakan hasil dari percobaan dengan akurasi yang optimal.

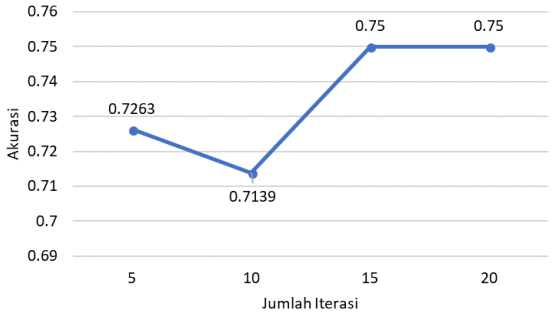
Tabel 4.40 Hasil rata-rata akurasi Optimal AdaBoost-SVM dengan 35% Variabel Terpenting

Parameter SVM		**Jumlah**	**Akurasi**
C	***Gamma***	**Iterasi**	**Total**
100	0.0001	5	0.7263
10000	0.0001	10	0.7139
100	0.0001	15	**0.75**
100	0.0001	20	0.7291

Angka bercetak tebal merupakan akurasi optimal.

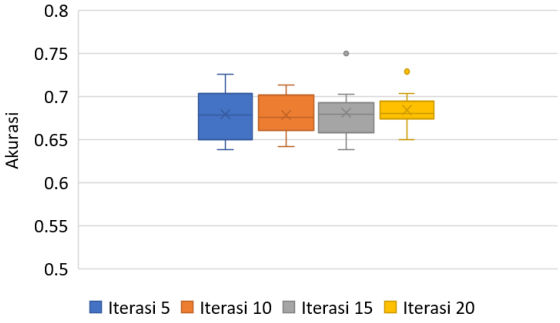
Hasil klasifikasi pada Tabel 4.39 merupakan akurasi yang optimal dari setiap jumlah iterasi yang dicobakan. Akurasi terendah terdapat pada jumlah iterasi 10 yaitu 0.7139, sedangkan akurasi tertinggi diperoleh pada jumlah iterasi 15 dengan akurasi

0.75 dengan nilai parameter C dan *Gamma* yang optimal adalah 100 dan 0.0001. Secara gambar hasil rata-rata akurasi lebih jelas dapat dilihat pada Gambar 4.12.



Gambar 4.12 Hasil rata-rata akurasi dari Klasifikasi AdaBoost-SVM dengan 35% Variabel Terpenting

Hasil rata-rata akurasi tertinggi pada data dengan 35% variabel terpenting adalah 0.75 dengan jumlah iterasi 15 dan parameter *C* dan *Gamma* 10 dan 0.0001. Berikut merupakan *Box plot* dari akurasi setiap jenis jumlah iterasi dengan nilai parameter C dan *Gamma* yang optimal yaitu pada *range* 10^{0} - 10^{4} dan 10^{-4} - 10^{0}



Gambar 4.13 *Box Plot* Nilai Akurasi Seluruh Model pada 35% Variabel Terpenting

Box plot yang ditunjukkan pada Gambar 4.13 merupakan nilai-nilai akurasi yang dihasilkan untuk keempat jumlah iterasi. Terlihat bahwa variasi akurasi pada jumlah iterasi 5, 10, dan 15

memiliki kecenderungan menghasilkan variasi akurasi yang semakin kecil apabila iterasi semakin besar. Variasi terkecil terdapat pada jumlah iterasi 20 yang ditunjukkan warna kuning, sedangkan variasi terbesar ada pada warna biru yaitu dengan jumlah iterasi 5.

Tabel 4.41 Akurasi Setiap *Fold* Data *Testing* Pada 35% Variabel Terpenting Jumlah Iterasi 15

Data	*Fold*	Akurasi *Testing*
35%	1	0.5556
	2	0.8889
	3	0.7778
	4	0.7778
	5	0.6250
	6	0.7500
	7	0.7500
	8	0.8750
	9	0.7500
	10	0.7500
Rata-rata		**0.75**

Berdasarkan Tabel 4.41, nilai akurasi pada masing-masing *fold* pada akurasi tertinggi yaitu pada jumlah iterasi 10. Akurasi data *testing* tertinggi terdapat pada *fold* ke-2 yaitu akurasi 0.8889 dan terkecil pada *fold* ke-1 yaitu 0.5556.

4.3 Pemilihan Metode Terbaik

Tabel 4.41 menunjukkan perbandingan antara klasifikasi dengan metode LORENS dan AdaBoost-SVM menggunakan perbandingan nilai akurasi. Metode LORENS menggunakan seluruh variabel prediktor dalam memprediksi, namun pada AdaBoost-SVM menggunakan *feature selection* dimana dibagi menjadi empat yaitu berdasarkan variabel terpenting 5%, 10%, 25%, dan 35%. Berikut adalah hasil akhir dari nilai akurasi tertinggi untuk setiap metode ditunjukkan pada Tabel 4.42.

Tabel 4.42 Perbandingan Akurasi dari Klasifikasi Menggunakan LORENS dan AdaBoost-SVM

Metode	Persentase Variabel	Akurasi	Parameter		
			C	***Gamma***	**Iterasi**
LORENS	100%	0.6904	-	-	-
AdaBoost-SVM	5%	**0.7889**	1	0.1	10
	10%	0.7861	10	0.001	15
	25%	0.7847	10	0.001	10
	35%	0.75	100	0.0001	15

Berdasarkan hasil analisis yang telah dilakukan dengan akurasi sebagai acuan, dilihat dari nilai akurasi tertinggi yaitu sebesar 0.7889 adalah metode AdaBoost-SVM dengan variabel prediktor yang digunakan adalah 5% variabel terpenting yaitu dengan jumlah 11 variabel prediktor.

(Halaman ini sengaja dikosongkan)

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang diperoleh dari hasil analisis dan pembahasan klasifikasi senyawa pelindung sel normal pada *Radioprotector* berdasarkan tingkat toksisitas menggunakan metode LORENS dan AdaBoost-SVM adalah sebagai berikut.

1. Analisis LORENS pada senyawa pelindung sel normal pada *Radioprotector* berdasarkan tingkat toksisitas dengan prosedur evaluasi *cross validation* menghasilkan partisi terbaik sebesar 30 dengan *threshold* 0.5. Ukuran kebaikan model yang digunakan adalah akurasi yaitu sebesar 0.6904%.
2. Klasifikasi senyawa pelindung sel normal pada *Radioprotector* berdasarkan tingkat toksisitas menggunakan AdaBoost-SVM dengan beberapa jenis data yaitu berdasarkan seleksi variabel *Mean Decrease Gini* pada variabel prediktor dengan peringkat terpenting 5%, 10%, 25%, dan 35%. Pada data 5% variabel terpenting didapatkan Hasil rata-rata akurasi 0.7889 dengan parameter C 1 dan *Gamma* 0.1 dengan jumlah iterasi 10. Data 10% variabel terpenting mendapatkan akurasi 0.7861 dengan parameter C 10 dan *Gamma* 0.001 dengan jumlah iterasi 15. Pada data 15% variabel terpenting dihasilkan akurasi sebesar 0.7847 dengan parameter C 10 dan *Gamma* 0.001 dengan jumlah iterasi 10, sedangkan pada data 35% variabel terpenting didapatkan akurasi sebesar 0.75 dengan parameter C 100 dan *Gamma* 0.0001 dengan jumlah iterasi 15.
3. Metode klasifikasi yang menghasilkan nilai akurasi terbesar yaitu metode klasifikasi AdaBoost-SVM dengan akurasi 0.7889 menggunakan variabel berjumlah 11 peringkat tertinggi berdasarkan *Mean Decrease Gini*.

5.2 Saran

Berdasarkan penelitian yang telah dilakukan, saran yang diberikan bagi dunia kedokteran adalah metode AdaBoost-SVM

dapat digunakan untuk klasifikasi senyawa pelindung sel normal pada *Radioprotector* berdasarkan tingkat toksisitas karena pada penelitian ini metode tersebut menghasilkan nilai akurasi paling besar dibandingkan dengan metode LORENS.

DAFTAR PUSTAKA

Ariyasu, S., Sawa, A., Morita, A., Hanaya, K., Hoshi, M., Takahashi, I., . . . Aoki, S. (2014). Design and Synthesis of 8-Hydroxyquinoline-Based Radioprotective Agents. *Bioorganic and Medicinal Chemistry, 22*(15), 3891-3905.

Bekkar, M., Djemaa, H. K., & Alitouche, T. A. (2013). Evaluation Measures for Models Assesment Over Imbalanced Data Sets. *Journal of Information Engineering and Applications, 3*(10).

Cortes, & Vapnik. (1995). Support Vector Networks. *Machine Learning 1 September pp*, 273-297.

Deeptchi CH, N., Kumar A, V., Rameshbabu, & Indirapriyadarshini, U. (2011, July 18). Role of Tumor Suppessor Protein p53 in Apoptosis and Cancer Therapy. *Journal of Cancer Science & Therapy*. doi:10.4172/1948-5956.S17-001

Dragomir, A., & Bezerianos, A. (2006). Improving Gene Expression Sample Classification Using Support Vector Machine *Ensemble*s Aggregated by *Boosting*. *CANCER GENOMICS & PROTEOMICS*, 63-70.

Durham, W. F. (1975). *Toxicity in N.I. Saz (ed):* Dan*gerous Properties of Industrial.* New York: Van Nostrand Reinhold Co.

Gunn, S. (1998). *Support Vector Machine for Classification and Regression.* Southampton: University of Southampton.

Hastie, T., Tibshirani, R., & Friendman, J. (2008). *The Elements of Statistical Learning : Data Mining, Inference, and Prediction* (2nd ed.). New York (US): Springer.

Hosmer, D. W., & Lemeshow, S. (2000). *Applied Logistic Regression* (2nd ed.). New York: John Wiley & Sons.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2010). *A Practical Guide to Support Vector Classification.* Taiwan: Department of Computer Science National Taiwan University.

Huang, C. M., Lee, Y. J., Lin, D. K., & Huang, S. Y. (2007). Model Selection for Support Vector Machines via Uniform Design. *Computational Statistics & Data Analysis*, 335-346.

Huang, M.-W., Chen, C.-W., Lin, W.-C., Ke, S.-W., & Tsai, C.-F. (2017). SVM and SVM *Ensemble*s in Breast Cancer. *Plus ONE*.

Kartika, U. (2013). Penderita Kanker di Indonesia Meningkat. Dipetik Februari 3, 2018, dari http://www.health.kompas.com

Kim, H.-C., Pang, S., Je, H.-M., & Bang, S.-Y. (2002). Support Vector Machine *Ensemble* with Bagging. *Lecture Notes in Computer Science* (pp. 397-408). Springer, Berlin, Heidelberg.

Kimura, M., Aoki, S., & Ohwada, H. (2017). Predicting Radiation Protection and Toxicity of p53 Targeting Radioprotectors using Machine Learning. *Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, 1-4. doi:0.1109/CIBCB.2017.8058540

Kuswanto, H., & Wedhana, R. W. (2018). Logistic Regression *Ensemble*s to classify Alzheimer gene expression. *Internetworking Indonesia Journal 10(01)*.

Kuswanto, H., Asfihani, A., Sarumaha, Y., & Ohwada, H. (2015). Logistic Regression *Ensemble*s for Predicting Customer Defection with Very Large Sample Size. *Procedia Computer Science 72 86-93*.

Lee, K., Ahn, H., Moon, H., Kodell, R., & Chen, J. (2013). Multinomial Logistic Regression *Ensemble*s. *Biopharm Stat*.

Li, X., Wang, L., & Sung, E. (2006). Adaboost with SVM-based component *classifier*. *Engineering Applications of Artificial*, 785-795.

Lim, N., Ahn, H., Moon, H., & Chen, J. (2010). Classification of high-dimensional data with *ensemble* of logistic regression

models. *Journal of Biopharmaceutical Statistics 20*, 160-171.

Loomis, T. (1978). *Toksikologi Dasar*. (I. A. Donatus, Trans.) Semarang: IKIP Semarang Press.

Marco, S., & Zuccolotto, P. (2006). Variable Selection Using Random Forests. *Classification and Data Analysis Group (CLADAG)* (pp. 263-270). Berlin Heidelberg: Springer

Matsumoto, A., Aoki, S., & Ohwada, H. (2016). Comparison of Random Forest and SVM for Raw Data in Drug Discovery: Prediction of Radiation Protection and Toxicity Case Study. *International Journal of Machine Learning and Computing, 6*(2), 145-148.

Meiyanto, E., Supardjan, D. M., & Agustina, D. (2006). Efek Antiproliferatif Pentagamavunon. *Jurnal Kedokteran Yarsi, 14*, 11-15.

Menze, B., Kelm, B., Masuch, R., Himmelreich, U., Bachert, P., Petrich, W., & Hamprecht, F. (2009). A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data. *BMC Bioinformatics*, 2. doi:10.1186/1471-2105-10-213

Okun, O. (2011). *Bioinformatics: Algorithmic Classification and Implementation.* United States of America: IGI Global.

Pal, M., & Mather, P. M. (2005). Support Vector Machines for Classification in Remote Sensing. *International Journal of Remote Sensing*, 1007-1011.

Pappu, V., & Pardalos, P. M. (2014). High Dimensional Data Classification. *Clusters, Orders, and Trees: Methods and Applications: In Honor of Boris Mirkin's 70th Birthday*, 120-154.

Pratiwi, E. (2016). *Analisis Dosis Radiasi Pada Radioterapi Eksternal Linear Accelerator (Linac) Carcinoma Cervix Dengan Variasi Sudut Wedge Di Rumah Sakit Umum Pusat Dr. Sardjito Yogyakarta*. Universitas Gadjah Mada,

Departemen Teknik Nuklir dan Teknik Fisika. Yogyakarta: Skripsi.

Rusyidi, I. (2009). *Deteksi Dini* dan *Pencegahan Kanker Pada Wanita*. Jakarta: Sagung Seto.

Santosa, B. (2007). *Data Mining Teknik Pemanfaatan Data untuk Keperluan Bisnis.* Yogyakarta: Graha Ilmu.

Schapire, R. E. (1999). A Brief Introduction to *Boosting*. *Proceedings of the Sixteenth International Joint*.

Sukardiman, R., R. Abdul, & P.N. Fatma. (2004). *Uji Praskrining Aktivitas Antikanker Ekstrak Eter* dan *Ekstrak Metanol Marchantia planiloba Steph dengan Metode Uji Kematian Larva U*dang dan *Profil Densitometri Ekstrak Aktif.* Surabaya: Majalah Farmasi Airlangga.

Syam, S., Dewang, S., & Abdullah, B. (2014). *Analisis Dosis Radiasi Pada Paru-paru Untuk Pasien Kanker Payudara.* Universitas Hasanuddin, Jurusan Fisika, Makassar.

Valentini, G., Muselli, M., & Ruffino, F. (2014). Cancer recognition with bagged *ensemble*s of support vector machines. *Neurocomputing*, 461-466.

Vapnik, V. N. (1999). *The Nature of Statictical Learning Theory 2nd Edition.* Springer-Verlag: New York Berlin Heidelberg.

Vapnik, V., & Chapelle, O. (2000). Bounds on *Error* Expectation for Support Vector Machine. *Neural Computation*, 2013-2036.

Wahyuni, T. (2015). Hubungan Antara Frekuensi Kemoterapi Dengan Kualitas Hidup Perempuan Dengan Kanker Payudara Yang Menjalani Kemoterapi Di Ruang Kemoterapi Rsud. A.M Parikesit Tenggarong. *Jurnal Ilmu Kesehatan, 3*, 1-3.

Wirasuta, I. A., & Niruri, R. (2006). *Buku Ajar : TOKSIKOLOGI UMUM.* Universitas UDAYANA: Jurusan Farmasi .

Witten, H. I., Frank, E., & Hall, M. A. (2011). *Data Mining : Pactical Machine Learning Tools and Techniques* (3rd ed.). Burlington: Morgan Kaufmann.

Yan, X., & Su, X.G. (2009). *Linear Regression Analysis : Theory and Computing* . Singapore: World Scientific.

Zhou, Z. (2012). *Ensemble Methods: Foundations and Algorithms.* Boca Raton (US): CRC Press.

(Halaman ini sengaja dikosongkan)

LAMPIRAN

Lampiran 1 Data Penelitian

No	Y	X1	X2	X3	X4	X5	…	X214	X215	X216	X217
1	0	20	0	19	0	0	…	137.588	648.929	586.566	351.91
2	0	11.4	0	11	0	0	…	226.45	487.94	455.418	224.66
3	1	7.9	0	10	0	0	…	163.621	394.179	368.299	170.81
4	1	8.3	0	11	0	0	…	144.222	429.132	397.033	188.3
5	1	20	0	11	0	0	…	22.785	361.33	331.297	140.62
6	0	20	0	12	0	0	…	107.026	454.484	417.73	202.36
7	0	10.1	0	13	0	0	…	180.347	549.419	505.647	261.02
8	0	20	0	13	0	0	…	236.294	573.78	530.977	288.11
9	0	11.5	0	12	0	0	…	207.014	525.909	487.739	244.9
10	0	20	0	10	1	0	…	23.232	352.614	334.742	139.6
11	0	20	0	11	1	0	…	16.821	391.018	367.106	154.69
…	…	…	…	…	…	…	…	…	…	…	…
73	0	11.4	0	29	0	0	…	41.414	836.824	730.724	423.94
74	0	20	1	13	0	0	…	144.684	541.087	509.638	266.16
75	0	20	1	18	0	0	…	117.589	623.372	586.612	319.33
76	1	7.1	0	9	0	0	…	130.002	331.421	314.56	125.19
77	0	20	0	27	0	0	…	260.317	868.205	808.732	428.4
78	1	4.8	0	9	1	0	…	44.197	325.554	312.936	126.22
79	1	20	0	16	0	0	…	177.091	642.541	589.412	349.17
80	0	20	0	11	1	0	…	82.935	442.921	418.891	204.42
81	0	20	0	14	0	0	…	218.11	613.437	565.538	315.21
82	1	20	0	12	1	0	…	44.392	443.131	416.288	193.1
83	0	20	0	15	0	0	…	191.184	615.137	566.256	306.98
84	0	9.8	0	20	0	0	…	214.178	693.091	622.967	363.23

Lampiran 2 Syntax Pembagian *fold*

```
#FOLD1
fold1=fold[[1]]
train1=dataa[-fold[[1]],]
test1=dataa[fold[[1]],]
#FOLD2
fold2=fold[[2]]
train2=dataa[-fold[[2]],]
test2=dataa[fold[[2]],]
#FOLD3
fold3=fold[[3]]
train3=dataa[-fold[[3]],]
test3=dataa[fold[[3]],]
#FOLD4
fold4=fold[[4]]
train4=dataa[-fold[[4]],]
test4=dataa[fold[[4]],]
#FOLD5
fold5=fold[[5]]
train5=dataa[-fold[[5]],]
test5=dataa[fold[[5]],]
#FOLD6
fold6=fold[[6]]
train6=dataa[-fold[[6]],]
test6=dataa[fold[[6]],]
#FOLD7
fold7=fold[[7]]
train7=dataa[-fold[[7]],]
test7=dataa[fold[[7]],]
#FOLD8
fold8=fold[[8]]
train8=dataa[-fold[[8]],]
test8=dataa[fold[[8]],]
#FOLD9
fold9=fold[[9]]
train9=dataa[-fold[[9]],]
test9=dataa[fold[[9]],]
#FOLD10
fold10=fold[[10]]
train10=dataa[-fold[[10]],]
test10=dataa[fold[[10]],]
```

Lampiran 3 Syntax *Feature Selection Mean Decrease Gini*

```
#import library
library(caret)
library(randomForest)
#import data
data <- read.csv("E: /DATA TA/DATA_TA.csv",sep=";")
data$Y <- as.factor(data$Y)
str(data)
#parameter tuning random forest
customRF <- list(type = "Classification", library =
"randomForest", loop = NULL)
customRF$parameters <- data.frame(parameter = c("mtry",
"ntree"), class = rep("numeric", 2), label = c("mtry",
"ntree"))
customRF$grid <- function(x, y, len = NULL, search = "grid")
{}
customRF$fit <- function(x, y, wts, param, lev, last, weights,
classProbs, ...) {
randomForest(x, y, mtry = param$mtry, ntree=param$ntree, ...)}
customRF$predict <- function(modelFit, newdata, preProc =
NULL, submodels = NULL)
predict(modelFit, newdata)
customRF$prob <- function(modelFit, newdata, preProc = NULL,
submodels = NULL)
predict(modelFit, newdata, type = "prob")
customRF$sort <- function(x) x[order(x[,1]),]
customRF$levels <- function(x) x$classes
control <- trainControl(method="repeatedcv", number=10,
repeats=10)
tunegrid <- expand.grid(.mtry=c(1:10), .ntree=c(100, 200,
            300, 400, 500, 600,700,800,900,1000))
custom <- train(Y~., data=data, method=customRF,
tuneGrid=tunegrid, trControl=control)
custom
summary(custom)
plot(custom)
#function var.share
var.share <- function(rf.obj, members) {
  count <- table(rf.obj$forest$xbestsplit)[-1]
  names(count) <- names(rf.obj$forest$ncat)
  share <- count[members] / sum(count[members])
  return(share)}
#function group.importance
group.importance <- function(rf.obj, groups) {
  var.imp <- as.matrix(sapply(groups, function(g) {
    sum(importance(rf.obj, 2)[g, ]*var.share(rf.obj, g))
  }))
  colnames(var.imp) <- "MeanDecreaseGini"
  return(var.imp)}
#mean decrease gini
rf.obj <- randomForest(Y ~., data=data, ntree=800, mtry=5)
```

```
groups=list(X1=
    c("X1"),
  X2=c("X2"),
  X3=c("X3"),
  X4=c("X4"),
  X5=c("X5"),
  X6=c("X6"),
  X7=c("X7"),
  X8=c("X8"),
  X9=c("X9"),
 X10=c("X10"),
 X11=c("X11"),
 X12=c("X12"),
 X13=c("X13"),
 X14=c("X14"),
 X15=c("X15"),
 X16=c("X16"),
 X17=c("X17"),
 X18=c("X18"),
 X19=c("X19"),
 X20=c("X20"),
 X21=c("X21"),
 X22=c("X22"),
 X23=c("X23"),
 X24=c("X24"),
 X25=c("X25"),
 X26=c("X26"),
 X27=c("X27"),
 X28=c("X28"),
 X29=c("X29"),
 X30=c("X30"),
 X31=c("X31"),
 X32=c("X32"),
 X33=c("X33"),
 X34=c("X34"),
 X35=c("X35"),
 X36=c("X36"),
 X37=c("X37"),
 X38=c("X38"),
 X39=c("X39"),
 X40=c("X40"),
 X41=c("X41"),
 X42=c("X42"),
 X43=c("X43"),
 X44=c("X44"),
 X45=c("X45"),
 X46=c("X46"),
 X47=c("X47"),
 X48=c("X48"),
 X49=c("X49"),
 X50=c("X50"),
  X51=c("X51"),
  X52=c("X52"),
  X53=c("X53"),
  X54=c("X54"),
  X55=c("X55"),
  X56=c("X56"),
  X57=c("X57"),
  X58=c("X58"),
  X59=c("X59"),
  X60=c("X60"),
  X61=c("X61"),
  X62=c("X62"),
  X63=c("X63"),
  X64=c("X64"),
  X65=c("X65"),
  X66=c("X66"),
  X67=c("X67"),
  X68=c("X68"),
  X69=c("X69"),
  X70=c("X70"),
  X71=c("X71"),
  X72=c("X72"),
  X73=c("X73"),
  X74=c("X74"),
  X75=c("X75"),
  X76=c("X76"),
  X77=c("X77"),
  X78=c("X78"),
  X79=c("X79"),
  X80=c("X80"),
  X81=c("X81"),
  X82=c("X82"),
  X83=c("X83"),
  X84=c("X84"),
  X85=c("X85"),
  X86=c("X86"),
  X87=c("X87"),
  X88=c("X88"),
  X89=c("X89"),
  X90=c("X90"),
  X91=c("X91"),
  X92=c("X92"),
  X93=c("X93"),
  X94=c("X94"),
  X95=c("X95"),
  X96=c("X96"),
  X97=c("X97"),
  X98=c("X98"),
  X99=c("X99"),
 X100=c("X100"),
 X101=c("X101"),
 X102=c("X102"),
 X103=c("X103"),
 X104=c("X104"),
 X105=c("X105"),
 X106=c("X106"),
 X107=c("X107"),
 X108=c("X108"),
 X109=c("X109"),
 X110=c("X110"),
 X111=c("X111"),
 X112=c("X112"),
 X113=c("X113"),
 X114=c("X114"),
 X115=c("X115"),
 X116=c("X116"),
 X117=c("X117"),
 X118=c("X118"),
 X119=c("X119"),
 X120=c("X120"),
 X121=c("X121"),
 X122=c("X122"),
 X123=c("X123"),
 X124=c("X124"),
 X125=c("X125"),
 X126=c("X126"),
 X127=c("X127"),
 X128=c("X128"),
 X129=c("X129"),
 X130=c("X130"),
 X131=c("X131"),
 X132=c("X132"),
 X133=c("X133"),
 X134=c("X134"),
 X135=c("X135"),
 X136=c("X136"),
 X137=c("X137"),
 X138=c("X138"),
 X139=c("X139"),
 X140=c("X140"),
 X141=c("X141"),
 X142=c("X142"),
 X143=c("X143"),
 X144=c("X144"),
 X145=c("X145"),
 X146=c("X146"),
 X147=c("X147"),
 X148=c("X148"),
 X149=c("X149"),
 X150=c("X150"),
 X151=c("X151"),
 X152=c("X152"),
```

```
X153=c("X153"),
X154=c("X154"),
X155=c("X155"),
X156=c("X156"),
X157=c("X157"),
X158=c("X158"),
X159=c("X159"),
X160=c("X160"),
X161=c("X161"),
X162=c("X162"),
X163=c("X163"),
X164=c("X164"),
X165=c("X165"),
X166=c("X166"),
X167=c("X167"),
X168=c("X168"),
X169=c("X169"),
X170=c("X170"),
X171=c("X171"),
X172=c("X172"),
X173=c("X173"),
X174=c("X174"),
X175=c("X175"),
X176=c("X176"),
X177=c("X177"),
X178=c("X178"),
X179=c("X179"),
X180=c("X180"),
X181=c("X181"),
X182=c("X182"),
X183=c("X183"),
X184=c("X184"),
X185=c("X185"),
X186=c("X186"),
X187=c("X187"),
X188=c("X188"),
X189=c("X189"),
X190=c("X190"),
X191=c("X191"),
X192=c("X192"),
X193=c("X193"),
X194=c("X194"),
X195=c("X195"),
X196=c("X196"),
X197=c("X197"),
X198=c("X198"),
X199=c("X199"),
X200=c("X200"),
X201=c("X201"),
X202=c("X202"),
X203=c("X203"),
X204=c("X204"),
X205=c("X205"),
X206=c("X206"),
X207=c("X207"),
X208=c("X208"),
X209=c("X209"),
X210=c("X210"),
X211=c("X211"),
X212=c("X212"),
X213=c("X213"),
X214=c("X214"),
X215=c("X215"),
X216=c("X216"),
X217=c("X217"))
group.importance(rf.obj, groups)
```

Lampiran 4 Syntax Pemilihan Parameter SVM (*grid search)*

```
#LIBRARY DAN DATA
source("LIBRARY DAN DATA.txt")

#DATA
fold=generatefolds(dataa$Y, nfold=10, stratified = TRUE,
seed =12345)

#FOLDING
#source("PEMBAGIAN FOLD 1-10.txt")

#PEMBAGIAN GAMMA DAN COST
c=c(0.0000001,0.000001,0.00001,0.0001,0.001,0.01,0.1,1,10,10
0,1000,10000,100000,1000000,10000000)
g=c(0.0000001,0.000001,0.00001,0.0001,0.001,0.01,0.1,1,10,10
0,1000,10000,100000,1000000,10000000)
a=expand.grid(c,g)
n=nrow(a)
r=10
error = matrix(0,n,r)
akurasi = matrix(0,n,r)
rata2 = matrix(0,n,1)

set.seed(150000)
#PARAMETER TUNING
for (i in 1:n){
  for(j in 1:10)
  {
    train=dataa[-fold[[j]],]
    test=dataa[fold[[j]],]
    model = svm(Y~., train, scale=F, kernel="radial",
                Gamma=a[i,2],
                cost=a[i,1])
    pred = predict(model, test)
    tablee = table(test$Y, pred)
    akurasi[i,j] = ((tablee[1,1]+tablee[2,2])/sum(tablee))
    #print(akurasi[1])
    error[i,j] = sum(test$Y!= pred)/nrow(test)
    rata2[i,] = mean(akurasi[i,])
    }
}
sink("E:/grid_akurasi_data25_(^-7 sd ^+7.csv")
u=data.frame(a,akurasi,rata2)
u
sink()
```

Lampiran 5 Syntax AdaBoost-SVM

```
adaboostori=function(X,y,X.test,y.test,iterasi,kernel,Gamma=
NULL,C){
  ## kernel yang digunakan hanya rbf dan linear
  if(!is.matrix(X)) X=as.matrix(X)
  if(!is.matrix(X)) X.test=as.matrix(X.test)
     n=nrow(X)
     final.test <- rep(0, length(y.test))
     bobot=rep(1/n,n)
     final.pred=list()
     error=c()
     a=c()
        Sign=function(x,kelas.positif,kelas.negatif){
           tanda=ifelse(x>=0,kelas.positif,kelas.negatif)
           return(tanda)
          }
  for (i in 1:iterasi){
    if(i == 1) { samp=sample(nrow(X), nrow(X), replace =
FALSE) } # no replacement in first iteration
       else if(i != 1) { samp=sample(nrow(X), nrow(X),
replace = TRUE, prob = bobot) }

    X.train=X[samp,]
    row.names(X.train)=NULL
    y.train=y[samp]

    if(length(y.train[y.train==-
1])==0|length(y.train[y.train==1])==0) {
      cat("y train hanya berisi satu kelas","\n")
      a[i]=0
      final.pred[[i]]=matrix(0,ncol=1,nrow=length(y.test))
      break}
      row.names(y.train)=NULL

   if (kernel=="linear"){

weight.svm=svm(x=X.train,y=y.train,scale=F,kernel=kernel,C=C
)
       }else
{
weight.svm=svm(x=X.train,y=y.train,scale=F,kernel=kernel,Gam
ma=Gamma,C=C) }

#weight.svm=wsvm(x=X,y=y,scale=F,kernel="radial",Gamma=Gamma
,C=C,case.weights = bobot)
       pred=predict(weight.svm,X)
       error[i]=sum(bobot*ifelse(pred!=y,1,0))/sum(bobot)

    #cat("error=",error,"\n")
    if (error[i]<=0.000001) { print("iterasi berhenti")
        cat("iterasi=",i,"\n")
```

Lampiran 5 Syntax AdaBoost-SVM (Lanjutan)

```
        cat("error=",error[i],"\n")
     #final.pred=ifelse(colnames(final.pred)=="-1/1",-
final.pred,final.pred)
        a[i]=0
        final.pred[[i]]=matrix(0,ncol=1,nrow=length(y.test))
        break
      } else if (error[i]>=0.49999) {a[i]=0;
bobot=rep(1/n,n)} else {a[i]=(1/2)*log((1-
error[i])/error[i]);bobot=(bobot*exp(-a[i]*ifelse(pred!=y,-
1,1)))/sum(bobot*exp(-a[i]*ifelse(pred!=y,-1,1)))}

final.pred[[i]]=attr(predict(weight.svm,X.test,decision.valu
es=TRUE),"decision.values")
    if(colnames(final.pred[[i]])=="-1/1"){
      final.pred[[i]]=-final.pred[[i]]}
else {final.pred[[i]]=final.pred[[i]]}}
  a=a/sum(a)
  if(is.nan(a[1])) {a=rep(0,length(a))}
  a=as.list(a)

dd=lapply(1:length(final.pred),function(i){final.pred[[i]]*a
[[i]]})
  final.test=do.call("cbind",dd)
  final.test=rowSums(final.test)
  prediksi.kelas=Sign(final.test,1,-1)
  #print(prediksi.kelas)

hasil=list(bobot.final=bobot,fit.y=final.test,prediksi.y=pre
diksi.kelas)
  return(hasil)
}
r=10
fold=generatefolds(data5$Y, nfolds=r, stratified=TRUE,
seed=12345)
TotalAccuracyTest=rep(0,r)
SensTest=rep(0,r)
SpesTest=rep(0,r)
AUCTest=rep(0,r)

#MODEL SVM-ADABOOST
for(i in 1:r)
{
  train=data5[-fold[[i]],]
  test=data5[fold[[i]],]
  model=adaboostori(train[,-1],train[,1],test[,-1],test[,1],
                    iterasi=10,  kernel="radial",
                    C  =1,
                    Gamma =0.1)
  tabel2=table(model$prediksi.y, test[,1])
TotalAccuracyTest[i]=((tabel2[1,1]+tabel2[2,2])/sum(tabel2))
```

Lampiran 5 Syntax AdaBoost-SVM (Lanjutan)

```
SensTest[i]=((tabel2[1,1])/(tabel2[1,1]+tabel2[1,2]))
  SpesTest[i]=((tabel2[2,2])/(tabel2[2,1]+tabel2[2,2]))
  AUCTest[i]=1/2*(SensTest[i]+SpesTest[i])
}

#MENYIMPAN HASIL
sink("akurasi_boosting-svm_data35_n10 (c=1, g=0.1).csv")
TotalAccuracyTest
SensTest
SpesTest
AUCTest
mean(TotalAccuracyTest)
mean(SensTest)
mean(SpesTest)
sink()
```

Lampiran 6 Nama Variabel Prediktor

Simbol	Variabel	Simbol	Variabel	Simbol	Variabel
X1	pKa(max20)	X31	ES_Count_ssCH2	X61	LogD
X2	Br_Count	X32	ES_Count_ssNH	X62	Molecular_Mass
X3	C_Count	X33	ES_Count_ssO	X63	Molecular_Solubility
X4	Cl_Count	X34	ES_Count_sssCH	X64	QED
X5	F_Count	X35	ES_Count_sssN	X65	QED_ALERTS
X6	H_Count	X36	ES_Count_ssssC	X66	QED_ALOGP
X7	I_Count	X37	ES_Sum_aaaC	X67	QED_AROM
X8	N_Count	X38	ES_Sum_aaCH	X68	QED_HBA
X9	O_Count	X39	ES_Sum_aaN	X69	QED_HBD
X10	S_Count	X40	ES_Sum_aaO	X70	QED_MW
X11	ALogP98	X41	ES_Sum_aasC	X71	QED_PSA
X12	ALogP_MR	X42	ES_Sum_ddsN	X72	QED_ROTB
X13	ES_Count_aaaC	X43	ES_Sum_ddssS	X73	QED_Unweighted
X14	ES_Count_aaCH	X44	ES_Sum_dO	X74	SAscore
X15	ES_Count_aaN	X45	ES_Sum_dsN	X75	SAscore_Complexity
X16	ES_Count_aaNH	X46	ES_Sum_dssC	X76	SAscore_Fragments
X17	ES_Count_aaO	X47	ES_Sum_sBr	X77	HBA_Count
X18	ES_Count_aasC	X48	ES_Sum_sCH3	X78	HBD_Count
X19	ES_Count_ddsN	X49	ES_Sum_sCl	X79	NPlusO_Count
X20	ES_Count_ddssS	X50	ES_Sum_sF	X80	Num_Aliphatic DoubleBonds
X21	ES_Count_dO	X51	ES_Sum_sI	X81	Num_Aliphatic SingleBonds
X22	ES_Count_dsN	X52	ES_Sum_sNH2	X82	Num_AromaticBonds
X23	ES_Count_dssC	X53	ES_Sum_sOH	X83	Num_AromaticRings
X24	ES_Count_sBr	X54	ES_Sum_ssCH2	X84	Num_AtomClasses
X25	ES_Count_sCH3	X55	ES_Sum_ssNH	X85	Num_Atoms
X26	ES_Count_sCl	X56	ES_Sum_ssO	X86	Num_Bonds
X27	ES_Count_sF	X57	ES_Sum_sssCH	X87	Num_ChainAssemblies
X28	ES_Count_sI	X58	ES_Sum_sssN	X88	Num_Chains
X29	ES_Count_sNH2	X59	ES_Sum_ssssC	X89	Num_DoubleBonds
X30	ES_Count_sOH	X60	Apol	X90	Num_ExplicitAtoms

Lampiran 6 Nama Variabel Prediktor (Lanjutan)

Simbol	Variabel	Simbol	Variabel	Simbol	Variabel
X91	Num_ExplicitBonds	X121	BIC	X151	SC_2
X92	Num_ExplicitHydrogens	X122	CHI_0	X152	SC_3_C
X93	Num_H_Acceptors	X123	CHI_1	X153	SC_3_P
X94	Num_H_Acceptors_Lipinski	X124	CHI_2	X154	SIC
X95	Num_H_Donors	X125	CHI_3_C	X155	V_ADJ_equ
X96	Num_H_Donors_Lipinski	X126	CHI_3_P	X156	V_ADJ_mag
X97	Num_Hydrogens	X127	CHI_V_0	X157	V_DIST_equ
X98	Num_NegativeAtoms	X128	CHI_V_1	X158	V_DIST_mag
X99	Num_PositiveAtoms	X129	CHI_V_2	X159	Wiener
X100	Num_RingAssemblies	X130	CHI_V_3_C	X160	Zagreb
X101	Num_RingBonds	X131	CHI_V_3_P	X161	Dipole_mag
X102	Num_RingFusionBonds	X132	CIC	X162	Dipole_X
X103	Num_Rings	X133	E_ADJ_equ	X163	Dipole_Y
X104	Num_Rings5	X134	E_ADJ_mag	X164	Dipole_Z
X105	Num_Rings6	X135	E_DIST_equ	X165	Jurs_DPSA_1
X106	Num_Rings7	X136	E_DIST_mag	X166	Jurs_DPSA_2
X107	Num_RotatableBonds	X137	IAC_Mean	X167	Jurs_DPSA_3
X108	Num_SingleBonds	X138	IAC_Total	X168	Jurs_FNSA_1
X109	Num_StereoAtoms	X139	IC	X169	Jurs_FNSA_2
X110	Num_StereoBonds	X140	JX	X170	Jurs_FNSA_3
X111	Num_TerminalRotomers	X141	JY	X171	Jurs_FPSA_1
X112	Num_TrueStereoAtoms	X142	Kappa_1	X172	Jurs_FPSA_2
X113	Organic_Count	X143	Kappa_1_AM	X173	Jurs_FPSA_3
X114	Molecular_FractionalPolarSASA	X144	Kappa_2	X174	Jurs_PNSA_1
X115	Molecular_Fractional PolarSurfaceArea	X145	Kappa_2_AM	X175	Jurs_PNSA_2
X116	Molecular_PolarSASA	X146	Kappa_3	X176	Jurs_PNSA_3
X117	Molecular_PolarSurfaceArea	X147	Kappa_3_AM	X177	Jurs_PPSA_1
X118	Molecular_SASA	X148	PHI	X178	Jurs_PPSA_2
X119	Molecular_SAVol	X149	SC_0	X179	Jurs_PPSA_3
X120	Molecular_SurfaceArea	X150	SC_1	X180	Jurs_RASA

Lampiran 6 Nama Variabel Prediktor (Lanjutan)

Simbol	Variabel
X181	Jurs_RNCG
X182	Jurs_RNCS
X183	Jurs_RPCG
X184	Jurs_RPCS
X185	Jurs_RPSA
X186	Jurs_SASA
X187	Jurs_TASA
X188	Jurs_TPSA
X189	Jurs_WNSA_1
X190	Jurs_WNSA_2
X191	Jurs_WNSA_3
X192	Jurs_WPSA_1
X193	Jurs_WPSA_2
X194	Jurs_WPSA_3
X195	AverageBondLength
X196	Energy
X197	Minimized_Energy
X198	RadOfGyration
X199	Strain_Energy
X200	PMI_mag
X201	PMI_X
X202	PMI_Y
X203	PMI_Z
X204	Shadow_nu
X205	Shadow_Xlength
X206	Shadow_XY
X207	Shadow_XYfrac
X208	Shadow_XZ
X209	Shadow_XZfrac
X210	Shadow_Ylength
X211	Shadow_YZ
X212	Shadow_YZfrac
X213	Shadow_Zlength
X214	Molecular_3D_PolarSASA
X215	Molecular_3D_SASA
X216	Molecular_3D_SAVol
X217	Molecular_Volume

Lampiran 7 Hasil *Feature Importance Mean Decrease Gini*

Variabel	MeanDecreaseGini	No
X197	0.30750553	1
X162	0.302821277	2
X199	0.289342084	3
X74	0.258406581	4
X73	0.224454547	5
X214	0.215447325	6
X196	0.210349878	7
X176	0.204831161	8
X1	0.191980492	9
X174	0.187582922	10
X163	0.185820849	11
X175	0.183255779	12
X161	0.183169236	13
X173	0.182364207	14
X125	0.181668487	15
X66	0.179977265	16
X56	0.177444741	17
X121	0.171006664	18
X167	0.169944467	19
X71	0.16580681	20
X139	0.164923816	21
X131	0.163123614	22
X39	0.161916684	23
X33	0.160855935	24
X191	0.159293955	25
X37	0.158611922	26
X114	0.156947917	27
X64	0.155681158	28
X166	0.15549796	29
X184	0.155311949	30
X190	0.154213809	31
X172	0.15183879	32
X53	0.1517051	33
X188	0.149577824	34
X141	0.148093276	35
X170	0.146690964	36
X169	0.145867081	37
X137	0.144708897	38
X154	0.144258713	39
X201	0.142997338	40

Lampiran 7 Hasil *Feature Importance Mean Decrease Gini* (Lanjutan)

Variabel	MeanDecreaseGini	No
X182	0.141635511	41
X181	0.141513385	42
X117	0.141365947	43
X185	0.140156741	44
X116	0.140009514	45
X143	0.137869227	46
X43	0.13729293	47
X61	0.136844309	48
X171	0.133978871	49
X204	0.131413134	50
X62	0.131114547	51
X115	0.130918114	52
X132	0.130321036	53
X138	0.128814928	54
X11	0.128748732	55
X168	0.128710727	56
X129	0.125691771	57
X6	0.1245518	58
X189	0.123590911	59
X179	0.123184868	60
X164	0.1218371	61
X165	0.121001754	62
X194	0.119894112	63
X183	0.119432336	64
X180	0.119086269	65
X211	0.118596313	66
X38	0.118084536	67
X135	0.116726122	68
X76	0.114844585	69
X130	0.113906231	70
X75	0.113636974	71
X178	0.11321062	72
X193	0.111851207	73
X84	0.110629563	74
X145	0.110048176	75
X195	0.109535115	76
X207	0.109408247	77
X177	0.107593972	78
X210	0.106637912	79
X160	0.106097725	80

Lampiran 7 Hasil *Feature Importance Mean Decrease Gini* (Lanjutan)

Variabel	MeanDecreaseGini	No
X147	0.104836204	81
X124	0.104503845	82
X203	0.103582272	83
X70	0.103345338	84
X41	0.102762507	85
X215	0.10174492	86
X202	0.101725877	87
X44	0.100640441	88
X206	0.100279206	89
X198	0.099703124	90
X127	0.099594918	91
X72	0.096145704	92
X212	0.095994425	93
X128	0.095627478	94
X136	0.095297964	95
X60	0.095245308	96
X216	0.093760672	97
X48	0.093460113	98
X120	0.093176466	99
X123	0.092542798	100
X133	0.09178326	101
X208	0.091009577	102
X205	0.0906718	103
X126	0.090583459	104
X140	0.089823437	105
X186	0.089535389	106
X92	0.088245817	107
X79	0.087879271	108
X187	0.086409465	109
X58	0.085602625	110
X69	0.084838395	111
X152	0.082479387	112
X158	0.082339486	113
X153	0.081254452	114
X157	0.081131298	115
X151	0.079895363	116
X192	0.078108645	117
X209	0.077258849	118
X119	0.077084884	119
X213	0.076343126	120

Lampiran 7 Hasil *Feature Importance Mean Decrease Gini* (Lanjutan)

Variabel	MeanDecreaseGini	No
X89	0.075966864	121
X146	0.075690235	122
X80	0.07475453	123
X97	0.073513	124
X77	0.073084269	125
X107	0.072450063	126
X217	0.07196275	127
X85	0.071801815	128
X148	0.07122884	129
X54	0.0710629	130
X159	0.07086381	131
X63	0.070539663	132
X88	0.068662382	133
X108	0.067083899	134
X30	0.065967365	135
X144	0.064353406	136
X118	0.063930487	137
X91	0.061260233	138
X200	0.060648744	139
X78	0.058978116	140
X94	0.058103699	141
X95	0.057434662	142
X122	0.056035966	143
X21	0.055065685	144
X12	0.053975772	145
X55	0.053916376	146
X155	0.053406522	147
X35	0.051031274	148
X25	0.049814021	149
X90	0.049653881	150
X134	0.047655161	151
X86	0.046540415	152
X96	0.046508024	153
X9	0.04649129	154
X81	0.045667349	155
X3	0.044617299	156
X18	0.044201251	157
X8	0.04354954	158
X93	0.041905479	159
X142	0.041737315	160

Lampiran 7 Hasil *Feature Importance Mean Decrease Gini* (Lanjutan)

Variabel	MeanDecreaseGini	No
X156	0.040948737	161
X49	0.040501158	162
X113	0.038891458	163
X10	0.038229922	164
X149	0.037372361	165
X20	0.030505002	166
X87	0.029956753	167
X150	0.029700285	168
X68	0.028168049	169
X14	0.026064934	170
X100	0.024105353	171
X31	0.023566255	172
X32	0.023298858	173
X65	0.01584089	174
X15	0.014614188	175
X99	0.014297106	176
X111	0.014101991	177
X47	0.013343152	178
X42	0.010641254	179
X4	0.010547184	180
X98	0.010053141	181
X102	0.009942824	182
X105	0.009569954	183
X82	0.008439868	184
X101	0.007917425	185
X26	0.007673765	186
X28	0.007607369	187
X83	0.007305361	188
X13	0.006398327	189
X2	0.006260387	190
X51	0.006043478	191
X46	0.005712716	192
X103	0.00489268	193
X67	0.003671057	194
X24	0.003522618	195
X34	0.003063483	196
X57	0.00243246	197
X36	0.00234375	198
X52	0.002142857	199
X112	0.002	200

Lampiran 7 Hasil *Feature Importance Mean Decrease Gini* (Lanjutan)

Variabel	MeanDecreaseGini	No
X7	0.001889746	201
X22	0.001875	202
X29	0.001300054	203
X106	0.001125	204
X19	0.001080357	205
X23	0.001056287	206
X5	0	207
X16	0	208
X17	0	209
X27	0	210
X40	0	211
X45	0	212
X50	0	213
X59	0	214
X104	0	215
X109	0	216
X110	0	217

Lampiran 8 Hasil LORENS partisi 30 *threshold* optimal

```
>modelcv<-lr.cerp.cv(y, x, nens=11, v_fold=10,
          fixsize=30, fixthres=NULL, search=FALSE)
> modelcv
$probability
 [1]   0.47365135   0.37124479   0.53290155   0.56786276
       0.53344996   0.54890454   0.27373990
 [8]   0.18421841   0.34288876   0.63806466   0.64603207
       0.53888079   0.58101197   0.62619904
[15]   0.37923173   0.40542955   0.64131217   0.62069557
       0.58111712   0.32468014   0.41582928
[22]   0.37929817   0.62159226   0.61240999   0.63850191
       0.52732808   0.63614432   0.52835670
[29]   0.16490053   0.43292983   0.70471828   0.66825626
       0.74300221   0.63339501   0.21176911
[36]   0.23838824   0.81335881   0.52939364   0.55875324
       0.57354718   0.45725609   0.25277902
[43]   0.61289783   0.65361464   0.80636641   0.82948616
       0.73193139   0.60324313   0.50023942
[50]   0.39241511   0.39677475   0.58515993   0.59492336
       0.66672686   0.68937127   0.60616567
[57]   0.57468644   0.71563710   0.57161005   0.35213687
       0.29554171   0.20548474   0.40625901
[64]   0.25007253   0.05020194   0.60289064   0.74950791
       0.62145865   0.59031770   0.60327596
[71]   0.69232868   0.44900586   0.64129520   0.41665497
       0.35239523   0.57958275   0.11421782
[78]   0.65268176   0.16164123   0.57191135   0.18387766
       0.58780233   0.24113907   0.42043209
$predicted
 [1] 0 0 1 1 1 1 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 0 1 1 1
     1 1 1 0 0 1 1 1 1 0 0 1 1 1 1
[41] 0 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0
     1 1 1 1 1 1 0 1 0 0 1 0 1 0 1
[81] 0 1 0 0
$partition.size.list
 [1] 30 30 30 30 30 30 30 30 30 30
$threshold.list
 [1] 0.5100000 0.4900000 0.5033333 0.4900000 0.5065789
     0.5065789 0.4934211 0.5000000
 [9] 0.5131579 0.4868421
$decision.table
         pred.pos pred.neg
real.pos       34        8
real.neg       18       24
```

Lampiran 9 Hasil LORENS partisi 30 *threshold* 0.5

```
>modelcv<-lr.cerp.cv(y, x, nens=11, v_fold=10,
          fixsize=30, fixthres=0.5, search=FALSE)
> modelcv
$probability
 [1]   0.41529091   0.31681919   0.52301809   0.50280513
       0.57031979   0.48706176   0.29391168
 [8]   0.21230231   0.26973921   0.63409681   0.60625966
       0.55083802   0.56552036   0.63939220
[15]   0.41025570   0.37172031   0.65166424   0.60664818
       0.59583561   0.34573389   0.42907346
[22]   0.30290743   0.63074958   0.63004268   0.58260442
       0.51454789   0.69305118   0.51679296
[29]   0.13445252   0.55614180   0.65321934   0.75288678
       0.71026425   0.61409885   0.21154411
[36]   0.22700487   0.82372706   0.53990452   0.54763256
       0.54183380   0.47234695   0.30548799
[43]   0.55039760   0.62708923   0.79513990   0.82021658
       0.74084230   0.64292697   0.54280181
[50]   0.46911039   0.43121758   0.52525235   0.62201292
       0.61113339   0.70984247   0.62585466
[57]   0.58155840   0.70760726   0.58638510   0.35633844
       0.26883055   0.19197528   0.33251313
[64]   0.21796688   0.06618832   0.57884667   0.60455795
       0.59429910   0.57695947   0.60156395
[71]   0.64974091   0.44943949   0.62072081   0.40910348
       0.38459750   0.63589463   0.11332318
[78]   0.53364218   0.16561944   0.57415135   0.11519240
       0.58372888   0.22112838   0.35109044
$predicted
 [1] 0 0 1 1 1 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 0 1 1 1
     1 1 1 0 1 1 1 1 1 0 0 1 1 1 1
[41] 0 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0
     1 1 1 1 1 1 0 1 0 0 1 0 1 0 1
[81] 0 1 0 0
$partition.size.list
 [1] 30 30 30 30 30 30 30 30 30 30
$threshold.list
 [1] 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
$decision.table
         pred.pos pred.neg
real.pos       34        8
real.neg       18       24
```

Lampiran 10 Surat Pernyataan Data

BIODATA PENULIS

Penulis dengan nama lengkap Erlin Sukmaputri dilahirkan di Malang pada 06 Maret 1996. Penulis menempuh pendidikan formal di SDN Kebon Kosong 15 Jakarta, SMPN 216 Jakarta, dan SMAN 21 Jakarta. Kemudian penulis diterima sebagai Mahasiswa Departemen Statistika ITS melalui jalur SBMPTN pada tahun 2014. Selama masa perkuliahan, penulis aktif di berbagai kepanitiaan salah satunya adalah panitia Penanggung Jawab *Statistics Competition* (STATION) 2016 untuk Region Jakarta yang merupakan olimpiade statistika bagi murid SMA dan sederajat. Selain itu, penulis juga aktif dalam organisasi yang menaungi Departemen Statistika yaitu sebagai staff Minat dan Bakat HIMASTA-ITS 2015/2016, staff Kominfo UKM Bola Basket 2015/2016, Sekretaris Kesenian dan Olahraga HIMASTA-ITS 2016/2017. Apabila pembaca ingin memberi kritik dan saran serta diskusi lebih lanjut mengenai Tugas Akhir ini, dapat menghubungi penulis melalui email erlinsukmaputri@gmail.com atau nomor telepon 081218189007.